DMA_Mem_to_Mem_1 for KIT_AURIX_TC375_LK

DMA transfer between memories

AURIX™ TC3xx Microcontroller Training
V1.0.1

Scope of work

The DMA is used to transfer ten words (32-bit) of data from one memory location to another without any CPU load.

The transfer of data is triggered by SW. The source is the Data Scratch Pad SRAM of CPU0 (DSPR0) and the destination is the Distributed Local Memory Unit (DLMU RAM). At the end of the transactions, the data is verified by comparing the source and destination buffers. The success of the data transfer is signaled through the LED connected to pin 6 of port 00. Otherwise, the LED connected to pin 5 of port 00 is used.
The same cycle is repeated each second.

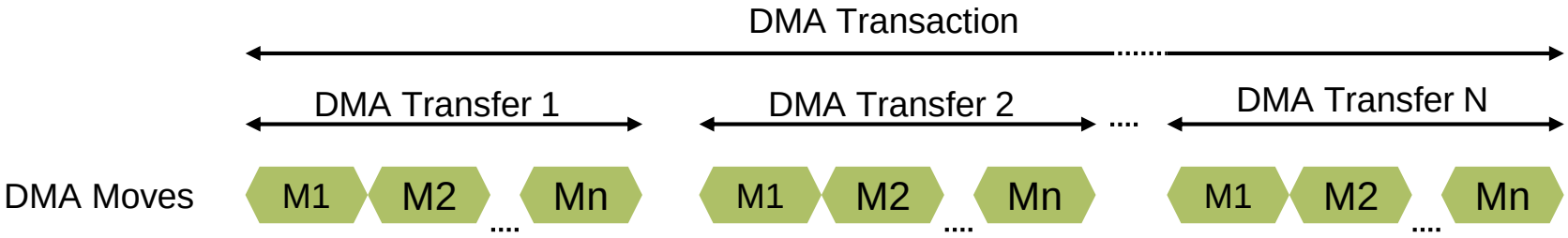
Introduction

- The Direct Memory Access (DMA) unit is a module which can execute data transfers from a source memory to a destination memory without any CPU load.

- The DMA controller mainly supports:
  - Two move engines for the parallel execution of DMA requests
  - Individually programmable DMA channels (up to 128)
    - DMA Channel 127 has the highest priority
  - DMA requests can be triggered by Hardware or Software
    - Any peripheral that can trigger an interrupt can initiate a DMA transfer

Introduction

DMA Move, Transfer, Transaction:



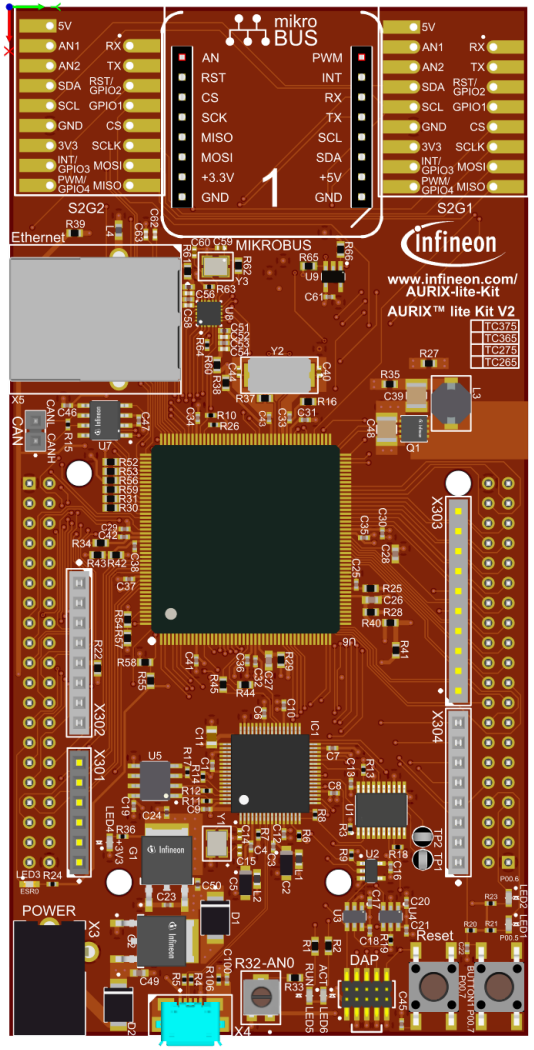
- A **DMA Move** is a Bus read **and** write operation
  - Supported data widths for DMA read & write moves: 8, 16, 32, 64, 128 or 256-bit
- A **DMA Transfer** consists of a configurable number of DMA moves
  - It can be composed of 1, 2, 3, 4, 5, 8, 9 or 16 DMA moves
- A **DMA Transaction** consists of several (at least one) DMA Transfers
  - It is possible to trigger the full DMA transaction or each DMA transfer of the transaction in order

Note:

- A **DMA Transfer** is an **un-interruptable** DMA operation
- Long **DMA Transfers** can block pending DMA Channels with **higher** priority

Hardware setup

This code example has been developed for the board KIT_A2G_TC375_LITE.



Implementation

Specify data storage for Source and Destination buffers

The ***__at*** Tasking compiler attribute is used to declare data buffers in specific memory locations:

› Source data buffer in DSPR0:

uint32 g_dataForDmaTransfer[DATA_ARRAY_LENGTH] __at(0x70000000);

› Destination data buffer in LMURAM (non cached memory)

uint32 g_dmaLmuDestination[DATA_ARRAY_LENGTH] __at(0xB0000000);

Implementation

All used iLLD functions for initializing and controlling DMA transfers are provided by the ***IfxDma_Dma.h*** header file

DMA configuration

Before the first DMA data transfer can be requested and executed, the DMA module has to be initialized. The following steps are done inside ***initDMA()***:

1. Load default module configuration into DMA configuration structure:
   IfxDma_Dma_initModuleConfig(&g_DMA.dmaConfig, &MODULE_DMA)
2. Apply default configuration on DMA hardware module:
   IfxDma_Dma_initModule(&g_DMA.dmaHandle, &g_DMA.dmaConfig)
3. Load the DMA default channel configuration:
   IfxDma_Dma_initChannelConfig(&g_DMA.dmaChNCfg, &g_DMA.dmaHandle)

Implementation

DMA configuration (Cont.)

4. Modify the channel configuration to fit the use case:
   - **DMA channel ID:** 0
   - **DMA move data width:** 32-bit
   - **DMA Transfer count:** 10
   - **DMA Transaction request mode:** Complete the transaction on each request
5. Apply configuration to DMA hardware channel in DMARAM
   IfxDma_Dma_initChannel(&g_DMA.dmaChannel, &g_DMA.dmaChNCfg)

Implementation

LEDs Configuration and Control

To provide status signals, two LEDs of the Application Kit board are used:
Failure signal, LED driven by port 00 pin 5:
#define LED_DMA_FAILURE &MODULE_P00,5
Success signal, LED driven by port 00 pin 6:
#define LED_DMA_SUCCESS &MODULE_P00,6

1. Set each used Port Pin as push-pull output with the ***IfxPort_setPinMode()*** iLLD function
2. The LEDs are low active:
   - Switch On LED: ***IfxPort_setPinLow()***
   - Switch Off LED : ***IfxPort_setPinHigh()***

Note: Two wrapper functions are implemented (***turnLEDOn()*** & ***turnLEDOff()***) to switch On/Off LEDs, e.g. ***turnLEDOn(LED_DMA_SUCCESS)***.

All port functions used to initialize and switch LEDs‘ state are provided in the iLLD header file ***IfxPort.h***.

Implementation

Request and verify a DMA data transfer

The following steps are done inside ***runDMA()*****,** based on the previous described DMA configuration:

1. Set the DMA source and destination buffers beginning addresses:
   - Source buffer: ***g_DMA.pSourceAddressForDmaTransfer***
   - Destination buffer: ***g_DMA.pDestinationAddressForDmaTransfer***
2. Trigger a DMA Software request:
   - ***IfxDma_Dma_startChannelTransaction()***
3. Poll for the DMA Channel end transfer flag to be set:
   - ***IfxDma_Dma_getAndClearChannelInterrupt()***
4. Verify each copied data byte (this is not an iLLD provided function):
   - ***verifyDMACopiedData()***
5. Set status LED according to the result.

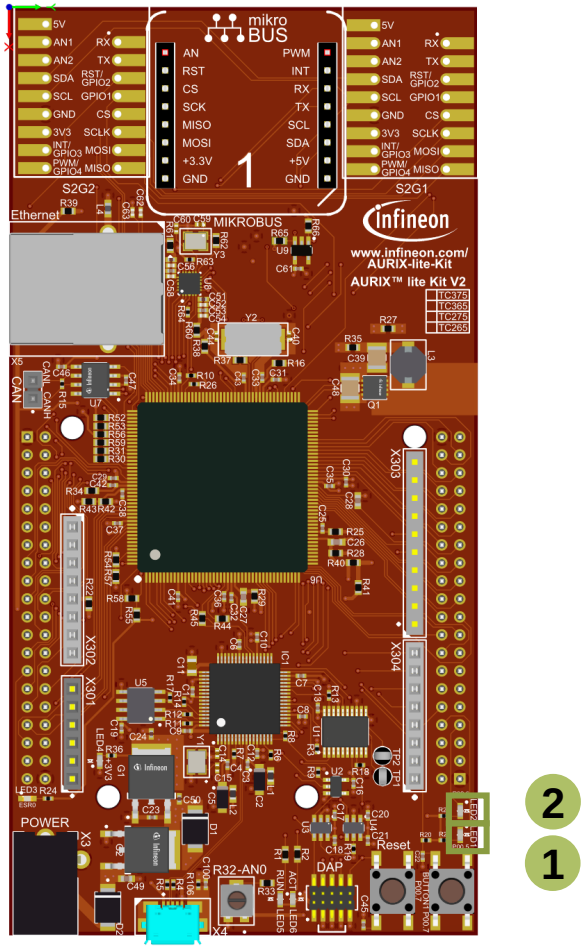
Note: ***runDMA()*** is called inside the infinite loop of the main function and executed every one second (the STM timer is used to ensure the one second delay).

Run and Test

LED signal interpretation after code compilation and device flashing:

- If a data mismatch is detected after the last DMA Transaction only FAILURE LED (1) will be ON.
- Otherwise only SUCCESS LED (2) will be ON.

- The user can watch the evolution of ***successfulDmaTransaction*** and ***failedDmaTransaction*** parameters of the global variable ***g_DMA***:
  - ***g_DMA.successfulDmaTransaction*** increments in case of a successeful DMA transaction
  - ***g_DMA.failedDmaTransaction*** increments in case of a failing DMA transaction



References

- AURIX™ Development Studio is available online:
- https://www.infineon.com/aurixdevelopmentstudio
- Use the *„Import...“* function to get access to more code examples.

- More code examples can be found on the GIT repository:
- https://github.com/Infineon/AURIX_code_examples

- For additional trainings, visit our webpage:
- https://www.infineon.com/aurix-expert-training

- For questions and support, use the AURIX™ Forum:
- https://www.infineonforums.com/forums/13-Aurix-Forum

Revision history

Revision	Description of change
V1.0.1	Updated description: fixed used memory (DLMU RAM instead of LMU RAM)
V1.0.0	Initial version

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2021-12
Published by
Infineon Technologies AG
81726 Munich, Germany

Do you have a question about this document?
Email: erratum@infineon.com

Document reference
DMA_Mem_to_Mem_1_KIT_TC375_LK

IMPORTANT NOTICE
The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenheitsgarantie") .

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

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

WARNINGS
Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.