**Adrien CELLIER**

**Nanotechnologies for ICTs Master**
**2018**

**EPFL Microelectronic Systems Laboratory**
**EPFL STI IEL LSM**
**ELD 340 – Station 11**
**CH-1015 Lausanne, Suisse**

# An Embedded Data Acquisition System with Leon Processor and Nand Flash for Wireless Sensor Networks

from 26/03/18 to 12/08/18

Confidentiality: no

**Under the supervision of :**

- **Company supervisor : Prof. Yusuf LEBLEBICI**
  **yusuf.leblebici@epfl.ch**

  Present at the defense: represented by Mr. Selman ERGÜNAY

- **Phelma Tutor : Prof. Lorena ANGHEL**

**Ecole nationale supérieure de physique, électronique, matériaux**

**Phelma**
Bât. Grenoble INP - Minatec
3 Parvis Louis Néel - CS 50257
F-38016 Grenoble Cedex 01

Tél +33 (0)4 56 52 91 00
Fax +33 (0)4 56 52 91 03

**http://phelma.grenoble-inp.fr**

# Acknowledgments

I would like to thank Professor Leblebici for proposing me this Master thesis in the nice place that is LSM. I would also like to thank him for the culture he brought me and the books I devoured eagerly.

I would like to thank Mr. Ergünay for his essential guidance and help throughout this project.

Finally I would like to thank Professor Vachoux, Mr. Narinx and Mr. Çelik for the help and answers they provided me.

# Abstract (English)

This report presents implementations of a system based on a 32-bit LEON processor for Virtex7 FPGA VC707 and ASIC 65nm, and concentrates on the direct memory access controller component. The project mostly relies on the code and tools from the library GRLIB from Cobham Gaisler AB. This work has been performed for the Microelectronic Systems Laboratory (LSM) at the Ecole Polytechnique Fédérale de Lausanne (EPFL).

This document describes how the direct memory access controller GRDAMC from Cobham Gaisler works and includes a hint on how to write a software for it. The implementation steps for FPGA and ASIC are described. Both implementations have been stopped at analysis step by unsolved errors.

Using the open source GRLIB library some errors have been triggered and solved. They have been reported to Cobham Gaisler AB.

# Résumé (Français)

Ce rapport présente les implémentations d'un système basé sur un processeur 32 bit LEON pour Virtex7 FPGA VC707 et ASIC 65nm. Il se concentre particulièrement sur le contrôleur d'accès direct mémoire. Ce projet fait amplement usage du code VHDL et des outils de la bibliothèque GRLIB développée par Cobham Gaisler AB. Il a été réalisé au Laboratoire de Systèmes Microélectroniques (LSM) de l'Ecole Polytechnique Fédérale de Lausanne (EPFL).

Ce document décrit le fonctionnement du controlleur d'accès direct mémoire et donne quelques conseils sur la manière de le programmer. Les différentes étapes des implémentations sont décrites. Les deux implémentations ont été arrêtées pendant l'analyse par des erreurs non résolues.

L'utilisation de la librarie libre GRLIB a permi d'y identifier des erreurs qui ont du être résolues. Les modifications apportées ont été signalées à Cobham Gaisler AB.

# Sommario (Italiano)

Il presente rapporto presenta delle implementazioni di un sistema basato su un processore a 32 bit "LEON" per FPGA Virtex7 VC707 e tecnologia ASIC a 65nm. Si concentra in particolar modo sul controllore di accesso diretto in memoria. Questo progetto fa ampio uso di descrizione hardware VHDL e degli strumenti della libreria GRLIB, sviluppata dalla società Cobham Gaisler AB. Questo lavoro è stato realizzato al Laboratorio di sistemi microelettronici (Laboratoire de Systèmes Microélectroniques, LSM) del Politecnico Federale di Losanna (EPFL).

Questo documento descrive il funzionamento del controllore di accesso diretto in memoria e fornisce qualche consiglio sul modo di programmarlo. I differenti passaggi dell'implementazione sono descritti. Entrambe le implementazioni sono state fermate durante l'analisi a causa di errori non risolti.

L'utilizzo della libreria libera GRLIB ha permesso di identificare degli errori, che sono stati in seguito risolti. Le modifiche apportate sono state segnalate a Cobham Gaisler AB.

# Table of Contents

# Glossary

ADC: Analog to Digital Converter, a component which converts an analog signal into a digital signal.

AHB: Advanced High-performance Bus, an AMBA high performance bus.

AMBA: ARM Advanced Microcontroller Bus Architecture, a widely used open standard for on-chip interconnect.

APB: Advanced Peripheral Bus, an AMBA bus designed for low bandwidth control accesses.

ASIC: Application-Specific Integrated Circuit, integrated circuit designed for a specific use.

CPU: Central Processing Unit.

DMA: Direct Memory Access, feature that allows memory access without involving the CPU.

EDA: Electronic Design Automation, software tools for designing electronic systems.

FPGA: Field Programmable Gate Array, a fully re-configurable integrated circuit.

GNU GPL: GNU General Public License.

GRLIB: Cobham Gaisler AB VHDL IP core library. Cobham Gaisler is a subsidiary of Cobham plc.

GUI: Graphical User Interface.

IP: Intellectual Property.

LEON, LEON3, LEON3v3: a General Public License 32-bit processor with a SPARC-V8 architecture developed by Cobham Gaisler AB. Cobham Gaisler AB is a subsidiary of Cobham plc.

Virtex7 FPGA VC707: A Xilinx FPGA.

VLSI: Very Large Scale Integration, process of creating an integrated circuit by combining hundreds of thousands of transistors or devices into a single chip.

# List of figures

# Introduction

The Microelectronic Systems Laboratory (LSM) is a part of the Institute of Electrical Engineering (IEL) at the Ecole Polytechnique Fédérale de Lausanne (EPFL) located in Switzerland. Directed by Professor Yusuf LEBLEBICI, most of its 25 members are research assistant. They concentrate on the design and implementation of high-performance digital and mixed-signal VLSI circuits, language-based modeling and validation of system-on-chip components, neuromorphic / bio-inspired system architectures, and integration of novel technologies for complex system design.

The facilities of the Microelectronic Systems Laboratory include a state-of-the-art EDA environment for VLSI design, as well as test and measurement labs for detailed chip-level and system-level characterization.

The number of applications which use embedded systems is soaring, and so does its market. The LSM has been working on 360° camera systems since 2009. Many systems of different size have been developed using several cameras to screen the space all around. In this perspective, the lab is now developing low power smart sensor nodes with local processing to be used in a network. The different parts of the smart sensor nodes are slowly designed and assembled together. The lab is exploring the relevance of Cobham Gaisler LEON3 processor for its embedded systems. In parallel an analog to digital converter (ADC) is being designed to be used in the smart node sensor. Other parts such as a custom memory will eventually improve it.

The ADC will provide a continuous flow of data from the camera. The main processor will not be able to run the instructions and fetch the data from the ADC at the same time. That is why the system will require a direct memory access controller. This component will copy into the main memory the continuously updated signals from the camera leaving the main processor free to process the data. The goal of this master thesis is to develop a low power system with a LEON3 processor and a direct memory access controller.

# I. Project Definition

## 1. State of the project

The LEON3 processor is a 32-bit RISC processor compliant with the SPARC V8E architecture. It is available in the IP cores library GRLIB, which is upgraded roughly every year. The library, except for specific components, is available under the GNU General Public License and also under a commercial license.

The previous work with the LEON3 processor had as a purpose to work on the open source code of the processor and implement it on FPGA to check its usability and relevance to the embedded applications of the laboratory. Mr. Yoann Biard implemented a system for Virtex7 FPGA VC707 (an FPGA sold by Xilinx) using mainly the GRLIB library from January 2012. The system is displayed hereunder on figure 1.

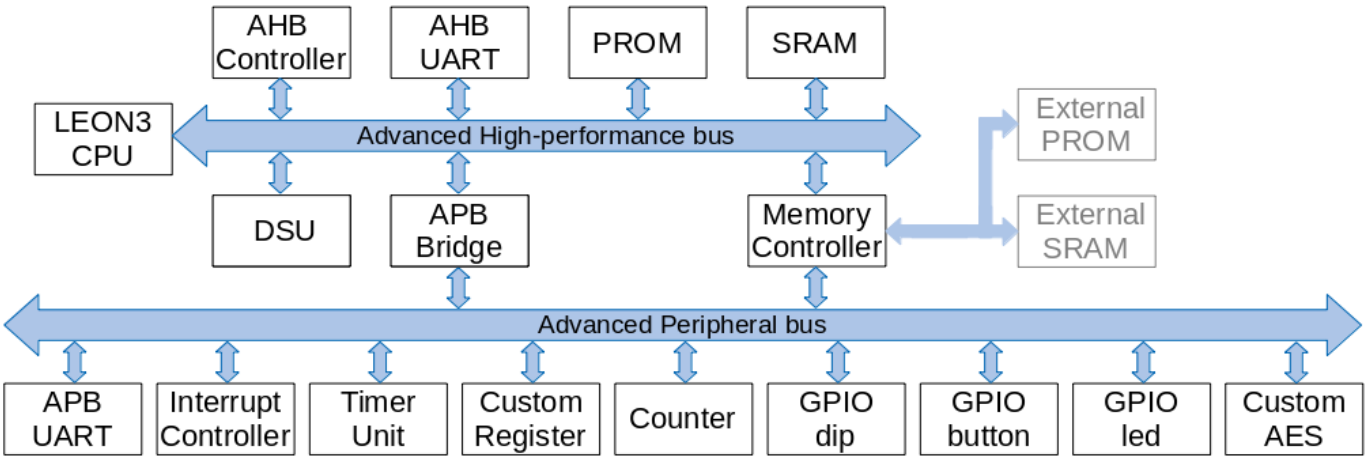


*Figure 1: Working base: a LEON3 design for Virtex7 FPGA VC0707*

This system uses the AMBA2.0 standard with 5 custom signals. The AHB and APB bridge control the buses. The DSU is the debug support unit, and the UART are APB and AHB serial debug interfaces. The PROM and SRAM are memories. GPIO stands for general-purpose input/output. The AES, for Advanced Encryption Standard, is an encryption component. The AES will be required when wireless transmission will be implemented.

## 2. Goals

A direct memory access (DMA) is a method of transferring data from/to a memory without requiring the CPU to process the memory access. A third party DMA uses a specific compponent called DMA controller to perform all DMA data transfers.

In a smart sensor node, an ADC will provide a continuous flow of data. It is advised to use a direct memory access controller to handle the memory access instead of processing them with the CPU. The purpose

of this master thesis is to add and investigate a DMA controller component to a system like the one described on the working base, i.e. to:

- add a DMA controller to the working base and configure the system to provide a working FPGA Virtex7 VC707 (figure 2).
- provide a working ASIC (figure 3) with a LEON3 processor, a DMA controller and maybe the ADC controller which is being designed in parallel.

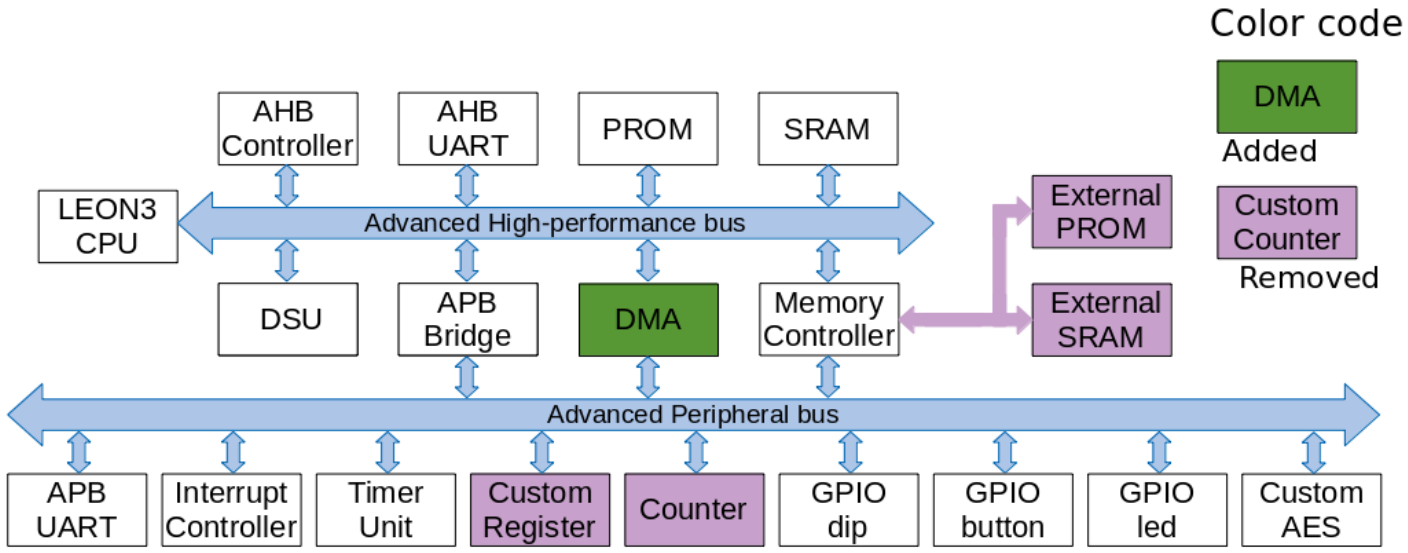


*Figure 2: Goal 1/2, an FPGA design*

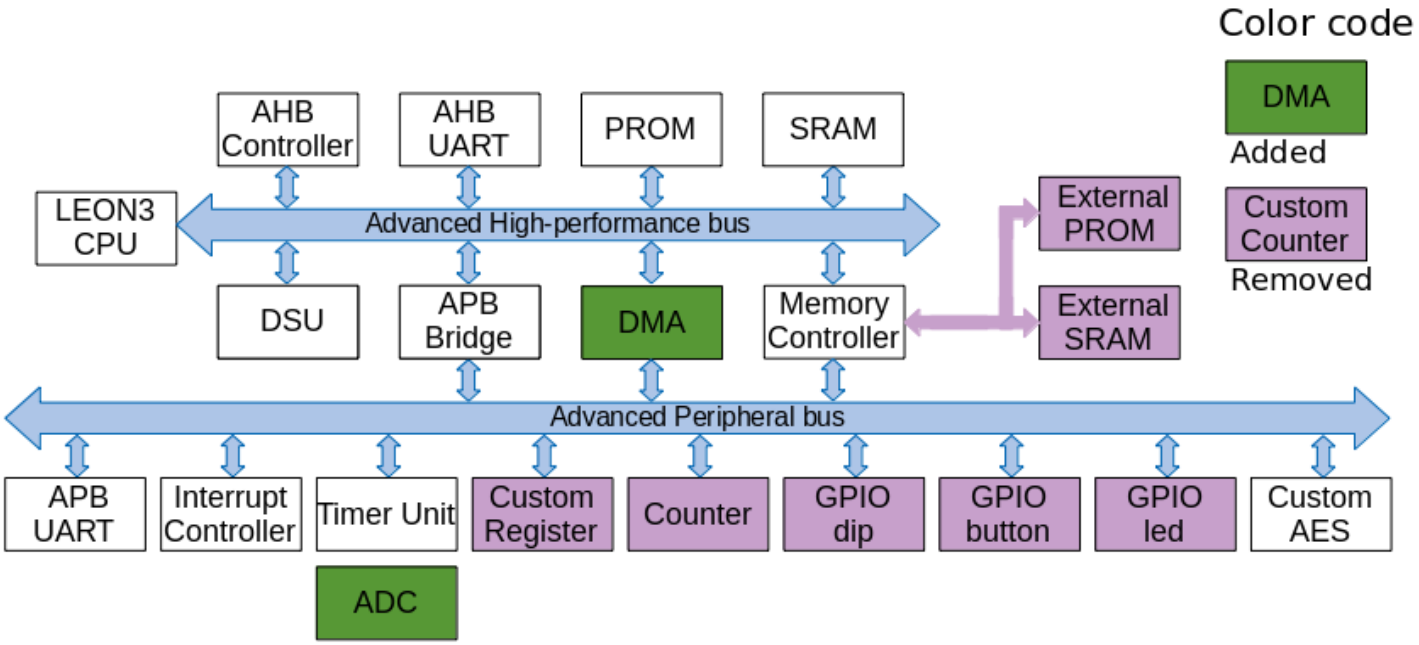


*Figure 3: Goal 2/2, an ASIC design*

# II. DMA

## 1. Generalities

Three modes of DMA operations can be distinguished: In burst mode, the DMA controller is granted control over the system bus, performs the transfer then frees the system bus for other components such as the CPU. In this mode the CPU can not have access to the system bus during the transfer duration. In cycle stealing mode, the DMA control over the system bus is deasserted after a byte of data has been transferred, enabling an other component to request control over the bus. In transparent mode, the CPU have priority for the control over the system bus, limiting the DMA transfers to cycles when the CPU is not using the system bus.

## 2. GRDMAC Overview

The available DMA solutions have been screened. The DMA controller which has been chosen is GRDMAC from COBHAM GAISLER AB. It is part of the GRLIB since 2016, and thus would be easy to add to the system given the highly configurable GRLIB environment. It is under GNU GPL license. It is a very satisfying DMA, that can perform bursts of data at aligned or unaligned memory addresses. It is possible to finely tune the interrupt on completion and it has detailed error flags. It has a configurable buffer size and one to sixteen DMA channels can be supported. As shown on its block diagram (figure 4) its configuration registers are accessible through an APB interface. This feature enables to modify its configuration while running a program using the CPU or a debug support unit. It is possible to add conditions to the DMA activation: it can poll a status register or monitor a triggering line such as an interrupt signal.

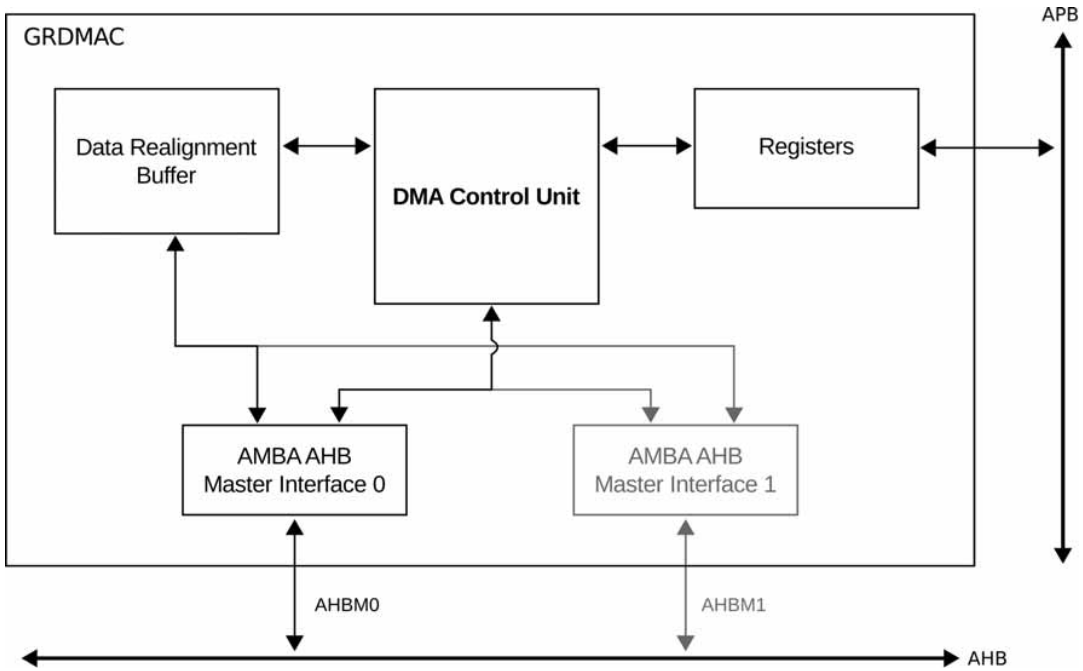


*Figure 4: GRDMAC block diagram*

*from [1] GRIP, Dec 2017, Version 2017.3, Cobham Gaisler AB*

# 3. GRDMAC Setup

The GRDMAC performs two types of operations: it bursts the data from the memory to its buffer (M2B) then bursts it from its buffer to the memory (B2M). Each of these operations is controlled by a list of 16 bytes descriptors. The GRDMAC performs the operations described on the M2B list of descriptors until it is completed or the buffer is full. Then it performs the operations described on the B2M list of descriptors. It will then switch back to the M2B list when the buffer is empty. [1]

During the component instantiation a generic parameter defines the mode of operation: simplified or normal. In simplified mode of operation, the GRDMAC configuration resides entirely in its configuration registers, which are shown on figure 5. The M2B and B2M lists of descriptors are then limited to only one descriptor. Their APB offset is respectively 0x20-0x2F and 0x30-0x3C on figure 5. These descriptors are called data descriptors. They contain the address of the next descriptor of the list (here NULL, for there is no next descriptor), the address from/to copy, the size of the data (control register) and a status register.

| APB address offset | Register |
|---|---|
| 0x00 | Control register |
| 0x04 | Status register |
| 0x08 | Interrupt mask register |
| 0x0C | Error register |
| 0x10 | Channel Vector Pointer |
| 0x14 | Timer Reset Value register |
| 0x18 | Capability register |
| 0x1C | Interrupt flag register |
| 0x20 | Reserved |
| 0x24 | M2B Descriptor Address register* |
| 0x28 | M2B Descriptor Control register* |
| 0x2C | M2B Descriptor Status register* |
| 0x30 | Reserved |
| 0x34 | B2M Descriptor Address register* |
| 0x38 | B2M Descriptor Control register* |
| 0x3C | B2M Descriptor Status register* |
| 0x40 | Internal Buffer Pointers Register |
| 0x800-0xFFF | Internal Buffer Readout Area |

*Only used in Simplified Mode of Operation

*Figure 5: GRDMAC controller registers*

*from [1] GRIP, Dec 2017, Version 2017.3, Cobham Gaisler AB*

In normal mode of operation only, data descriptors can be preceded in the list by conditional descriptors. These enable to poll a status register or monitor a triggering line. In this mode the M2B and B2M descriptors in the controller registers are not used. Instead, the channel vector pointer (APB address offset 0x10) links to a list of maximum 16 couples (channels) of descriptor lists. Once the GRDMAC has completed a channel it starts the next one or loops back to the first one if there is not any. [1]

# 4. GRDMAC Configuration Coding

Coding for the GRDMAC requires to create the adequate structures for the descriptors and controller register, and to populate them with the adequate value. Once the the GRDMAC is configured, it it triggered using the bit 0 of the control register (enable). Fortunately the test code for GRDMAC provides a good red thread. It is found in grlib-gpl-2018.1-b4217/software/leon3/grdmac.c [2]. The different tests provide insights on different way to use the GRDMAC. The test grdmac_test configures a GRDMAC in simplified mode. The test grdmac_apbuart_test configures two DMA controllers with polling condition using channel 0. The test grdmac_i2c_test configures a DMA controller using 8 channels.

# III. FPGA Implementation

The whole FPGA implementation has been performed using GRLIB environment. It provides a GUI for configuration (even though it is limited) and makefiles for compilation, simulation, synthesis and analysis. Learning this new environment took a long time.

The GRLIB 2017.1 library contains quite a lot of errors or undone parts. Those had to be debugged and written. I stumbled across them again in GRLIB 2018.3. They are summed up in annex II, and they have been reported to info@gaisler.com which is the GRLIB developer and provider.

## 1. System Setup

First it has been attempted to naively add the DMA to the working base. It had been soon reconsidered. The working base was primarily using GRLIB 2012.1 (from January 2012). The GRDMAC has been added to GRLIB in 2016. The selected GRDMAC is from GRLIB 2017.3 (from November 2017). Using solely GRLIB 2017.3 or a later version have large benefits:

- Some bugs have been fixed in this five years time.
- The GRLIB environment scripts and some parts are highly dependent on Vivado version, which in particular provides the unisim cells library for all Xilinx FPGA.
- A few communication signals have been removed and added, slightly modifying all blocks and buses.
- It avoids creating a chimaera library and it will be easier to transfer.

The new library also contains more components and can provide LEON4 processor and AMBA3.0.

The simulated system for Virtex7 FPGA VC707, using GRLIB 2017.3 is depicted on figure 6. Everything but the emulated external PROM and SRAM have been impacted one way or another by the library change when compared with the working base (figure1). These are plenty minor changes. An I2C controller is present by default in the new design and it has been decided to keep it. The useless custom counter has been removed.

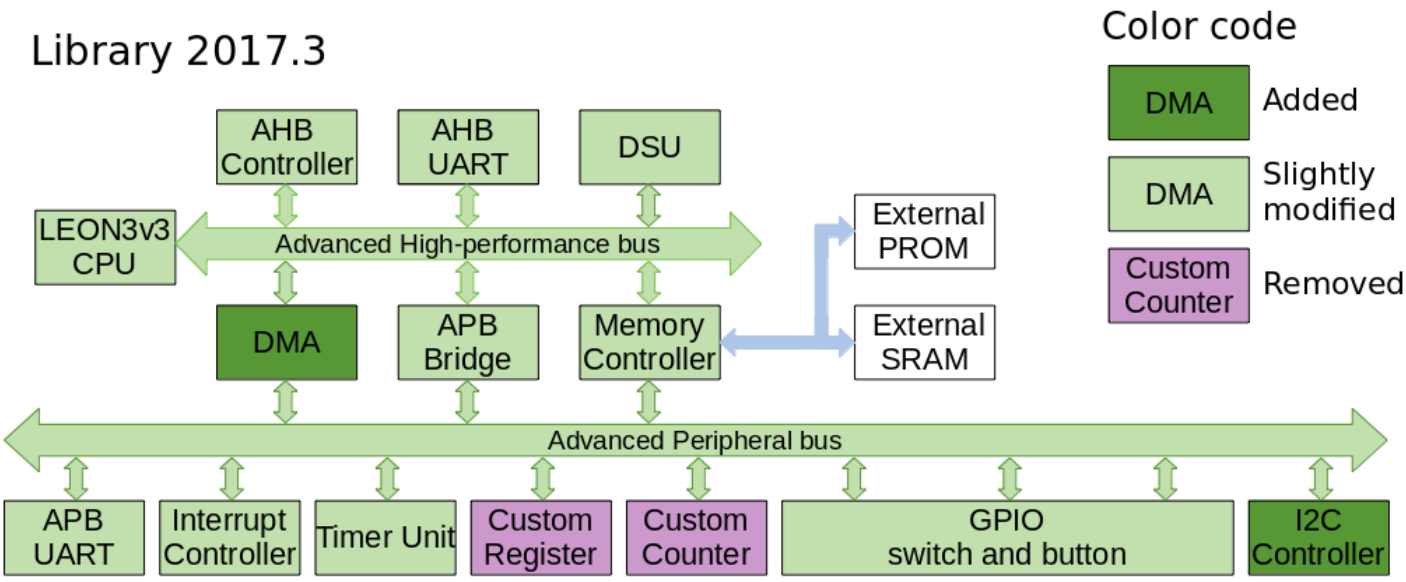


*Figure 6: The simulated system compared with the working base*

On the very first system which has been simulated using GRLIB 2017.3, the custom register had been modified to add a flag and has been kept as an input/output component for DMA custom test purposes. The tests were conclusive and it has been removed for the next implementations, as depicted on figure 6.

The processor used is now a LEON3v3, which is the only LEON3-type processor implemented in GRLIB 2017.1. This is a very minor change. The differences are listed below for reference:

- The data cache and AMBA behavior for cache have changed (the signal 'hcache' has been replaced by a test signal for synchronous RAM), which has a side effect on and the precise case of reading from ASI 0x1.
- Ancillary state registers ASR17 and ASR19-31 were made privileged on write.
- Double-mapped virtual addresses are possible with self-snooping.

The partition of the APB bridge between the APB slaves is handled with addresses and masks. Each component is manually configured an address and a mask. When a master requests an address, for each slave the bits of the requested address are filtered by the mask and compared with the slave address. The slave is then selected accordingly using the APB select signal. This mask system enables to to configure different sizes. An FFF mask will provide the smallest area: 256 bytes. A mask with a lower value will provide a larger area. The masters/slaves index and the partition of the APB bridge between the components is reported on figure 7.

The GRDMAC has a debug function which provides access to its buffer: When the GRDMAC APB slave is selected, if the 11$^{th}$ bit of the address is ‘1’ it will report its internal buffer. Otherwise it will report its input/output. Here the input/output address used is 0x80001000. The mask which has been used for GRDMAC is FF7, enabling the debug function on address 0x80001800 for the 256 first bytes of the buffer. This setup has been decided after a 256 bytes buffer size was chosen for the GRDMAC. This feature is not highlighted on figure 7. When designing a circuit with the GRDMAC one should either restrict the GRDMAC to its I/O address using a FFF mask or be cautious not to use the buffer access for other components.

```
LEON3 Xilinx VC707 Demonstration design
# GRLIB Version 2017.3.0, build 4208
# Target technology: inferred  , memory library: inferred
# ahbctrl: AHB arbiter/multiplexer rev 1
# ahbctrl: Common I/O area at 0xfff00000, 1 Mbyte
# ahbctrl: AHB masters: 16, AHB slaves: 16
# ahbctrl: Configuration area at 0xfffff000, 4 kbyte
# ahbctrl: mst0: Cobham Gaisler          LEON3 SPARC V8 Processor
# ahbctrl: mst1: Cobham Gaisler          AHB Debug UART
# ahbctrl: mst2: Cobham Gaisler          GRDMAC DMA Controller
# ahbctrl: slv0: European Space Agency   LEON2 Memory Controller
# ahbctrl:       memory at 0x00000000, size 512 Mbyte, cacheable, prefetch
# ahbctrl: slv1: Cobham Gaisler          AHB/APB Bridge
# ahbctrl:       memory at 0x80000000, size 256 Mbyte
# ahbctrl: slv2: Cobham Gaisler          LEON3 Debug Support Unit
# ahbctrl:       memory at 0xd0000000, size 256 Mbyte
# ahbctrl: slv3: Cobham Gaisler          Test report module
# ahbctrl:       memory at 0x20000000, size 1 Mbyte
# ahbctrl: slv4: Cobham Gaisler          Xilinx MIG DDR3 Controller
# ahbctrl:       memory at 0x40000000, size 256 Mbyte, cacheable, prefetch
# apbctrl: APB Bridge at 0x80000000 rev 1
# apbctrl: slv0: European Space Agency   LEON2 Memory Controller
# apbctrl:       I/O ports at 0x80000000, size 256 byte
# apbctrl: slv1: Cobham Gaisler          Generic UART
# apbctrl:       I/O ports at 0x80000100, size 256 byte
# apbctrl: slv2: Cobham Gaisler          Multi-processor Interrupt Ctrl.
# apbctrl:       I/O ports at 0x80000200, size 256 byte
# apbctrl: slv3: Cobham Gaisler          Modular Timer Unit
# apbctrl:       I/O ports at 0x80000300, size 256 byte
# apbctrl: slv4: Cobham Gaisler          Xilinx MIG DDR3 Controller
# apbctrl:       I/O ports at 0x80000400, size 256 byte
# apbctrl: slv5: Cobham Gaisler          GRDMAC DMA Controller
# apbctrl:       I/O ports at 0x80001000, size 256 byte
# apbctrl: slv7: Cobham Gaisler          AHB Debug UART
# apbctrl:       I/O ports at 0x80000700, size 256 byte
# apbctrl: slv8: Cobham Gaisler          LEON3 Statistics Unit
# apbctrl:       I/O ports at 0x80010000, size 1 kbyte
# apbctrl: slv9: Cobham Gaisler          AMBA Wrapper for OC I2C-master
# apbctrl:       I/O ports at 0x80000900, size 256 byte
# apbctrl: slv10: Cobham Gaisler          General Purpose I/O port
# apbctrl:       I/O ports at 0x80000a00, size 256 byte
# testmod3: Test report module
# apbuart1: Generic UART rev 1, fifo 4, irq 2, scaler bits 12
# grgpio10: 7-bit GPIO Unit rev 3
# gptimer3: Timer Unit rev 1, 8-bit scaler, 7 32-bit timers, irq 8
# irqmp: Multi-processor Interrupt Controller rev 4, #cpu 1, eirq 0
# i2cmst9: AMBA Wrapper for OC I2C-master rev 3, irq 10
# ahbuart7: AHB Debug UART rev 0
# lstat_8: LEON Statistics Unit, ncpu : 1, ncnt : 4, rev 1
# dsu3_2: LEON3 Debug support unit
# leon3_0: LEON3 SPARC V8 processor rev 3: iuft: 0, fpft: 0, cacheft: 0
# leon3_0: icache 1*4 kbyte, dcache 1*4 kbyte
```

*Figure 7: Memory repartition and masters/slaves index for FPGA*

## 2. Simulations

The test program is written in systest.c which is encoded into the Motorola S-record file ram.srec. First the initialisation file rom.srec is read, then the ram.

The base_test tests the memory, leon3 processor, interrupt controller, the general purpose timer unit and the APBUART. The grdmac_test performs the copy of 32 bits from RAM to RAM in simple mode. It generates

interrupt on completion. This test requires the initialization of the interrupt base which is performed during the interrupt controller test. The system was simulated using Mentor Modelsim and passed the tests.

The grdmac_apbuart_test requires two DMA controllers to perform a transmission and reception on a APBUART interface:

- The first DMA reads 4*4 contiguous bytes from RAM. Using a conditional descriptor it polls the UART status every 256 cycles and transfers the bytes one at a time to a fixed address in APB memory space. It uses channel 0.
- Simultaneously the second DMA polls the UART “Receiver Data ready” flag every 256 cycles using a conditional descriptor. It reads one byte at a time from a fixed address in APB memory space. It writes the 16 bytes to RAM and generates an interrupt on completion. It uses channel 0.

This test requires the initialization of the interrupt base which is performed during the interrupt controller test. In order to test polling conditions and descriptor lists, a second DMA has been momentarily added to the system. The system passed the tests.

# 3. Synthesis

Synthesis has been performed using Xilinx Vivado. It stopped with an error message: The memory interface generator (MIG) configured in GRLIB 2017.3 didn't match the version provided by Vivado 2017.2, and this is a proprietary IP. GRLIB 2017.3 is indeed adapted for Vivado 2017.1 and no GRLIB is adapted to Vivado 2017.2.

During the project, in May, library GRLIB 2018.1 was released. It is adapted to Vivado 2017.3. It might be back compatible for the memory generator, but also cause new trouble. The MIG error has put an end to the FPGA implementation project.

# IV. ASIC implementation

The newest library 2018.1 has been used for the ASIC implementation. The technology used was TSMC 65nm. The technology features can be found on annex 1.

## 1. System Setup

The system has been skimmed for ASIC. The unused input and outputs ports have been stripped. The I2C component have been removed and the FPGA input/output have been replaced by a 8 bits general purpose input/output component. As a memory generator fitting the manufacturer wasn't provided, the internal PROM and SRAM have been withdrawn and the memory controller and external memories have been kept.

A work done in parallel has shown that the encryption component was responsible for 540 mW dynamic power usage. Hence the encryption component which appeared on the goal (figure 3) has been removed from the design. The same work also highlighted the large impact of the cache size on the circuit. Following this report, the cache size has been reduced from 4 kbyte to 1kbyte.

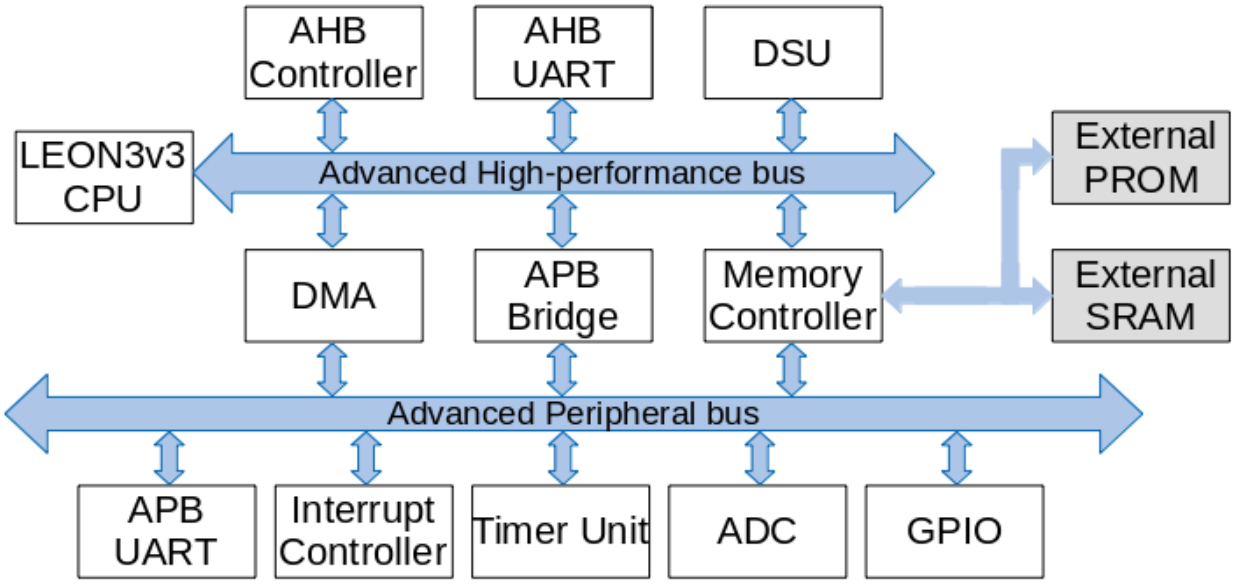


*Figure 8: The ASIC system*

The ADC designed in the lab has been added to the system. It is a 10 bit synchronous successive approximation ADC with a very wide range: 80 kHz to 80 MHz. It will be preceded by a dynamic amplifier.

# 2. Simulation

In GRLIB 2018.1 there is a new and recommended setting for the memory management unit which implements a supervisor bit to allows to check access permissions (CFG_MMUEN=2). The corresponding test has not been updated yet and returns an error. The system was configured using the previous implementation with CFG_MMUEN=1.

The simulation has been performed using Mentor Modelsim and the system passed the tests base_test and grdmac_test.

# 3. Synthesis

The synthesis has been performed using Synopsys Design Vision. A moderate optimization effort have been used for map and area, whilst a higher one has been used for power optimization. The design has not been flatten to facilitate the troubleshooting, but a mere boundary optimization has been used. Only the clock network, registers and combinational units are counted as internal power sources.

Design Vision optimizes the power usage only when power constraints are set. The goal of this device being low power consumption, a strict power constraint and a large clock period constraint have been used.

First, a 10ns clock frequency have been set with weak area and power constraints (1W maximum total power). It was easily achieved and led to a 654600um² cell area and 159 total mW total power.

Then, to force the power optimization over area optimization the design was analysed again using a 100mW total power constraint. It lead to the same exact design, already optimized.

A third analysis with strong power and area constraints provided a slightly better mapped design, which has been kept for place and route. The total area is 636206um² and the total power consumption 159mW, as shown on figure 9. The ADC controller has been reduced from 632um² to 65 um² and the GRDMAC from 10608 to 10498um² (1.7% of total). The ADC controller internal power and leakage power are 3.12uW and 12.3nW respectively. The GRDMAC internal power and leakage power are 463uW and 2617nW.

```
Global Operating Voltage = 1.08
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000pf
    Time Units = 1ns
    Dynamic Power Units = 1mW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =  42.0611 mW   (26%)
  Net Switching Power  = 116.8356 mW   (74%)
                         ---------
Total Dynamic Power    = 158.8967 mW  (100%)

Cell Leakage Power     = 132.0593 uW


                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000           0.0000            0.0000             0.0000  (   0.00%)
memory             0.0000           0.0000            0.0000             0.0000  (   0.00%)
black_box          0.0000           0.0000            0.0000             0.0000  (   0.00%)
clock_network      0.2300         116.7767        7.2517e+03           117.0133  (  73.57%)
register          41.8445        3.7780e-03        1.0444e+05            41.9534  (  26.38%)
sequential         0.0000           0.0000            3.9426         3.9426e-06  (   0.00%)
combinational  1.6443e-02        5.4796e-02        2.0367e+04         9.1604e-02  (   0.06%)
--------------------------------------------------------------------------------------------------
Total             42.0910 mW       116.8353 mW     1.3206e+05 nW       159.0583 mW
```

*Figure 9: summary power report. constraints: max area 0, max total power 50mW*

Out of curiosity an analysis has been carried out with a high clock frequency and weak power constraint. The critical timing path was found in the LEON3 memory cache. Taking into account the arrival time and data hold time, the clock period has to be at least larger than 2.62ns for this design. This analysis led to a mere 7.6% area increase, and, as expected, a large 795.3mW total power consumption.

The design contains two high-fanout nets : the clock (219476 loads) and the bit of the integer unit debug input [TIMER][30] (10215 loads).

# 4. Place and route

The place and route steps have been performed using Cadence Innovus. The software have been released in 2015 and the transition in the lab from Cadence SoC Encounter happened at the same time as this work.

Place and route encountered some clock and pads errors. In order to solve them, the pads have been stripped and instantiated with the correct technology in a wrapper, input/output ports have been de-multiplexed and the clock has been stripped from unused elements. The testbench have been modified accordingly. The new top level raises some errors in synthesis. The implementation could be pushed further due to time constraint.

# Conclusion

The integration into the smart sensor node of a DMA has been successfully inquired. The DMA controller GRDMAC provided by Cobham Gaisler AB suits the needs of the design. It is part of the same library as the processor, and hence the system can be set up and implemented using the library tools. However the use of the new DMA required to update the library GRLIB from the 2012 version. GRLIB 2017.3 has been used for the FPGA and GRLIB 2018.1, which has been released during the project, for the ASIC.

The implementation of the circuit for Virtex7 FPGA VC707 has been stopped by a version issue with a proprietary memory interface generator from Xilinx. The ADC developed in the LSM have been added to the implementation for ASIC. It has been shown that the DMA and the ADC use power and area sparingly. The ASIC implementation have not been conducted through the end.

The use of the open source GRLIB library revealed some errors and missing parts, which have been reported to Cobham Gaisler AB.

This master thesis enabled me to experience work in an academic laboratory – what is more to use the EPFL ressources. It has been my first real experience in implementing a design for FPGA and ASIC. Fixing errors and adding manually the Unisim library (which is protected by a Xilinx license) made me familiar with GRLIB, and in particular its tools for configuration, scripting and compiling.

# Cost Analysis

This project made use of Mentor ModelSim, Synopsys Design Vision, Cadence Innovus and Vivado HL System Edition. A Xilinx virtex-7 FPGA evaluation kit has also been used, and the system will be manufactured on an ASIC TSMC 65nm.

The LSM is an academic laboratory. It benefits from academic licenses and maintenance from Europractice for the EDA tools. It also receives promotional FPGA, and CERN provides them with ASIC fabrication. The overall cost is hence drastically reduced when compared to a private company. For instance Vivado Design Suite System Edition license is free of charge for academic institutes and costs 4295$/license otherwise. Furthermore its maintenance by Europractice is only 200€/year. Cadence IC package, which contains Innovus, costs 1800€ for each of the five firsts licenses and 360€ for additional ones. All disclosable prices can be found on Europractice website http://www.europractice.stfc.ac.uk.

The office costs are provided by EPFL.

# References

[1] Cobham Gaisler AB, GRIP, Dec 2017, Version 2017.3 “GRLIB IP Core User’s Manual”

[2] Cobham Gaisler AB, grlib-gpl-2018.1-b4217, May 2018.

[3] TSMC 65nm technology overview
On the internet: http://www.europractice-ic.com/technologies_TSMC.php?tech_id=65nm

# Annex

## TSMC 65 NM TECHNOLOGY OVERVIEW (MPW):

| TECHNOLOGY | MS/RF |
|---|---|
| **Geometry** | 65 nm |
| **Device Application** | Low Power |
| **Core Voltage (V)** | 1.2V |
| **I/O Voltage (V)** | 2.5V |
| **Poly Layers** | 1 |
| **Metal Layers (Min)** | 4 |
| **Metal Layers (Max)** | 9 |
| **RO speed Std-Vt (ps/gate) Wn/Wp=5/3.6, L=0.06** | 9.33089 |
| **RO speed High-Vt (ps/gate) Wn/Wp=5/3.6, L=0.006** | 14.5022 |
| **RO speed Low-Vt (ps/gate) Wn/Wp=5/3.6, L=0.006** | 7.34581 |
| **BEOL Dielectric** | Low-K |
| **BEOL Metal** | Cu |

*Annex I: TSMC 65nm technology overview*

*[3] http://www.europractice-ic.com/technologies_TSMC.php?tech_id=65nm*

grlib-gpl-2017.3-b4208 and grlib-gpl-2018.1-b4217 have been modified in order to solve errors:

Modified line 281 of leon3-xilinx-vc707\testbench.vhd (can_rxd => "0", to can_rxd => (others => '0'),) to remove a compilation error when not using can.

Commented line 506 of leon3-xilinx-vc707\leon3mp.vhd (nojtag : ...) to remove a simulation error.

Commented line 52 in software\leon3\grdmac.c "enable_irq(irq);". enable_irq() is not a defined function. From a GRDMAC point of view interrupt is enabled in control, interrupt_mask, m2b_desc.control and b2m_desc.control.

Modified (volatile unsigned int **)&cvp0 and (volatile unsigned int **)&cvp1 to (volatile unsigned int **)&cvp0[0] and (volatile unsigned int **)&cvp1[0] in software\leon3\grdmac.c.

Replaced a TODO line 874 of leon3-xilinx-vc707\leon3mp.vhd with a VHDL code when not using ethernet. Here is the explanation of my changes: emdc, erst, txp and txn are ethernet outputs which are not driven. They are defined even when ethernet is not used. I decided to set them to '0' and gnd(0), to respect how the outputs are handled in the vhd file. I randomly chose gnd instead of vdd. Rem: The same would apply to can_rxd and can_txd weren't they defined by std_logic_vector(0 to CFG_CAN_NUM-1). To me the best solution would be to define all these unused outputs signals from 0 to CFG_COMPONENT_NAME. It would't be as robust but would not implement useless output ports.

In leon3-xilinx-vc707\leon3mp.vhd, there is no implemented mean not to use the I2C, even though it would be easy to do. I didn't need to fix it.

Removed line 156 in software\leon3\grdmac.c: int i; i is already defined. Removed j from line 249 in software\leon3\grdmac.c: j is not used.

During synthesis for FPGA VC707 , two redundant clocks were defined:
in \designs\leon3-xilinx-vc707\vivado\leon3-xilinx-vc707\leon3-xilinx-vc707.srcs\sources_1\ip\mig\mig\user_design\constraints\mig.xdc
and in \boards\xilinx-vc707-xc7vx485t\xilinx-vc707-xc7vx485t.xdc.
On the one hand the clock is used further in xilinx-vc707-xc7vx485t.xdc. On the other hand Vivado regenerate the file mig.xdc everytime.

In GRLIB 2018.1 there is a new and recommended setting CFG_MMUEN=2. The corresponding test has not been updated yet and returns an error. The system was configured using the previous implementation with CFG_MMUEN=1.

*Annex II: List of errors and corrections errors in GRLIB*