NGMP Specification

Next Generation Multi-Purpose Microprocessor

Contract: 22279/09/NL/JK

TABLE OF CONTENTS

1 INTRODUCTION....6
1.1 Scope of the Document....6
1.2 Functional Requirements....6
1.3 Applicable Documents....6
1.4 Reference Documents....7
1.5 Acronyms....8

2 ARCHITECTURE....10
2.1 Architectural Overview....10
2.2 LEON4FT SPARC V8 Processor Cores....13
2.2.1 Description....13
2.2.1.1 Overview....13
2.2.1.2 Integer Unit....13
2.2.1.3 Floating Point Unit....13
2.2.1.4 Timer Unit....14
2.2.1.5 Interrupt controller....14
2.2.1.6 Memory Management Unit....14
2.2.1.7 L1 Cache....15
2.2.2 Performance counters....15
2.2.3 Block moves....15
2.2.4 Configuration....15
2.3 L2 Cache....16
2.3.1 Description....16
2.3.2 Configuration....17
2.4 DDR2 Memory Interface....17
2.4.1 Description....17
2.4.2 Reed-Solomon Codecs....17
2.4.3 Configuration....18
2.5 SDRAM Controller....18
2.5.1 Description....18
2.5.2 Configuration....19
2.6 On-chip SDRAM....19
2.6.1 Description....19
2.6.2 Configuration....19
2.7 Memory Scrubber....19
2.7.1 Description....19
2.7.2 Modes of Operation....20
2.7.3 Scrubber Bandwidth....20
2.7.4 Memory Initialisation....20
2.7.5 Memory Scrubbing and Re-generation....21
2.7.6 Configuration....22
2.8 AHB/AHB Bridge Connecting Debug AHB Bus....22
2.8.1 Description....22
2.8.2 Configuration....22
2.9 DSU4 – LEON4 Hardware Debug Support Unit....22

2.9.1 Description....22
2.9.1.1 Overview....22
2.9.1.2 Operation....23
2.9.1.3 AHB Trace Buffer....23
2.9.1.4 Instruction Trace Buffer....24
2.9.1.5 Data Watchpoints....24
2.9.1.6 Data Area Monitoring....24
2.9.1.7 AHB Performance Monitoring....24
2.9.2 Configuration....25
2.10 JTAG Debug Link Controller....25
2.10.1 Description....25
2.10.2 Configuration....25
2.11 USB Debug Link Controller....25
2.11.1 Description....25
2.11.2 Configuration....26
2.12 SpaceWire RMAP Target Debug Link Controller....26
2.12.1 Description....26
2.12.2 Configuration....26
2.13 Uni-directional AHB Bridge with Protection Functionality....26
2.13.1 Description....26
2.13.2 Configuration....27
2.14 Ethernet Controllers....27
2.14.1 Description....27
2.14.2 Configuration....28
2.15 SpaceWire Controllers....29
2.15.1 Description....29
2.15.2 Configuration....29
2.16 High Speed Serial Link Controller....29
2.16.1 Description....29
2.16.2 Configuration....30
2.17 PCI Controller....30
2.17.1 Description....30
2.17.2 Configuration....30
2.18 Uni-directional AHB Bridge Connecting Peripheral Slave Interfaces....30
2.18.1 Description....30
2.18.2 Configuration....31
2.19 PROM/IO Controller....31
2.19.1 Description....31
2.19.2 Configuration....31
2.20 APB Bridge....31
2.20.1 Description....31
2.20.2 Configuration....31
2.21 32-bit Programmable Timers....32
2.21.1 Description....32
2.21.2 Configuration....32
2.22 Multiprocessor Interrupt Controller....32
2.22.1 Description....32

2.22.2 Configuration....33
2.23 Secondary Interrupt Controllers....33
2.23.1 Description....33
2.23.2 Configuration....33
2.24 General Purpose I/O Port Controller....33
2.24.1 Description....33
2.24.2 Configuration....33
2.25 8-bit UART....33
2.25.1 Description....33
2.25.2 Configuration....34
2.26 General Purpose Register for Clock Gate Control....34
2.26.1 Description....34
2.26.2 Configuration....34
2.27 PCI Arbiter....34
2.27.1 Description....34
2.27.2 Configuration....34
2.28 AHB Status Registers....34
2.28.1 Description....34
2.28.2 Configuration....35
2.29 IRQSTAMP....35
2.29.1 Description....35
2.29.2 Configuration....36
2.30 AHBCTRL....36
2.30.1 Description....36
2.30.2 Configuration....36
2.31 Memory Map....36
2.32 Interrupt Assignments and Infrastructure....38
2.32.1 Overview....38
2.32.2 SMP Configuration....39
2.32.3 ASMP Configuration....39
2.32.4 Mixed Configurations....40
2.32.5 Interrupt Assignments....40
2.33 Error Handling....43
2.33.1 Overview....43
2.33.2 Access Violations and Errors Reported via AMBA....43
2.33.3 Errors in External Memory....44
2.33.4 Errors in LEON4FT Register File, Caches and MMU....44
2.33.5 Errors in Internal Buffers of IP cores....44

3 NGMP ASIC IMPLEMENTATION....45
3.1 Overview....45
3.2 Operating Frequency and Clock Domains....45
3.3 Power-up and Initialization State....46
3.4 Reset and Power Cycling....47
3.5 Electrical Constraints....47
3.6 Thermal Environment Constraints....47
3.7 Radiation Environment Constraints....47

3.8 Power Budget and Dissipation....47
3.9 Physical and Mechanical Constraints....48
3.10 Timing....48

4 FAULT-TOLERANCE....49

5 DESIGN VERIFICATION....50
5.1 Test Modes....50
5.1.1 BIST....50
5.1.2 Memory Test Functionality in Memory Scrubber....50
5.1.3 On-line Device tests....50
5.2 Fault Coverage Required at Production Test....51

6 REUSABILITY....52
6.1 Overview....52
6.2 Portability....52

7 OPEN ITEMS....53

8 REQUIREMENTS MATRICES....55
8.1 Statement of Work Key Requirements....55
8.2 Compliance with ECSS-Q-60-02A....56

1 INTRODUCTION

1.1 Scope of the Document

This document defines the architecture of the Next Generation Multi-Purpose Microprocessor (NGMP). The NGMP has been defined as part of an activity initiated by the European Space Agency under ESTEC contract 22279/09/NL/JK.

The work has been performed by Aeroflex Gaisler AB, Göteborg, Sweden. Drafts of the specification have been reviewed by EADS Astrium.

1.2 Functional Requirements

The NGMP has the following requirements:

- SPARC V8(E) based multi-core architecture
- An average performance of 400 MOPS over the benchmarks executed during the benchmarking of the GINA platform, with a minimum of 200 MOPS on any single benchmark
- Improved debug support with respect to LEON2FT
- Large on-chip memory banks (>= 32 MiB, subject to constraints imposed by target technology), extended caches
- Standardised interfaces such as HSSL, SpaceWire, Ethernet
- Efficient interface for scalable multi-processor architectures, co-processors and/or companion devices for I/Os and memory interfaces
- Power consumption of the NGMP ASIC core (without I/Os) under worst case operating conditions and maximum software load shall not exceed 6W. Maximum power consumptions in idle mode (no software activity, but conservation of status and SEE protection) shall not exceed 100 mW

1.3 Applicable Documents

[AD1] “Statement of Work, Next Generation Multi Purpose Microprocessor”, ESTEC, TEC-EDM/2007.23/RW Issue 2, Revision 4, June 2008

[AD2] “Minutes of Meeting”, Aeroflex Gaisler, MIN-2009-0602, May 2009

[AD3] “ECSS Q60-02A: ASIC and FPGA Development”, http://www.microelectronics.esa.int/asic/ECSS-Q-60-02A-17July2007.pdf

[AD4] “Architecture Exploration Report, Next Generation Multi-Purpose Microprocessor”, NGMP-EXPL-0003-i1r2, Aeroflex Gaisler, 2009-12-15

[AD5] “SEE Mitigation Plan, Next Generation Multi-Purpose Microprocessor”, Aeroflex Gaisler, NGMP-SEEM-0005-i1r0, 2009-12-15

[AD6] “Development Plan, Next Generation Multi-Purpose Microprocessor”, Aeroflex Gaisler, NGMP-DEVEL-0004-i1r0, 2009-12-15

[AD7] “Feasibility Report Spreadsheet, Next Generation Multi-Purpose Microprocessor”, Aeroflex Gaisler, NGMP-FSPR-0009, 2009-12-15

1.4 Reference Documents

[RD1] "GRLIB IP Core User's Manual", Aeroflex Gaisler, Version 1.0.21, August 2009

[RD2] "DDR2 SDRAM Specification", JEDEC Solid State Technology Association, JESD79-2E, April 2008

[RD3] "AMBA Specification", ARM Limited, Rev 2.0, May 1999

[RD4] "EIA Standard RS-232-C Interface Between Data Terminal Equipment and Data Communication Equipment Employing Serial Data Interchange", Electronics Industries Association, August 1969

[RD5] "An Introduction to the Intl QuickPath Interconnect", Intel Corporation, Document number 320412-001US, January 2009

[RD6] "HyperTransport I/O Technology Overview - An Optimized, Low-latency Board-level Architecture", HyperTransport Consortium, HTC_WP02, Rev 001, June 2004

[RD7] "Space engineering SpaceWire - Links, nodes, routers and Networks", European Cooperation for Space Standardization, ECSS-E-ST-50-12C, July 2008

[RD8] "Space Engineering SpaceWire Protocols", European Cooperation for Space Standardization, ECSS-E-50-ST-11C, January 2008

[RD9] "Universal Serial Bus Specification", Revision 2.0, April 2000

[RD10] "UTMI+ Low Pin Interface (ULPI) Specification", Revision 1.1, October 20 2004

[RD11] “Rad-Hard 32 bit SPARC V8 Processor”, AT697E, ATMEL

[RD12] OpenSPARC T1 documentation, http://www.opensparc.net

[RD13] “Simply RISC, an embedded version of the OpenSPARC T1”

[RD14] “-Overview- AMBA AVALON CORECONNECT WISHBONE”. V1.1, Patrick Pelgrims, Dries Driessens and Tom Tierens

[RD15] “GINA study”, Call-Off-Order 3 of ESA Contract 18533/04/NL/JD, Summary Report and Presentation

[RD16] “Overview Multi-layer AHB”, ARM DVI 0045B

[RD17] AMBA AXI Protocol Specification, v.1.0, ARM IHI 0022B

[RD18] “Deep Sub-Micron ASIC Technology Assessment and High-Speed-Serial-Links Design”, SoW for ESA ITT AO/2-1568/07/NL/JD and “High Perf. ASIC Platform Approach for Next Satellite Telecom Processors – Possible Application to Other Processors”

[RD19] “The LEON2-FT Processor User's Manual”, Gaisler Research, v. 1.0.9.16.1, February 2007

[RD20] “Recommended HW features for future VM evolution”, ASSERT technical note, Dec 2007

[RD21] “Improving ARM Code Density and Performance, New Thumb Extensions to the ARM Architecture”, Richard Phelan, June 2003

[RD22] "GRLIB IP Library User's Manual", Aeroflex Gaisler, Version 1.0.21, August 2009

1.5 Acronyms

AHB	Advanced High-performance Bus, part of AMBA 2.0 Specification
AMBA	Advanced Microcontroller Bus Architecture, bus architecture widely used for on-chip buses in SoC designs.
APB	Advanced Peripheral Bus, part of AMBA 2.0 Specification
ASIC	Application Specific Integrated Circuit
ASMP	Asymmetric Multi-Processing (in the context of this document: each of the CPU cores runs its own OS)
BCH	Bose-Hocquenghem-Chaudhuri, class of error-correcting codes
BIST	Built In Self Test
BSP	Board Support Package
CPU	Central Processing Unit
DCL	Debug Communication Link
DDR	Double Data Rate
DMA	Direct Memory Access
DSM	Deep-Sub-Micron ASIC technology
DSU	Debug Support Unit
EDAC	Error Detection And Correction
EDCL	Ethernet Debug Communication Link
FIFO	First-In-First-Out, refers to buffer type
FLOPS	Floating Point Operations Per Second
FPU	Floating Point Unit
Gb	Gigabit, 10^9 bits
GB	Gigabyte, 10^9 bytes
Gib	Gibibit, gigabinary bit, 2^{30} bits
GiB	Gibibyte, gigabinary byte, 2^{30} bytes, unit defined in IEEE 1541-2002
GINA	Giga INstruction Architecture
GRLIB	Aeroflex Gaisler's IP core Library
HSSL	High-Speed Serial Link
HDL	Hardware Description Language
I/O	Input/Output
IP	Intellectual Property
IP, IPv4	Internet Protocol
IPR	Intellectual Property Rights
ISR	Interrupt Service Routine
ITT	Invitation To Tender
JTAG	Joint Test Action Group (developer of IEEE Standard 1149.1-1990)
kB	Kilobyte, 10^3 bytes
KiB	Kibibyte, 2^{10} bytes, unit defined in IEEE 1541-2002
LRU	Least-Recently-Used
MAC	Media Access Controller, when referring to, for instance, an Ethernet MAC
Mb, Mbit	Megabit, 10^6 bits
MB	Megabyte, 10^6 bytes
MiB	Mebibyte, 2^{20} bytes, unit defined in IEEE 1541-2002
MILS	Multiple Independent Levels of Security
MIPS	Million of Instructions Per Second
NGMP	Next Generation Multi-Purpose microProcessor
OS	Operating System
PCI	Peripheral Component Interconnect
PROM	Programmable Read-Only Memory
RAM	Random Access Memory
RMAP	Remote Memory Access Protocol

RS232	Recommended Standard 232, standard for serial data signals
RTL	Register Transfer Level
SDRAM	Synchronous Dynamic Random Access Memory
SEE	Single Event Effects
SEL/SEU/SET	Single Event Latchup/Upset/Transient
SMP	Symmetric Multi-Processing
SPARC	Scalable Processor ARChitecture
SOC, SoC	System-On-a-Chip
SoW	Statement of Work
TCP	Transmission Control Protocol
TMR	Triple Modular Redundancy
UART	Universal Asynchronous Receiver/Transmitter
UDP	User Datagram Protocol
USB	Universal Serial Bus

2 ARCHITECTURE

2.1 Architectural Overview

The system will consists of five AHB buses; one 128-bit Processor bus, one 128-bit Memory bus, two 32-bit I/O buses and one 32-bit Debug bus. The Processor bus will include four LEON4FT cores connected to a shared L2 cache. The Memory bus is located between the L2 cache and the main external memory interfaces, DDR2 and SDRAM interfaces on shared pins, and will include a memory scrubber and possibly on-chip memory. The I/O bus has been split into two separate buses where all slave interfaces have been placed on one of the buses (Slave I/O bus) and all master interfaces have been placed on the other bus (Master I/O bus). The Master I/O bus connects to the Processor bus via an AHB bridge that provides access restriction functionality.

The two I/O buses include all peripheral units such as PCI, High-Speed Serial Link, Ethernet MAC, and SpaceWire interfaces. The dedicated 32-bit Debug bus connects one debug support unit (DSU), one JTAG debug link, one SpaceWire RMAP target, one USB debug communication link, and optionally two Ethernet debug communication links. The Debug bus allows for non-intrusive debugging through the DSU and direct access to the complete system, since the Debug bus is not placed behind an AHB bridge with access restriction functionality.

The target frequency of the LEON4FT and on-chip buses is 400 MHz, but depends ultimately on the implementation technology. The SDRAM interface will be able to run at the same or one quarter of the system frequency. The DDR2 interface will be run at the same or twice the system frequency. The clock scaling factor between the memory interfaces and the rest of the chip is selectable via an external signal.

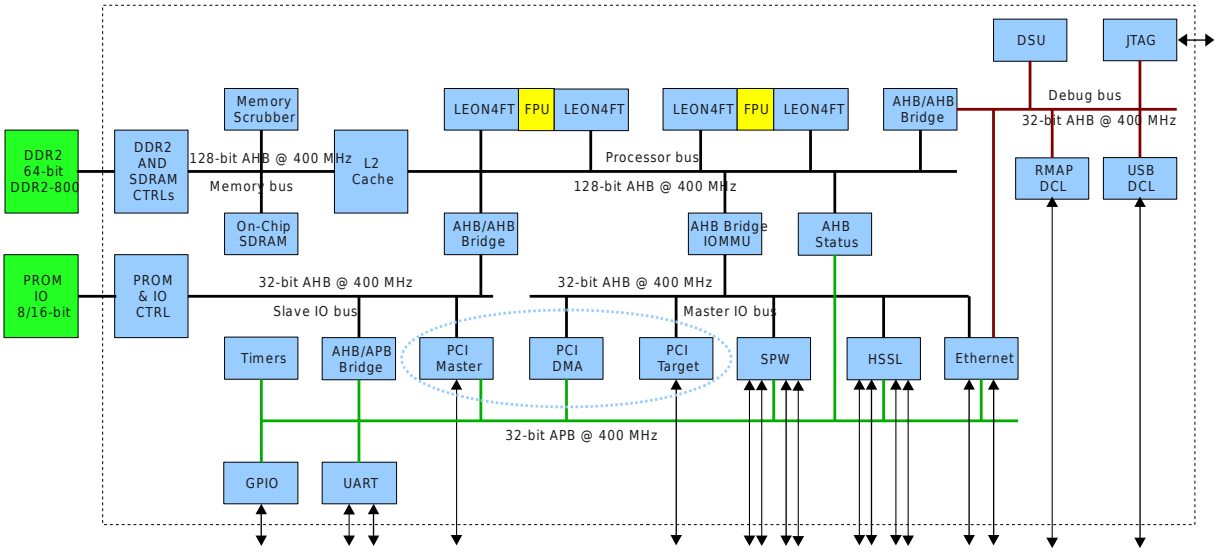

Illustration 1: NGMP Overview Block Diagram

An important factor to high processor performance and good SMP scaling is high memory bandwidth coupled with low latency. An 128-bit AHB bus will therefore be used to connect the LEON4FT processors. This will allow 32 bytes to be read in 2 clocks, not counting the

initial memory latency. To mask memory latency, the GRLIB L2 cache will be used as a high-speed buffer between the external memory and the AHB bus. A read hit to the L2 cache typically requires 3 clocks, while a write takes 1 clock. A 32-byte cache line fetch will be performed as a burst of two 128-bit reads. The first read will have a delay of 3 clocks and the second word will be delivered after one additional clock. A cache line will thus be fetched in 4 clocks (3 + 1). Error correction will add an additional latency of 1 clock to all read accesses to allow time for checksum calculation.

The DDR2 memory controller will be placed on a separate bus that is connected to the L2 cache. On this bus a memory scrubber and possibly on-chip SDRAM will be located. This will allow the memory scrubber to access the external memory without causing traffic on the Processor bus and the scrubber can also be used to scrub the on-chip SDRAM. Another advantage gained by this position of the on-chip SDRAM is that its data can be cached in the L2 cache, which will reduce the average access latency.

All I/O master units in the system contain dedicated DMA engines and are controlled by descriptors located in main memory that are set up by the processors. Reception of, for instance, Ethernet and SpaceWire packets will not increase CPU load. The cores will buffer incoming packets and write them to main memory without processor intervention.

The list below summarizes the specification for the NGMP system:

System architecture

- 128-bit Processor AHB bus
  - 4x LEON4FT with GRFPU shared between pairs of LEON4FT with 4-word instruction FIFO, or dedicated GRFPU for each processor core, per processor MMU and per processor L1 cache.
  - 1x Shared L2 cache
  - 1x 128-bit to 32-bit unidirectional AHB to AHB bridge (connecting Processor bus with Debug bus)
  - 1x 128-bit to 32-bit unidirectional AHB to AHB bridge (connecting Processor bus to slave IO bus)
  - 1x 32-bit to 128-bit unidirectional AHB to AHB bridge with IOMMU (connecting master IO bus to Processor bus)
- 128-bit Memory AHB bus
  - 1x 64-bit DDR2 memory interface with Reed-Solomon ECC
  - 1x 64-bit SDRAM memory interface with Reed-Solomon ECC
  - 1x Memory scrubber
  - 1x On-chip SDRAM (if available on the target technology)
- 32-bit Master I/O AHB bus
  - 4x SpaceWire cores with RMAP @ 200 Mbit/s
  - 4x High-Speed Serial Link, if available from foundry
  - 2x 10/100/1000 Mbit Ethernet interface with MII/GMII PHY interface
  - 1x 32-bit PCI target interface @ 66 MHz
- 32-bit Slave I/O AHB bus
  - 1x 32-bit PCI master interface @ 66 MHz
  - 1x 8/16-bit PROM/IO controller with BCH ECC
  - 1x 32-bit AHB to APB bridge, connecting 32-bit APB bus:
    - 1x General purpose timer unit
    - 1x 16-bit general purpose I/O port controller
    - 2x 8-bit UART interface
    - 1x Multiprocessor interrupt controller
    - 4x Secondary interrupt controller

        - 1x PCI arbiter with support for four agents
        - 2x AHB Status registers
        - 1x Interrupt time stamp unit
- 32-bit Debug AHB Bus
    - 1x Debug support unit
    - 1x USB debug link
    - 1x JTAG debug link
    - 1x SpaceWire RMAP target

Processor core

- LEON4FT with 16 + 16 KiB cache, SRMMU, physical snooping, 32-bit MUL/DIV
- GRFPU floating point, shared between two pairs of LEON4FT with 4-word instruction FIFO, or dedicated GRFPU for each processor core.
- Internal timer unit with five timers and watchdog functionality
- Internal interrupt controller

Memory interfaces

- DDR2SPA 64-bit DDR2 interface, 400 MHz (DDR2-800), with 16 or 32 RS ECC bits
- FTSDCTRL 64-bit SDRAM interface, 100 MHz (PC100), with 16 or 32 RS ECC bits
- FTSRCTRL8 based 8/16 bit PROM/IO controller with BCH ECC

Peripherals

- GRSPW2 SpaceWire core, 200 Mbit/s, RMAP target, redundant link drivers, DMA
- GRETH 10/100/1000 Mbit Ethernet MAC with EDCL and DMA
- GRPCI 32-bit data, 66 MHz, master/target/host functionality
- ESA PCI arbiter with support for four agents
- HSSL link, if available at time of implementation
- USBDCL USB debug link
- GRGPIO 16-bit General purpose I/O port controller
- APBUART 8-bit UART interface
- AHBJTAG JTAG debug link with AHB master interface
- SpaceWire RMAP target for dedicated SpaceWire debug link

System support cores

- AHBSTAT AHB status registers
- DSU4 Debug support unit with performance monitoring and AHB/INST trace buffers
- GPTIMER with five 32-bit timers, of which one is intended for use as a watchdog
- General purpose register for clock gating
- Multiprocessor interrupt controller
- Secondary interrupt controllers
- Memory scrubber
- Interrupt time stamp unit

The NGMP system can be utilised to build scalable multiprocessor architectures with loosely coupled operating systems. Building a tightly coupled architecture requires very fast communication links of a sort that is not available on the NGMP. Synchronisation between multiple NGMP systems should be done via message passing. Shared memory systems can be attained by mapping memory via PCI or by using special protocols over SpaceWire or Ethernet. With these communication interfaces, the shared memory will be a bottleneck, and data distribution may also become a limiting factor. Since the design lacks sufficient means to efficiently support hardware coherency between multiple chips, the NGMP is not

targeted at multiprocessing applications were several NGMP chips is viewed as a single entity. Multiprocessing applications involving several NGMP chips need to be handled as cluster computing.

The fault-tolerance in the NGMP system is aimed at detecting and correcting SEU errors in on-chip and off-chip RAM. The L1 cache and register files in the LEON4FT cores are protected using parity and BCH coding, while the L2 cache will use BCH. External DDR2 and SDR SDRAM memory will be protected using Reed-Solomon coding, while the boot PROM will use BCH. Any RAM blocks in on-chip IP cores will be protected with BCH or TMR. The protection capabilities of different cores is further discussed in section 2.33.

If the target technology has SEU hardened flip-flops, then they will be the preferred choice for all on-chip registers. If no hardened flip-flops are available, TMR will be used through out the whole design. The TMR cells will be added at netlist level and not in the RTL code and is further discussed in section 4.

During the architectural implementation phase the design will be implemented to be highly configurable. All cores in Aeroflex Gaisler's GRLIB IP library are highly configurable and the design can easily be parametrisable with regard to memory sizes, number of cores, and key core characteristics. The design described in this document is the baseline design.

2.2 LEON4FT SPARC V8 Processor Cores

2.2.1 Description

2.2.1.1 Overview

The system will contain four LEON4FT cores based on the LEON4 processor. LEON4 is a 32-bit processor core conforming to the IEEE-1754 (SPARC V8) architecture. It is designed for embedded applications, combining high performance with low complexity and low power consumption.

The following subsections describe the configuration of individual parts of the LEON4FT processor. The configuration is summarised in the last subsection.

2.2.1.2 Integer Unit

The configuration of the LEON4FT integer pipeline will be similar to what was implemented in the LEON3FT GINA study [RD15] with eight register windows, BCH SEU protection of register file, and hardware multiply and divide units. The multiplier will be 32x32 bits with two-stage internal pipeline, allowing single-cycle integer multiply operations.

2.2.1.3 Floating Point Unit

The implementation settings of the hardware floating point support has not been fully determined at the time of writing since neither option has been fully validated with the LEON4FT. There are two main options; 1) The system will contain two Aeroflex Gaisler GRFPU cores for all floating-point operations, where one GRFPU core is shared between two processors, with a 4-entry instruction FIFO added to the FPU controller. 2) Dedicated GRFPU unit for each processor core.

1) A shared GRFPU will be used together with a 4-entry instruction FIFO that has been shown to provide improvements of performance since several FPU instructions can be placed into the FIFO while non-FP instructions can still flow through the processor pipeline.

Aeroflex Gaisler experience shows that benchmarks written in a high-level language do not load the FPU by more than 30%. Floating point operations often have data dependencies which lead to stalls in one instruction stream. This allows an FPU to be efficiently shared between two processors with a typical performance decrease of 5% compared to using a dedicated GRFPU with 4-entry instruction FIFO for each processor. If an application running on one processor can achieve a very high FPU load this will still not lead to starvation of the other processor sharing the FPU. Round-robin arbitration between the processor cores guarantee that each processor can receive 50% of the FPU.

2) A dedicated GRFPU for each processor will provide a FPU solution with higher integration into the processor pipeline. Since the solution with the dedicated GRFPU lacks an instruction FIFO there will be a performance penalty incurred, compared to using a GRFPU together with a floating point controller that provides the instruction FIFO. However, a dedicated GRFPU for each processor core with high pipeline integration will allow the floating point register file to be tightly integrated with the processor's register file. This simplifies design and the implementation of radiation tolerance features. With a tightly integrated register file, accesses to the processor's floating point registers will receive a performance increase which is expected to mitigate the loss of the 4-word instruction FIFO.

The selection of a shared or dedicated GRFPU solution will be made during the architectural design phase. Both options are viable with the solution with dedicated GRFPU cores leading to less risk since it does not require a complex floating point controller unit handling the instruction FIFO.

2.2.1.4 Timer Unit

The LEON4 core will be extended to have a dedicated timer unit in the processor. The timer unit will include five timers and have watchdog functionality that can be used to either halt or reset the processor. The watchdog is utilised to guard the system against fault conditions where the system locks up and against infinite loops. A processor core will be able to detect if it has started execution as a result of watchdog reset or system reset.

The per-processor watchdog functionality is not intended to be used in SMP configurations. Although there is nothing in the hardware that prevents this functionality to be used in SMP configurations, software must be able to handle one core being halted or reset. Operating systems with SMP support often have other means to detect if one processor is stuck. To protect against scenarios where the whole system freezes, the watchdog in the global timer unit can be used.

The watchdog will be disabled after chip reset if the external PROMPRES signal indicates that the system lacks PROM to allow uploading of system software via RMAP. Otherwise the watchdog will be enabled.

2.2.1.5 Interrupt controller

The LEON4 core will be extended to have a dedicated interrupt controller in the processor. The dedicated interrupt controller's primary use is in ASMP configurations. During SMP operation the system should use the external multiprocessor interrupt controller. See section 2.32 for a description of the interrupt infrastructure.

2.2.1.6 Memory Management Unit

The standard LEON3/4 SRMMU will be used for memory management, with standard 4 KiB pages. The MMU uses hardware table-walk, and is fully compatible with the SPARC V8

SRMMU standard. The number of TLB entries is configurable, and the baseline configuration is to have 16 entries for instructions and 16 entries for data.

2.2.1.7 L1 Cache

Each LEON4FT core will have a 32 KiB L1 cache memory. The memory will be organised as 16 KiB for instruction and 16 KiB for data, with 4x4 KiB respectively 4x4 KiB organization and LRU replacement.

The cache will be accessed using virtual addresses, and duplicate physical tags will be used to allow cache snooping. The data cache controller from LEON3 has been modified to implement multi-way aliasing detection. This will avoid that one physical address that is mapped onto several virtual addresses can exist in multiple locations of the data cache. The cache RAM will be protected against SEU using four parity bits per 32-bit cache word and tag. The write policy for the data cache is write-through.

Locking of the L1 cache will not be supported.

2.2.2 Performance counters

Each processor core will have support for the following performance counters:

- Clock cycle counter
- Completed branches
- Taken branches that were mispredicted
- FPU arithmetic instructions, not counting floating point register file operations
- Load instructions
- Store instructions
- Other instructions, divided into the four SPARC types (LDST, CALL, BRA, Others)
- Control flow instructions (call, branch, jmp, trap, return)
- Instruction cache misses
- Data cache misses
- MMU Instruction TLB misses
- MMU Data TLB misses

The counters will be controlled from internal processor registers. Since counters are expensive to implement, it will not be possible to enable all the above counters at the same time. All counters can be disabled and one set of performance counters is available per CPU core.

In order to profile applications running on-top of an operating system it will be possible to specify if the counters should apply to software that executes in user mode, in supervisor mode, or both.

2.2.3 Block moves

The LEON4FT processor will be extended with write combining in the write-buffer. This will allow the processor to efficiently perform block moves and provide a speed-up for operations such as memcpy(..) without the need for special optimisation of existing software.

2.2.4 Configuration

The configuration of the processors can be summarised with:

- 128-bit wide AHB master interface
- 8 register windows
- GRFPU floating point unit
- Support for Debug Support Unit (DSU)
- Support for SPARC V8 MUL and DIV instructions
- Hardware multiply and divide units
- Hardware multiplier will be 32x32 bits pipelined, allowing single-cycle integer multiply operations
- Support for SPARC V8E SMAC/UMAC instructions
- 8 hardware watchpoints (Implementation dependant, actual number will be determined during the architectural design phase)
- 32 KiB L1 cache memory
  - 16 KiB for instruction cache, 4x4 organization and LRU replacement
  - 16 KiB for data cache, 4x4 KiB organization and LRU replacement
- Data cache snooping with extra physical tags
- Memory Management Unit fully compatible with the SPARC V8 SRMMU standard
  - Standard 4 KiB pages
  - Hardware table-walk mechanism
  - 16 TLB entries for instructions and 16 TLB entries for data
- Support for processor power-down
- Support for single-vector trapping
- Default reset start address will be configurable via registers in the NGMP system
- Multi-processor support
- Local timer unit, in each processor core, implementing 4 timers and watchdog functionality
- Local interrupt controller in each processor core
- Performance counters included in each processor core

2.3 L2 Cache

2.3.1 Description

The L2C core from GRLIB will implement the level 2 (L2) cache in the memory hierarchy. The L2 cache connects the Processor bus to the memory controller via a secondary (backend) AHB bus, named Memory bus in illustration 1 on page 10. The cache TAG and cache data is protected by BCH coding featuring a single error correction and double error detection. The L2 cache size will be at least 256 KiB and configured as a 4-way set associative cache (the cache configuration will depend on the technology used for the actually chip implementation).

The L2 cache core will include a memory protection unit that will provide read/write/execute protection of memory blocks that are specified via the core's register interface. The protection functionality can be enabled even if normal L2 cache operation is disabled. The core will also include counters for misses and total number of accesses to aid in software optimisation.

The L2 cache will include functionality to lock one way so that L2 cache memory can be used as on-chip RAM.

By monitoring the AMBA HPROT signal the L2 cache can be configured by software not evict or cache data on accesses from I/O cores. This will prevent I/O accesses from causing L2 cache trashing.

2.3.2 Configuration

The L2 cache will be implemented with the following characteristics:

- Cache is by default disabled at reset
- The core will look at the AMBA HPROT signal to not evict or cache data on accesses from the I/O cores
- The default write-policy will be copy-back
- Least-Recently-Used (LRU) replacement scheme
- 4 sets
- 64 KIB per set, 256 KiB total cache memory
- 32 byte cache line size
- Cache hit counter will be included

2.4 DDR2 Memory Interface

2.4.1 Description

A 96-bit DDR2 interface will be used as the baseline for the external memory interface of the NGMP system. The DDR2 controller will be implemented using the standard DDR2SPA core from GRLIB, which is compatible with the JEDEC JESD79-2E [RD2] specification. It will be amended with the Reed-Solomon code used in the FTMCTRL SDRAM controller in GRLIB, providing the capability to detect and correct two or four nibble (4-bit) errors in a 64-bit data word, depending on the number of check bits.

The DDR2 controller will include a side-band signal to indicate when an error has been corrected by the error correction logic. This side-band signal will be connected to the memory scrubber to allow the scrubber to keep statistics on how many correctable errors that are encountered.

The DDR2 interface will, if possible, share pins with the SDRAM interface. This means that the system will be able to interface with SDRAM or DDR2 SDRAM, but not both at the same time. The DDR2 SDRAM can be clocked at 1x or 2x the system frequency. Section 3.3 describes the external signals that determine the configuration used.

The DDR2 controller's register interface allows software to perform diagnostic accesses to read/write the EDAC check bits.

2.4.2 Reed-Solomon Codecs

The Reed-Solomon implementation for DDR2SPA will support two Reed-Solomon codes based on the following codecs:

- RS(24, 16, 8, E=1) which has 16-bit data, 8 check bits and corrects 4-bit error in one nibble
- RS(40, 32, 8, E=1) which has 32-bit data, 8 check bits and corrects 4-bit error in one nibble

By interleaving the above codes the following codes can be attained:

- Encoding A: RS(24, 16, 8, E=1) with interleave depth of 4 that gives:
  - 64-bit data,
  - 32 check bits
  - corrects 4-bit error in four nibbles.

- Requires a memory word length of 96 bits
- Encoding B: RS(40, 32, 8, E=1) with interleave depth of 2 that gives:
  - 64-bit data
  - 16 check bits
  - corrects 4-bit error in two nibbles.
  - Requires a memory word length of 80 bits

The encoding to be used will be selected by software. From an implementation point of view the selection of the codecs above is attractive since the input to logic for codec A can be padded with zeros to receive encoding B.

For optimal error correction capabilities using encoding B, the DDR2 memory should be implemented using 4-bit devices. Since encoding B requires a memory word length of 80 bits, one memory bank will consist of twenty DDR2 devices, with a total of 1 GiB usable memory when 512 Mib DDR2 devices are used and the maximum supported memory size of 2 GiB when 1024 Gib DDR2 devices are used. The maximum size of 2 GiB DDR2 memory comes from the DDR2 controllers map into AMBA address space, which is a 2 GiB memory area.

With encoding A, which can detect four nibble-errors, one memory bank can be implemented with twelve 8-bit wide DDR2 devices. To attain the maximum supported memory size of 2 GiB, the twelve 8-bit wide DDR2 devices must be 2 Gib devices.

The conclusions regarding maximum attainable memory size assumes that the DDR2 controller is implemented with two chip select signals. More memory can be attained by using more chip selects and connecting more than one chip to a data line. This would place higher demands on the drive capability of the pads connected to the DDR2 devices.

2.4.3 Configuration

The characteristics of the memory devices connected to the DDR2SPA core is configured by software via the DDR2SPA core's register interface. The DDR2SPA will be configured with a 64-bit wide data bus (plus support for 32 check bits giving a total memory controller word length of 96 bits) and with support for 8 banks per DDR2 device. The DDR2SPA will operate at the same or 2x the frequency of the Memory bus, which reduces the need for synchronization. The core will support 64 Mib, 128 Mib, 256 Mib, 512 Mib, 1 Gib, and 2 Gib devices with 9 – 12 column-address bits, up to 15 row-address bits, and eight internal banks.

2.5 SDRAM Controller

2.5.1 Description

If DDR2 cannot be used for the NGMP, the backup solution is SDRAM. The baseline is to include both types of controllers on-chip and share pins between the interfaces.

The SDRAM controller will be based on the FTSDCTRL core from GRLIB. The Reed-Solomon implementation of the FTMCTRL core will be updated and ported to FTSDCTRL providing the capability to detect and correct two or four nibble (4-bit) errors in a 64-bit data word read from external memory. The characteristics of the Reed-Solomon encoding is described in section 2.4.2. The SDRAM controller has support for two chip select signals, and using 512 Mib devices with a width of 4-bits will yield a memory size of 2 GiB.

The SDRAM interface will, if possible, share pins with the DDR2 interface. This means that the system will be able to interface with SDRAM or DDR2 SDRAM, but not both at the same time. The SDRAM can be clocked at 1x or ¼x the system frequency, depending on the capabilities of the target technology and maximum attainable system frequency this may be changed to 1x or ¼x the system frequency. The final decision will be taken when synthesising for ASIC. Section 3.3 describes the external signals that determine the configuration used.

The SDRAM controller's register interface allows software to perform diagnostic accesses to read/write the EDAC check bits.

2.5.2 Configuration

The characteristics of the memory devices connected to the FTSDCTRL core is configurable by software via the FTSDCTRL core's register interface. The FTSDCTRL core will be implemented with a memory interface of 64 data bits plus support for 32 check bits giving a total memory controller word length of 96 bits.

2.6 On-chip SDRAM

2.6.1 Description

Among the requirements for the NGMP is a minimum of 32 MiB on-chip RAM. Such large RAM blocks are typically implemented as SDRAM. It is at the time of writing not known if the target technology will provide suitable memory to include in the NGMP system. Should SDRAM be available on the target technology the SDRAM controller will be based on the FTSDCTRL core from GRLIB. Reed-Solomon EDAC will be ported from the GRLIB FTMCTRL core. Please see the Reed-Solomon discussion in section 2.4.2 for a description of the available codecs.

The use of on-chip SDRAM may lead to the AHB data bus width of the Memory AHB bus to be extended to 256 bits, if the on-chip SDRAM is able to deliver 256 bits of data in one clock.

The SDRAM controller register's interface allows software to perform diagnostic accesses to read/write the EDAC check bits.

2.6.2 Configuration

The width of the SDRAM interface will be defined when more is known about the target technology.

2.7 Memory Scrubber

2.7.1 Description

A memory scrubber will be implemented that will perform memory scrubbing of external memory and will also support scrubbing of a potential on-chip SDRAM memory. The scrubber will be placed on the Memory bus and perform both configurable periodic scrubbing and complete memory re-generation after a device failure. The memory scrubber will operate in the background without interrupting any of the processors. The core will also have functionality to assert an interrupt when an error, or series of errors, is detected.

2.7.2 Modes of Operation

The memory scrubber will have three main modes of operation:

- Initialisation: The scrubber will write a predefined pattern to specified ranges in memory. This needs to be done after power-on in order to initialise EDAC check bits. The starting address, range and pattern to be written will be software configurable.
- Scrubbing: The scrubber will read blocks of memory and keep track of corrected errors. If a corrected error is detected the scrubber will use locked accesses to read the memory location once more and write back the data as an atomic access. This will correct the error in main memory. The starting address, range and the interval with which the address range will be scrubbed is software configurable.
- Memory re-generation: In this mode the scrubber will use locked accesses to atomically read and write back all words within a specified address range. The starting address and range to be re-generated will be software configurable.

2.7.3 Scrubber Bandwidth

Since the scrubber will be placed on the Memory bus it will compete for access to main memory with the L2 cache. This will lead to longer latencies for L2 cache accesses. The impact of memory scrubber operation on L2 cache access latencies will be investigated during the architectural design phase and the scrubber and Memory bus arbitration will be configured so that the performance impact on L2 cache operation is minimised, while at the same time ensuring that memory scrubbing can be performed in reasonable intervals. The memory scrubber's speed when scrubbing memory will be configurable through the scrubber's register interface by having the ability to insert a delay between each performed operation.

Assuming that the maximum size of 2 GiB of memory is to be scrubbed with an interval of 10 minutes the memory bus utilization caused by the memory scrubber will be less than 0.7%.

2.7.4 Memory Initialisation

The flow of the memory scrubber during initialisation will be:

1. Start at the lowest address in the address range defined by software via the memory scrubber's register interface.
2. Write one cache line (the size of the write FIFO in the DDR controller)
3. Increase the address with the size of one cache line, if full range has been written go to 5.
4. Wait for the number of cycles specified by the delay counter and go to 2.
5. Done, optionally assert interrupt.

Initialisation of the entire main memory that will be used should be done during boot of the NGMP system. Software will control the start of memory initialisation by writing to the memory scrubber's register interface. The scrubber will then start memory initialisation by continuously writing a fixed pattern to the entire main memory area. The maximum main memory size in the NGMP system is 2 GiB and it is approximated that the memory scrubber will be able to initialize the entire 2 GiB memory area in less than four seconds in a system running at 400 MHz. Software does not need to wait for four seconds before taking the

system operational. By first initialising a part of memory software can start executing from the initialised part while the scrubber initialises the rest of main memory.

2.7.5 Memory Scrubbing and Re-generation

The DDR2 memory controller core will signal corrected errors via a side-band signal that is connected to the memory scrubber. The scrubber will include logic to count how many corrected errors that are generated during a scrub of the memory. The scrubber will also record errors corrected as a result of memory accesses from the L2 cache. Counters will keep track of the total number of errors encountered within a particular time frame and also the total number of errors encountered while reading a cache line. If either value is over a configurable threshold, which is configurable by software, the core will assert an interrupt and can be configured by software to also stop the scrubbing operation. The thresholds are utilised to prevent the scrubber from essentially performing a complete memory re-generation using, time consuming, atomic operations when the system may need the bandwidth on the memory bus available for the L2 cache. When software discovers a build-up of errors it will have the option to either increase the rate at which the memory is scrubbed, to allow the scrubber to continue with the scrubbing operation, to perform complete memory generation, or to wait until some critical stage of operation has passed.

The flow of the memory scrubber during scrubbing will be:

1. Start at the lowest address in the address range defined by software via the memory scrubbers register interface.
2. Read one cache line
3. Check if there were any correctable errors during the read. If no errors were corrected go to 5.
4. Perform atomic read/write sequence to correct the error(s) in main memory for all words which had an error corrected.
5. Increase the address with the size of one cache line
6. Wait for the number of cycles specified by the delay counter
7. If full address range has been written go to 1 else go to 2.

The flow of the memory scrubber during memory re-generation will be:

1. Start at the lowest address in the address range defined by software via the memory scrubbers register interface.
2. Lock the Memory AHB bus and read eight words, followed by a write back of the 8 words, atomically.
3. Increase the address, if full range has been written go to 5.
4. Wait for the number of cycles specified by the delay counter and go to 2.
5. Done, optionally assert interrupt.

The memory scrubber will have a 32 byte FIFO in order to save the data read during memory generation. The same buffer will be used for memory initialisation and will then typically be set to all zeroes. However, software will be able to initialise the FIFO with other values allowing it to write fixed patterns to the complete memory and thus performing simple memory tests.

The memory scrubber will also include functionality from the AHB status register (AHBSTAT) to be able to save the address of an access triggering an AMBA ERROR response from the DDR2 controller.

Since scrubbing of external memory and on-chip SDRAM should be done in parallel the scrubber functionality will be duplicated if on-chip SDRAM is available on the NGMP.

2.7.6 Configuration

Not applicable since the core is a new development.

2.8 AHB/AHB Bridge Connecting Debug AHB Bus

2.8.1 Description

The uni-directional AHB/AHB bridge from GRLIB will connect the Processor and Debug buses. The processor side is 128-bit wide, while the debug side will be 32-bit wide. The Debug bus will house the DSU and the dedicated USB, JTAG, Spacewire RMAP debug links, and can be configured via an external signal to connect the Ethernet debug links. The bridge will separate traffic flowing between the DSU and the dedicated debug links from the traffic on the Processor bus.

All cores on the Debug AHB bus can be disabled via the clock gate register described in section 2.26. The reset value for the clock gate register bits applicable to cores on the Debug AHB bus are set from the external DSU enable signal. If the DSU is disabled, all cores on the Debug AHB bus will be gated off after reset. When the debug cores are disabled the bridge will be configured so that it drives its AHB master interface connected to the Processor bus to a safe state and does not propagate anything from the Debug bus.

2.8.2 Configuration

The lists below summarises the configuration for the AHB/AHB bridge.

- Support for data pre-fetching with a read buffer size of eight 32-bit words
  - When pre-fetching data, 128-bit accesses will be used to reduce load on the Processor bus
- Write buffer with room for eight 32-bit words
  - When emptying the write buffer, 128-bit accesses will be used if possible
- The bridge will perform incremental read bursts with a maximum length of eight 32-bit words

2.9 DSU4 – LEON4 Hardware Debug Support Unit

2.9.1 Description

2.9.1.1 Overview

To simplify debugging on target hardware, the LEON4FT processor implements a debug mode during which the pipeline is idle and the processor is controlled through a special debug interface. The LEON4 Debug Support Unit (DSU) is used to control the processor during debug mode and provides functionality for hardware breakpoint/watchpoints, instruction and AHB trace buffers, and to continue execution for a given amount of time. It also supports debugging of multiple processing cores in parallel, with individual debug

mode settings. The trace buffers operate in a circular buffer mode, and can only be read out when the processor is halted. Performance measuring is implemented by counting the appropriate instruction for a fixed given time (e.g. 1E9 clocks) and then latching the values in holding registers. By periodically sampling the holding registers, a statistical performance view can be built.

The DSU acts as an AHB slave and can be accessed by the USB, JTAG, SpaceWire and Ethernet debug links that are connected to the Debug AHB bus.

The DSU provides separate control of each processor. One or several processors can be put into debug mode while the other processors continue with normal operation.

2.9.1.2 Operation

The DSU control registers can be accessed at any time, while the processor registers, caches and trace buffer can only be accessed when the processor has entered debug mode. In debug mode, the processor pipeline is held and the processor state can be accessed by the DSU. Entering the debug mode can occur on the following events:

- executing a breakpoint instruction (ta 1)
- integer unit hardware breakpoint/watchpoint hit (trap 0xb)
- rising edge of the external break signal (DSUBRE)
- setting the break-now (BN) bit in the DSU control register
- a trap that would cause the processor to enter error mode
- occurrence of any, or a selection of traps as defined in the DSU control register
- after a single-step operation
- one of the processors in a multiprocessor system has entered the debug mode, the DSU can be configured to force all cores into debug mode on this event.
- DSU AHB breakpoint hit

The debug mode can only be entered when the debug support unit is enabled through an external signal (DSUEN). When the debug mode is entered, the following actions are taken:

- PC and nPC are saved in temporary registers (accessible by the debug unit)
- an output signal (DSUACT) is asserted to indicate the debug state (this signal signals the state of CPU 0).
- the timer unit is (optionally) stopped to freeze the LEON timers and watchdog

The instruction that caused the processor to enter debug mode is not executed, and the processor state is kept unmodified. Execution is resumed by clearing the BN bit in the DSU control register or by deasserting DSUEN. The timer unit will be re-enabled and execution will continue from the saved PC and nPC. Debug mode can also be entered after the processor has entered error mode, for instance when an application has terminated and halted the processor. The error mode can be reset and the processor restarted at any address.

2.9.1.3 AHB Trace Buffer

The AHB trace buffer consists of a circular buffer that stores AHB data transfers. The address, data and various control signals of the AHB bus are stored and can be read out for later analysis. In addition to the AHB signals, the DSU time tag counter is also stored in the trace. In the NGMP system, the trace buffer input will come from the Processor bus.

When the trace buffer is enabled, each AHB transfer is stored in the buffer in a circular manner. Tracing is stopped when the EN bit is reset, or when a AHB breakpoint is hit. Tracing is temporarily suspended when the processor enters debug mode. Note that neither

the trace buffer memory nor the breakpoint registers can be read/written by software when the trace buffer is enabled.

In order to limit the amount of data placed in the AHB trace buffer, the following filters will be available:

- Filter to select which masters that should be included in the trace
- Filter to select which slaves that should be included in the trace
- Filter to select which type of access (read/write) that should be included in the trace
- Filter to select accesses specified by address ranges

The above filters can be combined so that the user can select to only include a certain type of accesses between master X and slave Y in the trace.

2.9.1.4 Instruction Trace Buffer

The instruction trace buffer consists of a circular buffer that stores executed instructions. The instruction trace buffer is located in the processor, and read out via the DSU. During tracing, one instruction is stored per line in the trace buffer with the exception of multi-cycle instructions. Multi-cycle instructions are entered two or three times in the trace buffer, with bits set in the buffer to mark the order of the entries. When the processor enters debug mode, tracing is suspended. The trace buffer and the trace buffer control register can be read and written while the processor is in the debug mode. During instruction tracing (processor in normal mode) the trace buffer and the trace buffer control register can not be accessed.

The instruction trace buffer will be extended so that it will be possible to filter on a certain instruction type, such as branch instructions.

2.9.1.5 Data Watchpoints

The DSU will include data watch points. The number of watchpoints and their implementation will be defined during the activity's architectural design phase. Data watchpoints are subject to trade-off depending on timing and area constraints.

2.9.1.6 Data Area Monitoring

The DSU will include functionality to trace accesses made to a specified area in memory. The purpose of this functionality is to be able to visualise how application data changes over time. Implementation details of this functionality will be determined during the activity's architectural design phase.

2.9.1.7 AHB Performance Monitoring

The DSU4 will be implemented with support for AHB performance monitoring:

- One clock counter to keep track of the time that has passed during collection of values
- Accumulated bus usage during the sample time, that is; number of cycles that the bus has been occupied
- A mask register that selects one or more masters for which the following statistics should be collected
  - Number of read accesses
  - Number of write accesses

- Number of words transferred
- Number of wait states inserted by slave (accumulated and maximum)
- Maximum bus request latency
- Number of SPLIT responses received
- Accumulated number of cycles waiting to be granted after a SPLIT response
- Number of idle cycles

The performance counters can be disabled/enabled via the DSU's register interface and the sample time is the time during which the performance counters are enabled. The performance counters may require several 32-bit counters. These counters may significantly increase the system's area and power requirements. Should area and power issues arise, the counters are subject to trade-off.

2.9.2 Configuration

The characteristics of the debug support unit for use in the NGMP are listed below:

- 4 KiB AHB trace buffer, providing a trace of the Processor bus
- The number of watchpoints and the size of the instruction trace buffer are configured for each LEON4FT core. The LEON4FT cores will be configured to have 8 watchpoints and a 4 KiB instruction trace buffer.

2.10 JTAG Debug Link Controller

2.10.1 Description

The JTAG debug interface (AHBJTAG) from GRLIB provides access to on-chip AMBA AHB bus through JTAG. The JTAG debug interface implements a simple protocol that translates JTAG instructions to AHB transfers. Through this link, a read or write transfer can be generated to any address on the AHB bus.

The JTAG debug link is only expected to be used under development and will be disabled with the use of the external DSU enable signal under other circumstances. The expected data throughput is roughly 0.5 Mb/s.

2.10.2 Configuration

The AHBJTAG core will be implemented with the following characteristics:

- Two-stage synchronisers between clock regions
- JTAG IDCODE instruction code: 9
- Manufacturer ID: 804
- Part number: TBD
- Version number: 0

2.11 USB Debug Link Controller

2.11.1 Description

The Universal Serial Bus Debug Communication Link (USBDCL) from GRLIB provides an interface between an USB 2.0 bus [RD9] and an AHB bus. An external High-/Full-Speed UTMI+ Low Pin Interface (ULPI) [RD10] transceiver is needed to connect to the USB. The USBDCL is an AHB master and provides read and write access to the whole AHB address space using a simple protocol over two USB bulk endpoints. In other words, the USB Debug Link Controller

can only be used to allow a debugger to make AHB accesses on the system. It can not be used to attach USB devices or be used to make the system act as a general USB device.

The USB debug link is only expected to be used under development and will be disabled with the use of the external DSU enable signal under other circumstances. The expected data throughput is roughly 20 Mb/s.

Note: The USB Debug Link Controller is quite complex and is subject to trade-off.

2.11.2 Configuration

The USBDCL will be configured to use a maximum burst length of eight words.

2.12 SpaceWire RMAP Target Debug Link Controller

2.12.1 Description

In addition to the four SpaceWire link interfaces described in section 2.15, the system has one SpaceWire Remote Memory Access Protocol (RMAP) target core connected to the Debug bus. The core is based on GRSPW2 from GRLIB. The expected data throughput is roughly 20 Mb/s.

2.12.2 Configuration

The GRSPW2 RMAP target core will have the following configuration:

- Hardware RMAP target
- 16 entries in the 32-bit receiver and transmitter AHB FIFOs
- 64 entries in the 9-bit receiver N-Char FIFO
- Eight buffers to hold RMAP replies
- Fault-tolerance in buffers
- One port
- The receiver type will be double data rate sampling
- The transmitter will use single data rate
- Both the receiver and transmitter will use the same clock to reduce the amount of needed synchronisation registers

2.13 Uni-directional AHB Bridge with Protection Functionality

2.13.1 Description

An extended uni-directional AHB bridge will connect the master I/O bus to the Processor bus. The added protection functionality will prevent connected I/O devices from overwriting protected system data or accessing memory areas of other cores. AHB bridge functionality such as data pre-fetching and posted writes will be included.

The protection functionality is intended to be implemented in the form of an IOMMU that will provide address translation and access restrictions. If a full IOMMU solution becomes too costly in terms of area or performance decrease the backup option is to only implement access restrictions, possibly in the form of a bit vector or by including registers that define an allowed address range for each master on the master I/O bus.

The following items give a rough summary of the current plans for the IOMMU:

- The IOMMU will be based on the uni-directional AHB bridge and SRMMU cores from GRLIB
- The IOMMU will have a similar interface to an IOMMU supported by existing operating systems
- The IOMMU may able, but will not be required, to use the same set of page tables as the processor MMU
- Each device on the I/O AHB bus may have its own set of page tables
- A goal is to have the ability to place devices in groups where all devices in a group share the same page tables
- When disabled the IOMMU will be in pass-trough mode (no address translation) and have the same basic functionality as the GRLIB uni-directional AHB bridge.
- The IOMMU may include a page walk mechanism in hardware or page misses may be handled by software.
- The size and structure of the IOMMU's Translation Lookaside Buffer (TLB) may need to be different from the SRMMU TLB implementation due to the difference in locality for I/O DMA traffic and the IOMMU's ability to place devices in groups.
- When an access violation (page fault) occurs, the IOMMU will issue an AMBA ERROR response to the master. Software will be able to detect this ERROR response though the master's status reporting. The IOMMU will also include the SRMMU fault register, that will record the master and address that triggered the AMBA ERROR response. The IOMMU will also be able to assert an interrupt on access violations.
- The IOMMU will be disabled (in pass-through mode) after reset in order to allow upload of system software via RMAP.

A more detailed discussion of the motivation for including an IOMMU can be found in the Architecture Exploration Report [AD4].

2.13.2 Configuration

Configuration options will be determined during the architectural design phase.

2.14 Ethernet Controllers

2.14.1 Description

The NGMP system will include two Ethernet interfaces, based on the GRETH_GBIT core from GRLIB, allowing 1 Gbit/s operation. GRETH_GBIT provides an interface between an AMBA AHB bus and an Ethernet network. It supports 10/100/1000 Mbit speed in both full- and half-duplex. The AMBA interface consists of an APB interface for configuration and control, and an AHB master interface which handles the dataflow. The dataflow is handled through DMA channels. There is one DMA engine for the transmitter and one for the receiver. Both share the same AHB master interface. GRETH_GBIT has internal RAM that allows the core to buffer a complete packet

The Ethernet interface supports the MII and GMII interfaces to connect to an external PHY. The GRETH also provides access to the MII Management interface which is used to configure the PHY. Some of the supported features for the DMA channels are Scatter Gather I/O and TCP/UDP over IPv4 checksum offloading for both receiver and transmitter.

The Ethernet Debug Communication Link protocol (EDCL) provided in the GRETH cores will be enabled to allow a fast and efficient debug link for software development. The EDCL contains no user accessible registers, apart from registers for setting the EDCL IP and MAC address, and runs in parallel with the DMA channels. In order to accommodate at least part

of the user base, the EDCL default IP address will be set to 192.168.0.1x and 192.168.0.2x for the first and the second GRETH, respectively. The 'x' denotes the value of the four least significant bits of the address, the value of these bits are read during reset from the eight first bits of the GPIO port. The first GRETH_GBIT EDCL core uses bits 0:3 and the second bits 4:7. This allows a user to set the EDCL address using external pull-ups or pull-downs. If a user wants to use another address she will need to connect to the system using one of the other debug links and then configure the EDCL addresses, or set up boot software to set the EDCL addresses. The expected data throughput is roughly 100 Mb/s.

The GRETH EDCL AHB master interface can be connected to the Debug bus, as seen in illustration 2, with the help of an external signal (one signal is present for each core). Otherwise the EDCL AHB master interface will be connected to the Master I/O bus. An EDCL will be disabled after system reset if it is connected to the Debug bus and the Debug bus has been disabled using the external DSU_EN signal, otherwise the EDCL will be enabled.

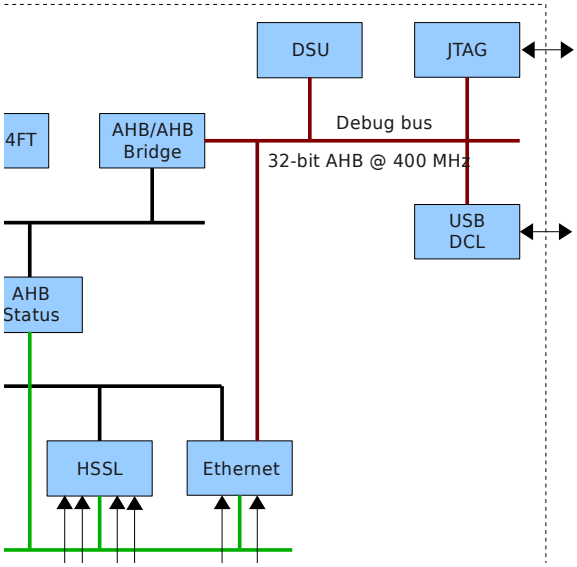

Illustration 2: Ethernet connection to Debug AHB bus

2.14.2 Configuration

The GRETH_GBIT core will be implemented with the following characteristics:

- EDCL buffer size of 2 KiB, allowing debug link throughput of 90 – 100 Mbit/s
- EDCL IP and MAC addresses will be configurable via the core's APB register interface.
- Default EDCL addresses will be 192.168.0.1x for the first EDCL and 192.168.0.2x for the second. The x denotes the four least significant bits of the addresses and the value is determined by GPIO[7:0] when coming out of system reset.
- 8 word burst length
- Support for MDIO interrupts
- Support for reception and hardware filtering of multicast packets

2.15 SpaceWire Controllers

2.15.1 Description

The system has four ECSS-E-50-12 [RD7] compatible SpaceWire link interfaces directly on-chip. The links will be based on the GRSPW2 core from GRLIB, implementing a Remote Memory Access Protocol (RMAP) target as defined in the ECSS standard ECSS-E-50-11 [RD8]. The transmitter uses Single Data Rate (SDR) to achieve 200 Mbit/s while the receiver use Double Data Rate (DDR) sampling registers which results in two times oversampling for a 200 Mbit input. This configuration thus supports 200 Mbit/s in both directions and only one additional clock domain is needed for all cores instead of one per receiver and one in total for all transmitters. If the pin-out permits, the GRSPW2 core will be implemented with redundant link capability, with two link drivers per core for redundancy.

The RMAP target of the SpaceWire cores will be enabled or disabled after power-up and system reset depending on the value of a top-level signal for each core. The RMAP target can also be enabled or disabled by software, regardless of the initial values of the top-level signals.

2.15.2 Configuration

The GRSPW2 cores will have the following configuration:

- Hardware RMAP target
- Hardware RMAP CRC logic for DMA channels
- 16 entries in the 32-bit receiver and transmitter AHB FIFOs
- 64 entries in the 9-bit receiver N-Char FIFO
- Receiver unaligned write support. The receiver can write any number of bytes to any start address without writing any excessive bytes.
- Eight buffers to hold RMAP replies
- Fault-tolerance in buffers
- Two ports per core
- One DMA channel
- The receiver type will be double data rate sampling
- The transmitter will use single data rate
- Both the receiver and transmitter will use the same clock to reduce the amount of needed synchronisation registers
- The maximum link bit rate will be at least 200 Mbit/s

2.16 High Speed Serial Link Controller

2.16.1 Description

The availability and specification of the HSSL IP cores to be integrated within the DSM ASIC platform is at the time of writing very limited. It is therefore difficult to comment on the possibilities of using these IP cores for various high-speed interconnects.

ESA requires Spacefibre to be instantiated in the NGMP. This will be done if working IP is provided. As backup, a simple descriptor based DMA core based on GRETH_GBIT or GRSPW2 can be used.

2.16.2 Configuration

Unknown.

2.17 PCI Controller

2.17.1 Description

The PCI Master/Target core (GRPCI) from GRLIB will act as a bridge between the PCI bus and the Master and Slave I/O buses. The core will be connected to the PCI bus through two interfaces, a target and master. The PCI master interface will be the main I/O extension interface for off-chip devices, as it is compatible with the current range of processors. The PCI interface will be 32-bit with 66 MHz operation (if the timing of the target technology allows it). The GRPCI target and DMA interface will be placed on the Master I/O bus while the GRPCI AHB slave interface will be placed on the Slave I/O bus.

The PCI and AMBA interfaces belong to two different clock domains. Synchronization is performed inside the core through FIFOs.

Aeroflex Gaisler is currently developing a new PCI core, GRPCI2, which provides the same functionality as the GRPCI. If GRPCI2 passes validation in time for the NGMP development the new core will replace GRPCI as the new core will have improved performance.

2.17.2 Configuration

The GRPCI will be implemented with the following characteristics:

- Support for PCI DMA controller
- Support for pre-fetching data for the 'memory read' command
- Least significant implemented bit of BAR0 and PAGE0 registers: 28
  - Giving a BAR size of 256 MiB
- Least significant implemented bit of BAR1 and PAGE1 registers: 26
  - Giving a BAR size of 64 MiB
- FIFO depth of eight 32-bit words
- PCI device ID: 0x0061
- PCI vendor ID: 0x1AC8
- Class code base class 0x0B (processor), sub class 0x0B4000, revision 0
- Support for PCI master interface
- AHB slave address space of 1 GiB
- Support for 4 PCI input interrupts
- One interrupt signal connected from Secondary IRQCTRL 1 to PCI interrupt A, see section 2.32 for more information on the interrupt infrastructure.
- Two-stage clock domain synchronisation
- Core must be connected to little endian PCI bus
- Support for driving PCI reset when acting as a host
- Support for AMBA AHB bursts of 8 words

2.18 Uni-directional AHB Bridge Connecting Peripheral Slave Interfaces

2.18.1 Description

The uni-directional AHB bridge (AHB2AHB) from GRLIB will connect the 128-bit Processor bus to the 32-bit Slave I/O bus. The AHB2AHB bridge will be extended to let the processor

make 128-bit accesses when performing a 32 byte cache line fill. The bridge will fill its read buffer using 32-bit accesses to the PROM and then let the processor read out the buffer using 128-bit accesses.

2.18.2 Configuration

The core will be configured with:

- A read FIFO capable of buffering eight 32-bit words
- Interrupt forwarding and synchronization
- A write FIFO capable of buffering four 32-bit words

2.19 PROM/IO Controller

2.19.1 Description

A 16-bit PROM/IO controller will be used to attach external boot PROM and simple memory mapped I/O units. The controller will be based on the FTSRCTRL8 from GRLIB and will include support for parallel Flash memory devices. The waitstates generation will be extended to allow a bus cycle of at least 300 ns at 400 MHz system frequency.

The PROM/IO Controller will be extended with a signal that will be connected to a top-level port. This signal will be used to determine if there is any PROM attached to the controller. In case there is no PROM device attached the PROM/IO controller will issue an AMBA ERROR response to accesses to the PROM area. This will ensure that the processors go into error mode if the NGMP system lacks a boot PROM.

2.19.2 Configuration

The PROM/IO controller will be implemented with the following characteristics:

- 8- and 16-bit bus width, supporting BCH EDAC for the PROM area
- 256 MiB AMBA area for PROM
- 256 MiB AMBA area for memory mapped I/O

2.20 APB Bridge

2.20.1 Description

The APBCTRL core from GRLIB will be used as AMBA AHB/APB bridge, which is an APB bus master according the AMBA 2.0 specification [RD3]. Decoding (generation of PSEL) of APB slaves is done using the plug&play method explained in the GRLIB IP Library User's Manual [RD22]. A slave can occupy any binary aligned address space with a size of 256 bytes - 1 MiB. Writes to unassigned areas will be ignored, while reads from unassigned areas will return an arbitrary value. AHB error responses will never be generated. Please see the APB memory map in section 2.31 for a list of the slaves connected to the APBCTRL core.

2.20.2 Configuration

The core will be configured to support 26 slaves.

2.21 32-bit Programmable Timers

2.21.1 Description

The General Purpose Timer Unit (GPTIMER) will provide five global 32-bit decrementing timers with a 16-bit prescaler. The prescaler is clocked by the system clock and decremented on each clock cycle. When the prescaler underflows, it is reloaded from the prescaler reload register and a timer tick is generated. The operation of each timer is controlled through its control register. A timer is enabled by setting the enable bit in the control register. The timer value is then decremented on each prescaler tick. When a timer underflows, it will automatically be reloaded with the value of the corresponding timer reload register if the restart bit in the control register is set, otherwise it will stop at -1 and reset the enable bit. The timer unit acts as a slave on AMBA APB bus. The unit is capable of asserting interrupts on timer underflows.

By setting the chain bit in the control register timer n can be chained with preceding timer n-1. Decrementing timer n will start when timer n-1 underflows.

2.21.2 Configuration

The GPTIMER core will be implemented with the following characteristics:

- 16-bit pre-scaler
- Five 32-bit timers, of which one is intended for use as a watchdog
- Each timer, except the watchdog timer, will drive a separate interrupt line
- The watchdog will be disabled after chip reset if the external PROMPRES signal indicates that the system lacks PROM to allow uploading of system software via RMAP. Otherwise the watchdog will be enabled.
- The last timer will have watchdog functionality

2.22 Multiprocessor Interrupt Controller

2.22.1 Description

The IRQMP core from GRLIB is included to support SMP configurations and existing LEON3 BSPs. The AMBA system in GRLIB provides an interrupt scheme where interrupt lines are routed together with the AHB/APB bus signals, forming an interrupt bus. Interrupts from AHB and APB units are routed through the bus, combined together, and propagated back to all units. The multiprocessor interrupt controller core is attached to AMBA bus as an APB slave, and monitors the combined interrupt signals. Interrupt signals can also be routed to a secondary interrupt controller that is then attached to the IRQMP core via a normal interrupt signal.

The interrupts generated on the interrupt bus are all forwarded to the interrupt controller. The interrupt controller prioritizes, masks and propagates the interrupt with the highest priority to the processor. In multiprocessor systems, the interrupts are propagated to all processors. The interrupt assignments are shown in section 2.32.

The IRQMP core will include a register for setting the reset start address of each processor core. Processor cores can be powered down, and restarted, via the IRQMP APB register interface. During system boot only one processor is activated and performs system initialisation. The first processor may then bring up the other processors via the IRQMP register interface. In a system without PROM, an external entity may upload system

software, for instance via SpaceWire RMAP, and then enable the processors via the IRQMP register interface.

2.22.2 Configuration

The core will be instantiated with support for four processors.

2.23 Secondary Interrupt Controllers

2.23.1 Description

The system includes four secondary interrupt controllers. The secondary interrupt controllers have two uses; provide additional interrupt lines to the multiprocessor and dedicated (internal to the processors) interrupt controllers, and to provide space partitioning in ASMP configurations.

For a detailed description on the interrupt infrastructure, please see section 2.32.

2.23.2 Configuration

Each core will be configured to support 31 interrupt lines.

2.24 General Purpose I/O Port Controller

2.24.1 Description

The system will have a 16-bit GRGPIO core from GRLIB with interrupt support. Each bit in the general purpose input output port can be individually set to input or output, and lines 0 to 14 can optionally generate an interrupt. For interrupt generation, the input can be filtered for polarity and level/edge detection.

The I/O ports are implemented as bi-directional buffers with programmable output enable. The input from each buffer is synchronized by two flip-flops in series to remove potential meta-stability. The synchronized values can be read-out from the I/O port data register.

2.24.2 Configuration

The core will be implemented with the following characteristics:

- 16-bit I/O port
- The 15 first I/O lines will be able to generate interrupts on separate lines

2.25 8-bit UART

2.25.1 Description

The APBUART core from GRLIB provides an interface for serial communication that can be used to connect to a RS232 serial link [RD4]. The UART supports data frames with 8 data bits, one parity bit and one stop bit. To generate the bit-rate, the UART has a programmable 12-bit clock divider. Two eight byte FIFOs are used for data transfers between the APB bus and UART. Hardware flow-control is supported through the RTSN/CTSN handshake signals.

Two 8-bit UART cores will be instantiated in the NGMP system.

2.25.2 Configuration

The cores will be implemented with the following characteristics:

- Support for parity
- Support for flow control
- Eight byte receiver and transmitter FIFOs

2.26 General Purpose Register for Clock Gate Control

2.26.1 Description

The GRGPREG core from GRLIB will be included in order to provide control over which parts of the design that should be clock gated. The core implements a field of configurable length that can be written and read via the AMBA APB. The contents of the register is also propagated out of the core and its values will be used to clock gate parts of the design.

The reset value for the register bits utilised to clock gate the cores on the Debug AHB bus will be taken from the external DSUEN signal so that the cores will be gated off after reset if DSU operation is not enabled. The strategy for the rest of the system will be to disable as much as possible. After reset the parts of the system required for CPU0 to execute from PROM will be clocked. System software can then enable the rest of the system. If the external PROMSEL signal indicates that there is no PROM available, the RMAP targets in the SpaceWire cores will be clocked so that system software can be uploaded by an external entity.

2.26.2 Configuration

The number of register bits in the GRGPREG will be defined when the clock and power gating scheme is finalised. This will be done during the architectural design phase.

2.27 PCI Arbiter

2.27.1 Description

The system will include the ESA PCI arbiter with a GRLIB wrapper. The arbiter can be configured to support 4, 8, 16 or 32 agents. A priority level (0 = high, 1 = low) is assigned to each device, with the exception of the highest numbered agent which has always lowest priority.

2.27.2 Configuration

The PCI arbiter will be implemented with the following characteristics:

- Support for four PCI agents
- Programmable priority levels via AMBA APB interface.

2.28 AHB Status Registers

2.28.1 Description

The AHBSTAT core provides status registers that stores information about AMBA AHB accesses triggering an error response. There is a status register and a failing address register

capturing the control and address signal values of a failing AMBA bus transaction, or the occurrence of a correctable error being signaled from a fault-tolerant core.

The core monitors AMBA AHB transactions and store the current HADDR, HWRITE, HMASTER and HSIZE values internally. Monitoring is always active after startup and reset until an AMBA ERROR response is detected. When an error is detected, the status and address register contents are frozen and the New Error (NE) bit is set to one. At the same time an interrupt is generated. Note that many of the fault-tolerant units containing EDAC signal an uncorrectable error as an AMBA error response, so that it can be detected by the processor as described above.

Not only error responses on the AHB bus can be detected. Many of the fault-tolerant units containing EDAC have a correctable error signal that is asserted each time a single error is detected. When such an error is detected, the effect will be the same as for an AHB error response. The only difference is that the Correctable Error (CE) bit in the status register is set to one when a single error is detected. When the CE bit is set the interrupt routine can acquire the address containing the single error from the failing address register and correct it. When it is finished it should reset the CE bit and the monitoring becomes active again.

The system has two AHBSTAT cores. One AHBSTAT core monitors the Processor bus and may have error correction signals from the L2 cache and PROM memory controller connected to it. The second AHBSTAT core monitors the Slave I/O bus and has a error correction signal connected from the PROM/IO controller. The APB bus interface is used for accessing status and failing address registers.

Note that AHBSTAT functionality is also available on the Memory bus where it is built into the memory scrubber core.

2.28.2 Configuration

The core listening to the Processor bus may be implemented with support for two error correcting slaves; the L2 cache and the PROM/IO Controller. The core listening on the Slave I/O bus will have support for one error correcting slave, the PROM/IO controller.

2.29 IRQSTAMP

2.29.1 Description

The IRQSTAMP core will be connected to all the interrupt lines in the NGMP system and can be configured to time stamp up to four of the interrupt lines. The core has the following features:

- One 32-bit up-counter accessible via the core's register interface
- One mask register to select up to four interrupt lines that should receive time stamps
- Four registers to latch the timer value when a selected interrupt line is asserted
- One configuration register to select the action when multiple interrupts occur on the same line before previous interrupt(s) have been served. When this happens the core can perform one of the following actions:
  - Keep the tag for when the first interrupt occurred
  - Update the tag for each assertion of the interrupt line

2.29.2 Configuration

The core will be instantiated implementing the following features:

- Support for stamping up to four selected interrupt lines. Selection of lines is configurable by software.

2.30 AHBCTRL

2.30.1 Description

The AHBCTRL AMBA AHB controller core from GRLIB is a combined AHB arbiter, bus multiplexer and slave decoder according to the AMBA 2.0 standard.

All AMBA AHB buses in the NGMP system will be controlled by AHBCTRL cores configured to use round-robin arbitration. In round-robin arbitration, priority is rotated one step after each AHB transfer. The maximum burst length of eight 32-bit words guarantees that no master will be starved.

If no master requests the bus, the last owner will be granted (bus parking). On a fully loaded system each of the processors, and the masters behind the IOMMU, connecting the Master I/O bus to the Processor bus, can receive up to 20% of the bandwidth of the Processor bus. If the bus is not fully loaded, the active processors or I/O cores may receive more than 20% of the available bus time.

2.30.2 Configuration

The core will be instantiated implementing the following features:

- Full decoding of GRLIB AMBA Plug'n'Play records
- Round-robin arbitration
- Support for AMBA SPLIT responses

2.31 Memory Map

Table 1 shows the memory map for the NGMP system. Since the AHB/AHB bridge connecting the debug bus to the processor bus is uni-directional the processors will not be able to see the DSU memory areas (0xF0000000 – 0xF37FFFFF) nor the I/O and configuration area for Debug AHB bus (0xFFE00000 – 0xFFEFFFFF).

The L2 Cache memory area (0x00000000 – 0x7FFFFFFF) covers the DDR2 SDRAM and, if implemented, the on-chip SDRAM. This means that a potential on-chip SDRAM will shadow the highest addresses of the DDR2 RAM. If a system has memory devices giving 2 GiB of DDR2 memory, the top 32 MiB (required size of on-chip RAM) of DDR2 memory will not be accessible since this memory will be hidden by the on-chip SDRAM area.

Address range	Description
0x00000000 - 0x7FFFFFFF	L2 Cache memory area
0x80000000 - 0xBFFFFFFF	GRPCI PCI memory area
0xC0000000 - 0xCFFFFFFF	PROM area of PROM/IO Controller
0xD0000000 - 0xDFFFFFFF	I/O area of PROM/IO Controller
0xF0000000 - 0xF07FFFFF	DSU area for processor 0
0xF1000000 - 0xF17FFFFF	DSU area for processor 1
0xF2000000 - 0xF27FFFFF	DSU area for processor 2
0xF3000000 - 0xF37FFFFF	DSU area for processor 3
0xF8000000 - 0xF80FFFFF	AHB/APB bridge (see table below for devices mapped on the bridge)
0xFFC00000 - 0xFFCFFFFF	I/O and configuration area for Slave I/O AHB bus
0xFFD00000 - 0xFFFDFFFF	I/O and configuration area for Memory AHB bus
0xFFE00000 - 0xFFEFFFFF	I/O and configuration area for Debug AHB bus
0xFFF00000 - 0xFFFFFFFF	I/O and configuration area for Processor AHB bus

Table 1: NGMP AHB memory map

Access to addresses outside the ranges described in the table above will return an AHB error response. The detailed register layout is defined, or will be defined, in the manual for each IP core. The control registers of most on-chip peripherals are accessible via the AHB/APB bridge located on the Slave I/O bus. The memory map of the APB bus is shown in table 2. The base addresses of the on-chip peripherals have been placed on 4K boundaries. This is to allow mapping one, and only one, peripheral within a virtual MMU page. This is necessary to allow full separation between processes.

Address range	Description
0xF8000000 - 0xF80000FF	APBUART 1
0xF8001000 - 0xF80010FF	APBUART 2
0xF8002000 - 0xF80020FF	GPTIMER
0xF8003000 - 0xF80030FF	IRQMP
0xF8004000 - 0xF80040FF	IRQCTRL2 1
0xF8005000 - 0xF80050FF	IRQCTRL2 2
0xF8006000 - 0xF80060FF	IRQCTRL2 3
0xF8007000 - 0xF80070FF	IRQCTRL2 4
0xF8008000 - 0xF80080FF	GRGPREG (clock gating)
0xF8009000 - 0xF80090FF	GPIO
0xF800A000 - 0xF800A0FF	GRETH 1

Address range	Description
0xF800B000 - 0xF800B0FF	GRETH 2
0xF800C000 - 0xF800C0FF	GRSPW2 1
0xF800D000 - 0xF800D0FF	GRSPW2 2
0xF800E000 - 0xF800E0FF	GRSPW2 3
0xF800F000 - 0xF800F0FF	GRSPW2 4
0xF8010000 - 0xF80100FF	GRPCI
0xF8011000 - 0xF80110FF	PCIDMA
0xF8012000 - 0xF80120FF	PCI arbiter
0xF8013000 - 0xF80130FF	HSSL 1
0xF8014000 - 0xF80140FF	HSSL 2
0xF8015000 - 0xF80150FF	HSSL 3
0xF8016000 - 0xF80160FF	HSSL 4
0xF8017000 - 0xF80170FF	AHBSTAT (listening to Processor bus)
0xF8018000 - 0xF80180FF	AHBSTAT (listening to Slave I/O bus)
0xF8019000 - 0xF80190FF	IRQSTAMP

Table 2: APB memory map

2.32 Interrupt Assignments and Infrastructure

2.32.1 Overview

The NGMP interrupt infrastructure includes the multiprocessor interrupt controller that is traditionally used in LEON SMP systems. In order to efficiently support ASMP configurations each processor will be extended with a dedicated interrupt controller located within the processor core. Some interrupts, most notably the interrupts from the general purpose timer unit, are connected directly to the interrupt controllers in the processor cores and to the multiprocessor interrupt controller. Other interrupt sources in the system are routed via four secondary interrupt controllers, which are in turn directly connected to each processor and also to the multiprocessor interrupt controller.

The multiprocessor interrupt controller (IRQMP) is instantiated in the system to support standard SMP configurations. The multiprocessor interrupt controller has the same interrupt lines connected to it as the dedicated (internal) processor interrupt controllers, with the exception of the internal processor timer units that are not connected to the shared multiprocessor interrupt controller. Illustration 3 depicts the interrupt infrastructure. The illustration shows IRQMP connected to all processors using the interrupt controller interface of the LEON4FT. The secondary interrupt controllers have a single line connecting each secondary controller both to the dedicated interrupt controller in each processor core and to the IRQMP unit. Note that a processor core will be configured by software, by writing to the processor's internal register space, to use either the internal interrupt controller or the external multiprocessor interrupt controller. Subsections 2.32.2 and 2.32.3 describe the parts of the interrupt infrastructure used in SMP and ASMP configurations.

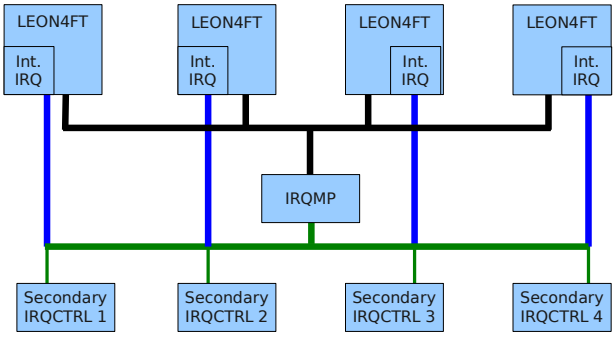

Illustration 3: Interrupt infrastructure allowing both SMP and ASMP operation

2.32.2 SMP Configuration

During normal SMP operation the (external to the processor cores) multiprocessor interrupt controller will be used. Most of the peripheral interrupts are connected to the secondary interrupt controllers and software configures the IRQMP core to use interrupts from one or more of the secondary interrupt controllers. In this configuration, the dedicated (internal) interrupt controllers are not used and the processors do not listen to the lines directly connecting them to the secondary interrupt controllers. When a processor is using the external multiprocessor interrupt controller it will not be able to receive interrupts via the internal interrupt controller, there will be a multiplexer in each processor that selects if the interrupt request lines should come from the external or the internal controller. Illustration 4 shows a diagram of the interrupt infrastructure used when running all processor cores in SMP configuration. Note that the use of more than one secondary interrupt controller is optional.

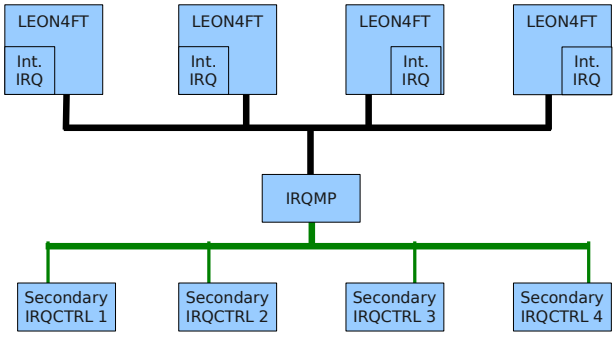

Illustration 4: Interrupt infrastructure, all cores in SMP configuration

2.32.3 ASMP Configuration

In ASMP operation a processor core's internal interrupt controller should be used instead of the shared IRQMP core. The operation and the layering of interrupts (with most peripheral interrupts coming via a secondary interrupt controller) will be the same in ASMP as in SMP operation, the difference being that instead of accessing the shared IRQMP over the AMBA bus, the processor core's dedicated interrupt controller is accessed via internal register accesses. Illustration 5 shows a diagram of the interrupt infrastructure used when running all cores in ASMP configurations.

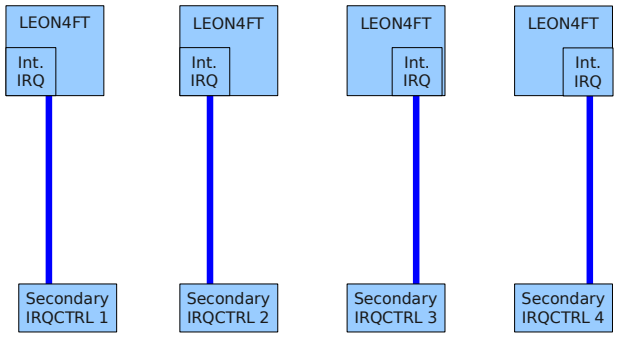

Illustration 5: Interrupt infrastructure, all cores in ASMP configuration

2.32.4 Mixed Configurations

The specified interrupt infrastructure also allows users to use mixed configurations with one SMP operating system (occupying two or three processor cores) and one or two ASMP operating systems (one running on each of the remaining processor cores). The specified interrupt infrastructure will not allow straight forward operation with two SMP configurations in parallel, with each OS using two processor cores, since this would require two external multiprocessor interrupt controllers.

There may be ASMP applications where one operating system wishes to synchronise with another operating system by asserting an interrupt. In this case, OS A wishing to alert OS B can write to the force register of the secondary interrupt controller that is used by OS B. In addition to this, the multiprocessor interrupt controller has unassigned interrupts that can be used for synchronisation between processes.

The specified solution does not prevent malfunctioning software that is not guarded by a processor MMU, or that has corrupted the MMU page tables, from disturbing the operation of another OS by writing the secondary interrupt controllers. Software that is not restricted by a processor's MMU will always have the permissions necessary to disrupt operation of the NGMP.

2.32.5 Interrupt Assignments

Table 3 shows the interrupt assignments for the interrupt controllers included in the LEON4 processor cores. The first column shows the interrupt line used on a specific core. The second column shows the interrupt line as seen by high-level software. Table 4 shows the interrupt assignments on the (external) multiprocessor interrupt controller (IRQMP). Interrupt assignments for all secondary interrupt controllers, attached to interrupt lines 1 - 4 on each processor and also on the multiprocessor interrupt controller, are shown in table 5.

Processor line	Virtual line	Description
0	0	DSU - Debug support unit
1	1	IRQCTRL2 - Secondary interrupt controller 1
2	2	IRQCTRL2 - Secondary interrupt controller 2
3	3	IRQCTRL2 - Secondary interrupt controller 3
4	4	IRQCTRL2 - Secondary interrupt controller 4
5	5	Timer 1 from local processor timer unit
6	6	Timer 2 from local processor timer unit
7	7	Timer 3 from local processor timer unit
8	8	Timer 4 from local processor timer unit
9	9	GPTIMER – General purpose timer, timer 1
10	10	GPTIMER – General purpose timer, timer 2
11	11	GPTIMER – General purpose timer, timer 3
12	12	GPTIMER – General purpose timer, timer 4
13	13	Unassigned
14	14	Unassigned
15	15	Unassigned, non-maskable interrupt

Table 3: Interrupts on LEON4 internal interrupt controller(s)

Processor line	Virtual line	Description
0	0	DSU - Debug support unit
1	1	IRQCTRL2 - Secondary interrupt controller 1
2	2	IRQCTRL2 - Secondary interrupt controller 2
3	3	IRQCTRL2 - Secondary interrupt controller 3
4	4	IRQCTRL2 - Secondary interrupt controller 4
5	5	Unassigned – Can be used for sync. between cores
6	6	Unassigned – Can be used for sync. between cores
7	7	Unassigned – Can be used for sync. between cores
8	8	Unassigned – Can be used for sync. between cores
9	9	GPTIMER – General purpose timer, timer 1
10	10	GPTIMER – General purpose timer, timer 2
11	11	GPTIMER – General purpose timer, timer 3
12	12	GPTIMER – General purpose timer, timer 4
13	13	Unassigned, reserved for SMP synchronisation
14	14	Unassigned, reserved for SMP synchronisation
15	15	Unassigned, non-maskable interrupt

Table 4: Interrupts on (external) multiprocessor interrupt controller

Extended interrupt controller interrupt line	Virtual line	Description
0	16	GRGPIO line 0
1	17	GRGPIO line 1
2	18	GRGPIO line 2
3	19	GRGPIO line 3
4	20	GRGPIO line 4
5	21	GRGPIO line 5
6	22	GRGPIO line 6
7	23	GRGPIO line 7
8	24	GRGPIO line 8
9	25	GRGPIO line 9
10	26	GRGPIO line 10
11	27	GRGPIO line 11
12	28	GRGPIO line 12
13	29	GRGPIO line 13
14	30	GRGPIO line 14
15	31	GRSPW2 1
16	32	GRSPW2 2
17	33	GRSPW2 3
18	34	GRSPW2 4
19	35	HSSL 1
20	36	HSSL 2
21	37	HSSL 3
22	38	HSSL 4
23	39	GRETH 1
24	40	GRETH 2
25	41	GRPCI
26	42	AHBSTAT (shared)
27	43	Memory scrubber
28	44	L2 cache
29	45	APBUART 1
30	46	APBUART 2
31	47	IOMMU

Table 5: Interrupt mapping on secondary interrupt controllers

2.33 Error Handling

2.33.1 Overview

The following subsections describe how the architecture handles different types of errors. The errors discussed in this section that can be detected and/or corrected are:

- Access violations and errors reported via AMBA
- Upsets in registers and buffers

Handling of errors such as checksum errors in incoming data packets are not discussed in this document.

2.33.2 Access Violations and Errors Reported via AMBA

Access violations in the AMBA bus and errors detected by AMBA slaves are signalled in the form of AMBA ERROR responses. AMBA ERROR responses may be issued under the following conditions:

- Cores on the Master I/O bus will receive an AMBA ERROR response from the IOMMU on access violations
- The memory protection unit in the L2 cache will issue an AMBA ERROR response when an access is made to a protected area
- The memory controller will issue an AMBA ERROR response when an uncorrectable error is detected. This error will be propagated by the L2 cache.

When the IOMMU issues an AMBA ERROR response to a unit on the Master I/O bus, as a result of an access violation, the IOMMU will record the offending master and the address of the accesses. The IOMMU can be configured to issue an interrupt to the processor when this condition arises. The I/O core that initiated the access will be stopped and appropriate status bits in the core's status register will be set to allow software to detect the error.

The L2 cache may issue an ERROR response as a result of receiving an AMBA ERROR response from the DDR2 controller or as a result of an access violation in the memory protection unit. Both sources indicate an unrecoverable error. Since write operations in the L2 cache are posted writes, errors can also be signalled by the L2 cache asserting an interrupt. The action taken upon reception of this interrupt should be the same as the actions taken on an AMBA ERROR response.

AMBA ERROR responses will lead to different traps depending on the operation performed by the processor. Data loads and stores will result in trap 0x09 (data_access_exception) or trap 0x2B (data_store_error). An AMBA ERROR response on instruction fetch will result in trap 0x01 (instruction_access_exception). Since data_store_errors may be delayed due to the LEON4FT write-buffer, it is recommended that data_store_errors are masked and that error handling is done by reacting to the interrupt that will be asserted by the AHB status register, described in section 2.28, when it detects an AMBA ERROR response.

When the processor receives an AMBA ERROR response, or an interrupt, as a result of an uncorrectable error, the error should be viewed as unrecoverable requiring restart of a task, the entire system, or as a permanent error that will require permanent shutdown of the system.

2.33.3 Errors in External Memory

Correctable errors in external memory are signalled by the DDR2 memory controller core, which will include a side-band AMBA signal that is asserted on accesses where data has been corrected. This signal is connected to the memory scrubber that will keep statistics of the amount of correctable errors. If the amount of correctable errors suddenly increases, it is recommended that software increases the rate at which memory is scrubbed so that the amount of errors do not build up.

Uncorrectable errors will result in an AMBA ERROR response that will be handled as described in section 2.33.2.

2.33.4 Errors in LEON4FT Register File, Caches and MMU

The register files of the LEON4FT processors will be implemented with SEU error protection in the form of BCH checksums. The BCH checksum detects two bit errors and corrects one bit error per word. The SEU error detection has no impact on behaviour, but in case of a correctable error a correction cycle will delay the current instruction with six clock cycles. An uncorrectable error in the IU register file will cause trap 0x20 (register_access_error).

Each word in the tag and data cache memories is protected by four check bits. An error during cache access will cause a cache line flush, and re-execution of the failing instruction. This will ensure that the complete cache line (tags and data) is refilled from external memory.

An error in the MMU registers will cause a page fault. Software is then able to deduct if the page fault is a fault that should be handled or if the page fault is an unexpected error that should be treated as an unrecoverable error.

2.33.5 Errors in Internal Buffers of IP cores

Internal buffers, in cores other than the LEON4FT processor, will be guarded with error correcting codes, error detection and sparing, or be implemented using registers. When the buffers are implemented with registers, no additional error detection is needed. It is assumed that the radiation hardened, or TMR, registers provide adequate protection.

When the core buffers are implemented with RAM cells it is assumed that the error correcting codes, or the sparing, is sufficient to correct any errors. Some cores, such as the GRETH_GBIT Ethernet core, perform checksum calculations when reading data from the buffer, and will thus provide additional error checking.

All cores that instantiate RAM blocks provide means to perform diagnostic accesses to these RAM blocks. This functionality can be used for static RAM radiation testing.

3 NGMP ASIC IMPLEMENTATION

3.1 Overview

The target technology for the NGMP is the Space-DSM technology that is under development in a separate ESA activity. At the time of writing, the specification of the targeted Space-DSM technology is very limited.

3.2 Operating Frequency and Clock Domains

The target frequency of the LEON4FT and on-chip buses is 400 MHz, but depends ultimately on the implementation technology. The minimum frequency is determined by the external memory device. For DDR2 SDRAM the minimum system frequency is 62.5 MHz when clocking the DDR2 interface with twice the system frequency, attaining the minimum DDR2 frequency of 125 MHz. If SDRAM is used then the minimum frequency will be limited by the memory bandwidth that must be attained.

The system has nine clock domains, as listed in the table below:

Clock domain name	Description
SYS	System main clock domain
MEM	Memory interface clock domain
JTAG	JTAG clock domain. Encompasses part of JTAG Debug Link core.
USB	USB clock domain. Encompasses part of USB Debug Link core.
ETH0RX, ETH1RX	Ethernet receiver clock domains. Encompasses part of Ethernet core.
ETH0TX, ETH1TX	Ethernet transmitter clock domains. Encompasses part of Ethernet core.
PCI	PCI clock domain. Encompasses part of PCI core.
SPW	SpaceWire clock domain. Encompasses part of each SpaceWire core, the SpaceWire receiver and transmitter in each core use a common clock.

The SYS and MEM clock domains have their clocks generated from the same source. When using DDR2 SDRAM the SYS clock domain can be clocked at a frequency that is 1x or 1/2x of the MEM clock domain frequency. When using SDR SDRAM the SYS clock domain can be clocked at 1x or 4x of the MEM clock domain frequency. Assuming that the source for generating the system and memory clocks is a 100 MHz clock, the clock generator will need to generate clocks that are 1x, 2x and 4x the input clock. This will cover the following cases:

- Memory is DDR2 SDRAM and SYS and MEM on same frequency. MEM and SYS will both be clocked by the clock generator's 4x output.
- Memory is DDR2 SDRAM and SYS is run at half the frequency of MEM. MEM will be clocked by the clock generator's 4x output and SYS by the generator's 2x output.
- Memory is SDR SDRAM and SYS is run at 4x the MEM clock frequency. MEM will be clocked by the clock generator's 1x output and SYS will be clocked by the generator's 4x output.
- Memory is SDR SDRAM and SYS is run at the same frequency as MEM. Both domains will be clocked by the clock generator's 2x output.

The remaining clock domains are all driven by external clocks. The sources are listed in the Feasibility Report Spreadsheet [AD7].

3.3 Power-up and Initialization State

Power-up and initialisation state has dependencies on the implementation technology. This section describes the functional state at power up and after system reset.

During power-up, and during system reset, all parts of the design will be active and clocked. When system reset is de-asserted the register controlling clock gating in the design will keep all main cores clocked while the bits controlling clock gating for the cores on the Debug AHB bus will receive its reset value from the external DSU enable signal (DSU_EN). If the DSU is enabled, all debug links will be clocked after system reset, if the DSU is disabled the DSU, Debug bus infrastructure and the dedicated debug communication links will be gated off.

There are also other external signals that define the system state after reset:

- Since the SDRAM and DDR2 controllers are expected to share pins the external signal MEM_IFSEL will be used to select which controller that will be connected to the memory interface pads.
- The memory interfaces can be clocked at a multiple of the clock frequency. The external signal MEM_IFFREQ selects if the SDRAM memory device should be clocked at 1x or ¼x the system frequency. The same signal selects if the DDR2 memory device should be clocked at 1x or 2x the system frequency. The factors used here are only indicative, the final clock scaling factors will be determined when more is known about the target technology and attainable maximum system frequency.
- The presence of external PROM is signalled to the PROM/IO Controller via en external signal (PROM_PRES). If the design lacks external PROM the first access made by the activated processor will receive an AMBA ERROR response.
- An external vector (SPW_CLKDIV10[7:0]) determines the initial SpaceWire clock divisor value.
- Four external signals (SPW_RMAPEN[0:3]) determines if the RMAP target in SpaceWire core 0 – 3 should be enabled.
- The external signal SPWD_RMAPEN determines if the dedicated RMAP target on the debug bus should be enabled.
- An external vector (ETH_EDCLBSEL[0:1]) controls to which bus (Master I/O or Debug) an Ethernet controller's debug link AHB master interface is attached to. If the Debug bus is gated off, the EDCL master interfaces are always connected to the Master I/O bus. If the ETH_EDCLBSEL signal indicates that en Ethernet core is to be connected to the Debug bus and the external DSU_EN signal is deasserted, the EDCL will be disabled.
- The eight first general purpose IO pins (GPIO[7:0]) are used to set least significact nibbles of the EDCL IP address, as described in section 2.14.1.
- If the external DSU break (DSU_BRE) signal is asserted, and the DSU functionality is enabled via the DSU enable signal, the processors will go into debug mode after reset. Otherwise the first processor will start executing from PROM.

If the DSU is disabled, or if the DSU break signal is de-asserted, the first processor will start executing from the first address in PROM. Depending on software the first processor may

then initialise the memory scrubber to wash main memory, perform other system initialisation and ultimately bring the other processors on-line.

In the case where the processors do not enter debug mode and the first processor receives an AMBA ERROR response due to the design lacking external PROM the processor will go into error mode. Following this software can be uploaded into main memory via, for instance, RMAP and the processors started via the multiprocessor interrupt controller, as described in section 2.22.1.

3.4 Reset and Power Cycling

Depends ultimately on the implementation technology. The design will use synchronous reset as baseline. Interfaces such as the PCI interface and tri-state buses have asynchronous resets to ensure correct behaviour during power-up.

3.5 Electrical Constraints

Depends ultimately on the implementation technology.

3.6 Thermal Environment Constraints

Depends ultimately on the implementation technology.

3.7 Radiation Environment Constraints

The NGMP will be fault-tolerant against SEU and SET effects tolerating a maximum rate of 10^{-5} uncorrectable errors per NGMP device per day corresponding to a MTBF of 264 years in geostationary orbit (conditions: solar minimum, Z=1 Al shielding of 1g/cm^2).

Total dose (TD) and (SEL) effects are assumed to be mitigated by the DSM technology; no actions have been made during the definition of the NGMP system to mitigate these effects.

3.8 Power Budget and Dissipation

The baseline requirements are that the power consumption of the NGMP ASIC core (without I/Os) under worst case operating conditions and maximum software load shall not exceed 6W. Maximum power consumptions in idle mode (no software activity, but conservation of status and SEE protection) shall not exceed 100 mW.

Without detailed information about target technology characteristics, it is difficult to estimate the power consumption of the NGMP device. Power consumption will be minimized by using both dynamic and static clock-gating as far as possible. Static clock-gating will be implemented in the same way as in the UT699 LEON3FT device, where each on-chip core can be clock gated under software control, and where the processor will clock-gate itself whenever it is in idle mode.

Dynamic clock-gating will be implemented using the synthesis tool's (power compiler) capabilities, if available. The tool can detect 'enabling' nodes for groups of flip-flops in the netlist, and automatically insert clock-gates controlled by the operation of the logic. If the flip-flops will not change value during a certain logic state, the clock will be gated to reduce the power consumption in the clock tree. This also has the benefit of reducing the likelihood of latching SEU errors in combinational logic.

3.9 Physical and Mechanical Constraints

Depends ultimately on the implementation technology.

3.10 Timing

Depends ultimately on the implementation technology. Measures have been taken during the definition phase to reduce AMBA bus loads which have been shown to house the critical path in previous LEON/GRLIB SoC designs.

4 FAULT-TOLERANCE

The fault-tolerance in the NGMP system is aimed at detecting and correcting SEU errors in on-chip and off-chip RAM. The L1 cache and register files in the LEON4FT cores are protected using parity and BCH coding, while the L2 cache will use BCH. External DDR2 memory will be protected using Reed-Solomon coding, while the boot PROM will use BCH. Any RAM blocks in on-chip IP cores will be protected with BCH or TMR. The protection capabilities of different cores has been further discussed in section 2.33.

If the target technology has SEU hardened flip-flops, then they will be the preferred choice for all on-chip registers. If no hardened flip-flops are available, TMR will be used through out the whole design. The TMR cells will be added at netlist level and not in the RTL code. No logic to detect or correct SEU errors in combinational logic is foreseen to be implemented. However, low-level clock-gating as discussed 3.8 will provide an excellent reduction of effective combinational SEU errors, as any register being clock-gated off cannot latch in a combinational error. Unfortunately, clock-gating can lead to error build-up in TMR registers if some part is powered-down for an extended period of time. This effect will need to be analysed in detail if TMR will be used with low-level clock gating.

Please see the Architecture Exploration Report [AD4] and SEE Mitigation Plan [AD5] for further information about fault-tolerance and measures taken to increase radiation tolerance.

5 DESIGN VERIFICATION

5.1 Test Modes

5.1.1 BIST

Built In Self Test (BIST) of on-chip RAM blocks that will perform tests on all RAM blocks of the system. The BIST can either be included in the VHDL implementation of the system or can be added during the backend design when RAM blocks are being placed.

5.1.2 Memory Test Functionality in Memory Scrubber

The NGMP system may include support for external memory testing. The implementation is briefly discussed in section 2.7.1.

5.1.3 On-line Device tests

On-line device tests can be used to verify the functionality of the following devices without additional test equipment.

LEON4 and FPU

The straightforward solution for testing the functionality of the processor cores and FPU is to execute a software test that stresses the different parts of a LEON4 processor such as the ALU, multiplier, register file, interrupt controller, timers, and FPU. The result of such a test could be monitored by one of the other LEON4 cores or through one of the debug interfaces.

PCI

The PCI interface can be tested by performing accesses on the PCI bus that is addressed to the systems own PCI interface. This way communication initialized by the PCI master will be answered by the PCI target and the data that is read or written can be verified that it has not been altered. The PCI standard does not account for the operation when a device accesses its own target. The PCI core will arbitrate for the bus, perform the cycle but not read the reply from the pads and instead use internal loop back using multiplexers.

Ethernet

All Ethernet PHYs are specified to perform loop-back communication, which can be used to test the functionality of the Ethernet interface. This would be done by configuring the PHY to be in loop-back mode and then send a Ethernet frame that would be received by the system and verified if it is the same data as sent.

High-Speed Serial Link

How to test the HSSL will have to be assessed once more information is available about the IP to be used.

UART

The UART has built in support for loop-back that can be used to verify that data that is sent is also received correctly.

5.2 Fault Coverage Required at Production Test

Production tests will be based on the scan chain methodology where all flip-flops in the NGMP design will be connected to a scan chain. The minimum fault coverage will be 98%.

6 REUSABILITY

6.1 Overview

The HDL code will be VHDL-RTL, being readable, well structured and commented, following adequate coding rules. All design variants or scalability will be selectable by the means of configuration constants or generics. All cores have licensing constraints described in the activity's Development plan [AD6]. Some cores will only be delivered in netlist format.

Details of the architecture will be documented in executable and readable form in the HDL code contained within the Architectural Design Database. Additional information will be provided within the Architecture Definition Report.

6.2 Portability

All developed or reused IP cores will implement the technology independence scheme used in LEON4/GRLIB. This means that the complete model will be fully portable between FPGA and ASIC, and between different foundry technologies. During the whole development, emphasis will be put on low complexity and low power consumption. All code will be compliant with VHDL'93, and fully synthesisable with Synplify and Synopsys DC.

The robust design style used in GRLIB will be maintained, which means that all signals will be clocked on the rising edge, and transparent latches or asynchronous feed-back will not be used. All inputs will be registered when possible, to improve external timing. For modules where signals pass between two separate clock domains, the necessary precautions will be taken to minimize the risk for metastability problems. This includes re-sampling of signals and the use of dedicated control-flow signals.

7 OPEN ITEMS

The table below lists items that are to be considered or determined during a later phase of the activity.

#	Description	Justification	Applies to section
1	Baseline is to use LRU as the replacement scheme for the L1 cache. The replacement scheme could be made soft configurable with the options to use LRU or random replacement.	Has been left open since no obvious advantage of making the scheme soft configurable has been found.	2.2.1.7
2	Baseline is to use DDR2 as the primary memory interface. However the selection between DDR2 and DDR1 should be regarded as open.	The flight models of the NGMP are scheduled 4.5 years into the future. At that time there may be additional information available regarding device availability. Availability of I/O standard may also impact the final decision. The instantiation of the DDR2 controller can easily be replaced an instantiation of a DDR(1) controller.	2.4
3	SDRAM could potentially be made soft configurable between 32 and 64 data bits (plus check bits).	It could be beneficial to be able to create NGMP systems with a reduced width of the memory interface to support packages with low pin count. This is not considered technically feasible for DDR/DDR2.	2.5
4	Increase the number of timers in the general purpose timer unit.	If users chain the timers it may be necessary to add extra 32-bit timers in order to have enough timers that can be used.	2.21
5	Definition of IOMMU interface and implementation details postponed.	The interface and implementation details of the IOMMU will be determined during the architectural design phase due to the need to perform prototyping.	2.13
6	The L2 cache could be extended so that upon an L2 cache miss, the master is SPLIT. Another CPU can then gain access the bus and – in case of cache hit – be served immediately.	This is an extension that is planned for the L2 cache. However it may not be possible to implement it within the scope of the NGMP activity. There are also cases where this functionality will degrade performance.	2.3
7	Data watchpoints are subject to trade-off.	Since data watchpoints will be implemented in the processor pipeline they may affect parameters such as maximum system frequency. If the implementation of data watchpoints lead to a degradation of system performance they will be dropped.	2.9.1.5
8	LEON4FT register file protection	The LEON4FT core has several configuration options for the protection of the register file. The final selection of error correction mechanism will be made when more information about the target technology is available.	2.33.4
9	On-chip SDRAM may not be	On-chip SDRAM will only be included if the	2.6

#	Description	Justification	Applies to section
	available	target technology can provide the necessary cells.	
10	Number of processor cores may be decreased	If the design becomes severely area constrained the number of processors may be decreased to two.	2.2
11	USB debug link may be dropped	The USB debug link is a complex core and is subject to trade-off.	2.11
12	Propagate master index through bridge connecting master I/O bus	This would allow to tag the trace buffer with the indexes of masters on the master I/O bus. The current design will tag all accesses from the master I/O bus with the AHB master index of the IOMMU bridge. This has been left as an open item since even without this functionality it should be possible to deduct the master responsible for an access by looking at the address(es) accessed and the new functionality would require an one-off modification to the trace buffer and IOMMU. The possibility to propagate the master I/O index to the performance counters in order to perform measurements on a selected master behind the IOMMU will be considered, even if the trace buffer is not extended.	2.9
13	A clock frequency factor of 1, 2, 4 between the Processor bus and other buses may be introduced.	If dynamic clock gating is deemed to give insufficient savings in power or if all parts of the design does not meet the target frequency, the design can be modified to allow using different frequencies between the Processor bus and the rest of the buses in the system. This is discussed in [AD4] section 3.2.11.4.	3.2
14	Allow spare column of external memory.	To further improve resilience against permanent memory errors, a scheme with a spare device column could be envisaged. If a permanent error occurred in one memory device, the spare column would replace the faulty one. This could potentially cause a timing problem since the Reed-Solomon codes are slow to generate, and the multiplexer tree would follow the codec directly. It is therefore proposed to defer the decision on using column sparing until the final technology and package has been selected. Section 3.5.3 of [AD4] contains a more elaborate description.	2.4, 2.5
15	L2 cache may allow to lock more than one way.	The specification states that one way can be locked in the L2 cache. It may be convenient to be able to lock more than one way. This possibility will be explored during the architectural design phase.	2.3

8 REQUIREMENTS MATRICES

8.1 Statement of Work Key Requirements

The table below lists the key requirements specified under *A.2.1 Overall Requirements* in the activity's Statement of Work [AD1].

Requirement	Text	NGMP-SPEC-0001	Comment
NGMP-01	SPARC V8(E) based multi-core architecture	2.2	
NGMP-02	An average performance of 400 MOPS over the benchmarks executed, with a minimum of 200 MOPS on any single benchmark	-	Performance of prototype system is available in the Architecture Exploration Report [AD4].
NGMP-03	SPARC compliant Memory Management Unit (MMU)	2.2.1.6	
NGMP-04	Improved debug support with respect to LEON2FT	2.9	
NGMP-05	Large on-chip memory banks (>= 32 MB), extended caches	2.6, 2.3	On-chip memory will be included if available on target technology.
NGMP-06	Standardised interfaces (e.g. HSSL, SpW, MIL-STD-1553, CAN, RSS422)	2.14, 2.16, 2.17, 2.24, 2.25	
NGMP-07	Efficient interface for scalable multiprocessor architectures, co-processors and/or companion devices for I/Os and memory interface	2.4, 2.17, 2.16	
NGMP-08	Power consumption of the NGMP ASIC core (without IOs) under worst case operating conditions and maximum software load shall not exceed 6W. Maximum power consumption in idle mode (no SW activity, but conservation of status and SEE protection) shall not exceed 100 mW.	3.8	Depends on target technology

8.2 Compliance with ECSS-Q-60-02A

The table below lists the items specified to be included in a ASIC and FPGA requirements specification by ECSS-Q-60-02A [AD3].

Requirement	Text	NGMP-SPEC-0003	Comment
ECSS-Q-60-02A-a	Overall system partitioning, system configuration and operating modes	2.1	
ECSS-Q-60-02A-b	Interfaces of the ASIC and FPGA to the system and communication protocols to external devices, including memory mapping	2.1, 2.4, 2.10, 2.11, 2.12, 2.14, 2.15, 2.16, 2.17, 2.19, 2.24, 2.25, 2.27, 2.31	
ECSS-Q-60-02A-c	Operating frequency range	3.2	
ECSS-Q-60-02A-d	Electrical constraints, (e.g. voltage and current supply, drive capabilities and external load)	3.5	Depends ultimately on target technology
ECSS-Q-60-02A-e	Functional requirements	2	
ECSS-Q-60-02A-f	Applicable algorithms	2	
ECSS-Q-60-02A-g	Power-up and initialization state	3.3	
ECSS-Q-60-02A-h	Reset and power cycling requirements	3.4	Depends ultimately on target technology
ECSS-Q-60-02A-i	Error handling	2, 2.33	
ECSS-Q-60-02A-j	Test modes: system and device tests, on ground and in flight	5.1	
ECSS-Q-60-02A-k	Fault coverage required at production test	5.2	
ECSS-Q-60-02A-l	Timing of critical signals	3.10	
ECSS-Q-60-02A-m	Radiation environment constraints	3.7	

ECSS-Q-60-02A-n	Thermal environment constraints	3.6	Depends ultimately on target technology
ECSS-Q-60-02A-o	Power budget and dissipation	3.8	Depends ultimately on target technology
ECSS-Q-60-02A-p	Physical and mechanical constraints: pin assignment, size, encapsulation	3.9	Depends ultimately on target technology
ECSS-Q-60-02A-q	Reusability or additional functions for future applications	6	
ECSS-Q-60-02A-r	Portability to different or newer technologies	6.2	
ECSS-Q-60-02A-s	Intellectual property rights of the design to be developed	-	The activity's Development Plan [AD6] contains information regarding IP rights.
ECSS-Q-60-02A-t	Proprietary designs (IP cores) to be used as building blocks of the design to be developed, if already identified	2 (subsection for each core)	