



Dependable Real-time Infrastructure for Safety-critical Computer

Project number: 869945

Project acronym: De-RISC

<http://www.derisc-project.eu/>

D1.1 Interim platform and domain requirement specification and definition

Work Package	WP1	Lead Beneficiary	CG
Type	Report	Dissemination level	Public
Due Date	31/12/2019	Version	0.2

Brief description

The purpose of this document is to collect requirements defining technology stack consisting of software architecture, system-on-chip architecture, and PCB board. The document also defines cross-cutting requirements that affect multiple layers in this stack to form a complete requirement set for the Dependable Real-time Infrastructure for Safety-Critical Computer (De-RISC) platform







Document control page

Written by

Name	Beneficiary	Date
Jan Andersson	CG	18/11/2019
Jimmy Le Rhun	TRT	26/11/2019
Miguel Masmano Tello	FEN	03/12/2019
Jaume Aubella	BSC	06/12/2019
David Trilla	BSC	10/12/2019
Nils-Johan Wessman	CG	16/12/2019

Reviewed by

Name	Beneficiary	Date
Jimmy Le Rhun	TRT	30/12/2019

Approved by

Name	Beneficiary	Date
Francisco Gómez Molinero	FEN	31/12/2019

Change log

Version	Date	Name	Beneficiary	Comments
0.0	18/11/2019	Jan Andersson	CG	Initial draft
0.1	03/12/2019	Miguel Masmano Tello	FEN	Internal draft
0.2	30/12/2019	Jimmy Le Rhun	TRT	First release



Disclaimer

This document may contain material that is copyright of certain De-RISC beneficiaries, and may not be reproduced, copied, or modified in whole or in part for any purpose without written permission from the De-RISC Consortium. The commercial use of any information contained in this document may require a license from the proprietor of that information. The information in this document is provided “as is” and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

The De-RISC Consortium comprises the following partners:

#	Partner legal name	Short name	Acronym	Country
1	FENT INNOVATIVE SOFTWARE SOLUTIONS SL	fentISS	FEN	Spain
2	BARCELONA SUPERCOMPUTING CENTER - CENTRO NACIONAL DE SUPERCOMPUTACIÓN	BSC	BSC	Spain
3	THALES SA	THALES	TRT	France
4	COBHAM GAISLER AB	COBHAM GAISLER	CB	Sweden





Table of contents

1. Scope of the document.....	7
2. Applicable and reference documents.....	8
3. Terms, definitions and acronyms.....	10
3.1. Terms and definitions.....	10
3.2. Acronyms.....	10
4. Introduction.....	11
4.1. Requirement format.....	11
5. Global Platform architecture.....	13
5.1. Hardware platform.....	13
5.2. Software platform.....	21
5.3. Preliminary block diagram.....	23
5.3.1. Hardware platform.....	23
5.3.2. Software platform.....	23
5.4. Physical and Resource constraints.....	24
5.5. Environment constraints.....	25
6. Baseline elements description.....	28
6.1. Microprocessor core.....	28
6.1.1. Relevant RISC-V standards.....	28
6.1.2. Processor core overview.....	28
6.1.3. NOEL-V subsystem.....	29
6.1.4. Memory management unit.....	31
6.1.5. On-chip debug support.....	31
6.2. GRLIB IPs.....	32
6.3. XNG hypervisor overview.....	32
6.3.1. XNG configuration.....	33
6.3.2. Partitioning.....	34
6.3.3. Partition scheduling.....	34
6.3.4. Spatial partitioning.....	34
6.3.5. Partition virtual execution environment (PVEE).....	34
6.3.6. Health monitor.....	35
6.3.7. Inter-partition communication.....	36
6.3.8. Delegated I/O devices.....	36
6.4. LithOS overview.....	36
6.4.1. Configuration.....	37
6.4.2. Partition management.....	38
6.4.3. Process management.....	38
6.4.4. Time management.....	38
6.4.5. Inter-partition communication.....	39
6.4.6. Intra-partition communication.....	39
6.4.7. Health Monitor.....	39
6.4.8. Multiple module schedule.....	40



6.4.9. Interrupt management.....	40
6.4.10. System management.....	40
6.5. Development tools.....	40
6.5.1. Gcc + binutils version (CG).....	40
6.5.2. GRMON debugger.....	41
6.5.3. xcparser tool.....	41
7. Detailed sub-systems specification.....	42
7.1. Processor core.....	44
7.1.1. MMU & privilege levels.....	45
7.1.1.1. Hypervisor support.....	45
7.1.2. Core interface.....	47
7.1.2.1. Hypervisor and RTOS considerations (FEN).....	47
7.2. Memory subsystem.....	48
7.2.1. Tightly coupled memory.....	49
7.2.2. L1 cache memory.....	49
7.2.3. L2 cache memory.....	51
7.2.4. External memory controller.....	54
7.2.4.1. Fault tolerance properties.....	54
7.2.4.2. Request arbitration.....	54
7.2.5. Global memory mapping.....	54
7.3. Interconnect.....	55
7.3.1. L1 to L2.....	55
7.3.2. Main interconnect.....	56
7.3.3. Cache coherency.....	56
7.3.3.1. Interference channels and isolation properties.....	57
7.3.3.2. IPC (inter-partition communication) support.....	57
7.3.4. IOMMU.....	57
7.3.4.1. Hypervisor support.....	57
7.4. IO Peripherals.....	58
7.4.1. UARTs.....	58
7.4.2. Timers.....	58
7.4.3. Interrupt controller.....	59
7.4.4. Ethernet.....	60
7.4.5. DMAs.....	60
7.4.6. SpaceWire.....	60
7.5. Software architecture.....	60
7.5.1. Time and space partitioning.....	60
7.5.2. Deterministic partition scheduling.....	61
7.5.3. Explicit inter-partition communications.....	61
7.6. Debug and monitoring.....	61
7.6.1. Debug interface.....	61
7.6.2. Performance and interference monitoring.....	62
7.7. Reset and Boot process.....	62
8. Documentation.....	63



1. Scope of the document

The purpose of this document is to collect requirements defining technology stack consisting of software architecture, system-on-chip architecture, and PCB board. The document also defines cross-cutting requirements that affect multiple layers in this stack to form a complete requirement set for the Dependable Real-time Infrastructure for Safety-Critical Computer (De-RISC) platform.

The work has been performed by FentISS (ES), Cobham Gaisler (SE), Barcelona Supercomputing Center (ES), and Thales Research and Technology (TRT).

2. Applicable and reference documents

- [AD01] ECSS-E-ST-40C, Software general requirements Available at <http://www.ecss.nl>
- [AD02] ECSS-Q-ST-80C, Software product assurance Available at <http://www.ecss.nl>
- [AD03] ECSS-E-ST-20C, Electrical and Electronics Available at <http://www.ecss.nl>
- [AD04] ECSS-E-ST-50-12C , SpaceWire – Links, Nodes Routers and Networks Available at <http://www.ecss.nl>
- [AD05] ECSS-E-ST-50-13C, Interface and communication protocol for MIL-STD-1553B data bus onboard spacecraft, Available at <http://www.ecss.nl>
- [AD06] ECSS-E-ST-50-14C, Spacecraft Discrete interfaces Available at <http://www.ecss.nl>
- [AD07] ECSS-E-ST-50-15C, CANbus extension protocol Available at <http://www.ecss.nl>
- [AD08] ECSS-E-ST-50-51C, SpaceWire – Protocol identification Available at <http://www.ecss.nl>
- [AD09] ECSS-E-ST-50-52C, SpaceWire – Remote Memory Access Protocol Available at <http://www.ecss.nl>
- [AD10] ECSS-E-ST-50-53C, SpaceWire – CCSDS packet transfer protocol Available at <http://www.ecss.nl>
- [AD11] ECSS-E-70-41C, Telemetry and telecommand packet utilization, ECSS-E-70-41c Available at <http://www.ecss.nl>
- [AD12] ECSS-E-ST-10-06, Functional and Technical Specifications Available at <http://www.ecss.nl>
- [AD13] ECSS-Q-ST-30-02C, Failure modes, effects and criticality analysis (FMECA) Available at <http://www.ecss.nl>
- [AD14] ECSS-Q-ST-60-02C, ASIC and FPGA Development Standard Available at <http://www.ecss.nl>
- [AD15] SAVOIR-GS-001 SAVOIR – SAVOIR generic OBC functional spec Available at <http://essr.esa.int>
- [AD16] SAVOIR-GS-002 SAVOIR – Flight Computer Initialisation Sequence Available at <http://essr.esa.int>
- [AD17] ECSS-Q-ST-70C Materials, mechanical parts and processes Available at <http://www.ecss.nl>
- [AD18] SAVOIR-TM-002 SAVOIR UART protocol and interface specification Available at <http://essr.esa.int>
- [AD19] ECSS-E-ST-32-10C Structural factors of safety for spaceflight hardware Available at <http://www.ecss.nl>
- [AD20] GRLIB IP Core User’s Manual. Available at <https://www.gaisler.com/getglib>
- [AD21] Software User Manual – XNG hypervisor. Fent Innovative Software Solutions14-033.009.sum, Issue 7.



2. Applicable and reference documents

[RD01] Avionics Application Software Standard Interface. Part 1 - Required Services. ARINC Specification 653P1-3. Aeronautic Radio, Inc.

[RD02] Avionics Application Software Standard Interface. Part 2 - Extended Services. ARINC Specification 653P2-2. Aeronautic Radio, Inc.

[RD03] Multi-core Processors - Position Paper. Technical Report CAST 32-A, FAA, 2016.

3. Terms, definitions and acronyms

3.1. Terms and definitions

Term	Definition
Horizon 2020	It is the biggest EU Research and Innovation programme ever with nearly €80 billion of funding available over 7 years (2014 to 2020) – in addition to the private investment that this money will attract. The purpose of Horizon 2020 is to foster the growth of breakthrough technologies, inventions and advanced developments by the promotion of scientific ideas from the laboratories to the market.
RISC-V	Pronounced "risk-five". It is an open-source hardware instruction set architecture based on established reduced instruction set computer principles.

3.2. Acronyms

Acronym	Definition
BSC	Barcelona Supercomputing Center
CG	Cobham Gaisler
EC	European Commission
EU	European Union
ESA	European Space Agency
FEN	fentISS
H2020	Horizon 2020
HW	Hardware
ISA	Instruction Set Architecture
NoC	Network-on-Chip
RISC	Reduced Instruction Set Computer
RISC-V	Reduced Instruction Set Computer Five
SoC	System-on-Chip
SW	Software
TC	Telecommand
TM	Telemetry
TRL	Technology Readiness Levels
TRT	Thales Research & Technology
WP	Work Package

4. Introduction

The De-RISC platform to be developed within this project consists of three levels:

- Software architecture: Software environment consisting of the XtratuM Hypervisor and the LithOS Operating System.
- System-on-Chip (SoC) architecture: Digital hardware design consisting of general-purpose microprocessors, peripherals, communication controllers and interconnect
- De-RISC FPGA board: Implementation of SoC architecture in programmable logic on a custom PCB board.

This requirement specification has the following structure:

- Section 5 is the coarse-grain view of the platform
- Section 6 describes the existing building blocks used for the platform
- Section 7 details specific functional units to be developed/adapted, consolidating requirements for both hardware, software and (non)functional properties in the same place to ensure consistency.

4.1. Requirement format

The following table describes the wording used for the requirements specification:

Shall	The word “Shall” in the text expresses a mandatory requirement of the specification.
Should	The word “Should” in the text expresses a recommendation or advice on implementing such a requirement of the specification.
Must	The word “Must” in the text is used for legislative or regulatory requirements (e.g. health and safety). It is not used to express a requirement of the specification.
Will	The word “Will” in the text denotes a provision or service or an intention in connection with a requirement of the specification.
May	The word “May” in the text expresses a permissible practice or action. It does not express a requirement of the specification.

N/A	Not applicable.
-----	-----------------

The following template shall be used for the formulation of requirements:

<identifier> DeRISC-T1.x-AA-ID000	<requirement description according to the SMART criteria “Specific Measurable Acceptable Realistic and Time-bound”> e.g. The demonstrator shall
<status> {proposed, accepted, rejected}	
<qualification method> e.g. black box test, (code) inspection, demonstration	
<rationale>	
<comment>	

T1.X = task number

ID000 = requirement ID number

ID0xx	Fentiss
ID1xx	Cobham Gaisler
ID2xx	BSC
ID3xx	Thales

AA = Requirement type

Functional Requirements	FU
Input/Output Requirements	IO
Hardware Requirements	HW
Software Requirements	SW
Mechanical/Physical Requirements	ME
Performance Requirements	PR
Safety Requirements	SA
Security Requirements	SE

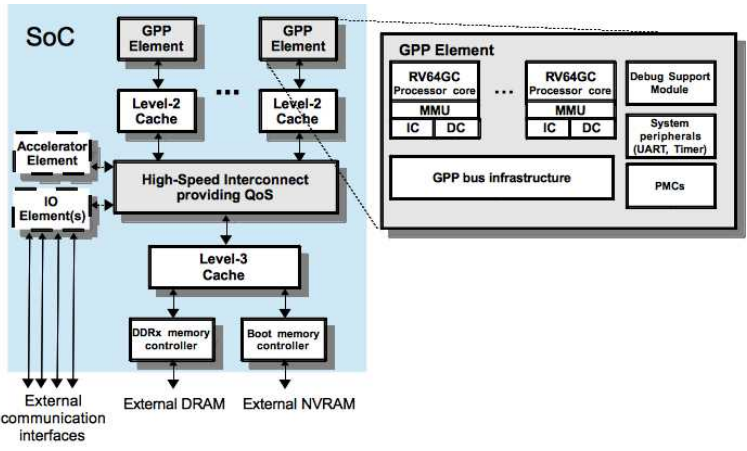
5. Global Platform architecture

5.1. Hardware platform

DeRISC-T1.1/T1.2-FU-ID100	The De-RISC FPGA platform shall be available as a package of software that can be used together with the De-RISC board
proposed	
Qualification method: Demonstration	
Rationale:	
Comment:	

DeRISC-T1.1/T1.4-HW-ID100	The De-RISC SoC shall be implemented in programmable logic (FPGA)
proposed	
Qualification method: Inspection	
Rationale: FPGA technology is the only feasible target technology within the project timeline. FPGAs also allows application specific adaptations of the SoC system.	
Comment: The choice of target FPGA will be settled for the final issue of this document.	

DeRISC-T1.1-HW-ID101	The SoC design shall implement at least four microprocessor cores
proposed	
Qualification method: Inspection	
Rationale: Addition of additional four cores provides additional computational performance where target technologies are often limited in maximum frequency. Current applications within space use multi-core systems to achieve higher integration where independent software functions run on separate cores within one processor device.	
Comment: The number of processor cores affect how much logic that is required on the target technology. Number of cores drives FPGA selection and FPGA selection may limit the number of cores.	

DeRISC-T1.1-HW-ID102	<p>The SoC topology should match the topology put forth in the De-RISC proposal:</p>
proposed	 <p>The diagram illustrates the SoC topology. At the top, two GPP Elements are connected to Level-2 Caches. These connect to a High-Speed Interconnect providing QoS. Below this is a Level-3 Cache, which connects to DDRx memory controller and Boot memory controller. These controllers connect to External DRAM and External NVRAM. On the left, Accelerator Element and IO Element(s) connect to the High-Speed Interconnect. On the right, a detailed GPP Element block shows RV64GC Processor core, MMU, IC, DC, Debug Support Module, System peripherals (UART, Timer), and PMCs, all connected to GPP bus infrastructure.</p>
Qualification method: Inspection	
Rationale: Re-use of building blocks places requirements on topology	
Comment:	

DeRISC-T1.1-HW-ID103	<p>The general purpose microprocessor core shall implement the RISC-V 64-bit ISA with the GC (IMAFDC) extensions</p>
proposed	
Qualification method: Demonstration	
<p>Rationale: G has been identified in ESA studies as the required feature set to match existing space-grade processors. C improves memory utilization and possibly performance for some applications. A is required for multi-core applications.</p>	
Comment:	

DeRISC-T1.1-HW-ID104	The general purpose microprocessor should implement the RISC-V N extension
proposed	
Qualification method: Demonstration	
Rationale: N is likely to be found to be required however few other implementations exist.	
Comment:	

DeRISC-T1.1-HW-ID105	The general purpose microprocessors shall implement a Memory Management Unit
proposed	
Qualification method: Demonstration	
Rationale: MMU is required to run target software environments	
Comment:	

DeRISC-T1.1-HW-ID106	The hardware platform shall maintain cache coherency between L1 caches within a GPP element.
proposed	
Qualification method: Demonstration	
Rationale: Hardware assisted cache coherency reduces effort for SW. Legacy software from the space domain may depend on this functionality.	
Comment: Current building blocks make use of bus snooping. Driving the selection of interconnect.	

DeRISC-T1.1-HW-ID107	The interconnect between the GPP elements and Level-2 cache shall be based on AMBA2 AHB.
proposed	
Qualification method: Inspection	
Rationale: Required for hardware cache coherency	
Comment:	

DeRISC-T1.1-HW-ID104	The interconnect between the GPP elements and Level-2 cache should make use of multi-level buses to reduce interference in the general case.
proposed	
Qualification method: Inspection	
Rationale: Multi-layer address-striped interconnect has been measured to improve average processing performance	
Comment:	

DeRISC-T1.1/T1.4-HW-ID108	The De-RISC platform shall support use of external DDR2, DDR3, or DDR4 SDRAM as primary/working memory
proposed	
Qualification method: inspection	
Rationale: DDR2/3/4 SDRAM is required for bandwidth and required memory density (size) for space applications.	
Comment: Considering that the FPGA implementation will be limited in maximum operating frequency there could be a case for use of SSRAM running synchronously with the system clock. However SSRAM has lower memory density compared to DDRx SDRAM.	

DeRISC-T1.1/T1.4-HW-ID109	The platform shall support use of external parallel non-volatile memory (NOR Flash, MRAM)
proposed	
Qualification method: Inspection	
Rationale: Parallel NOR Flash and parallel MRAM are traditionally used for boot memory in space applications	
Comment: While the platform shall support use of parallel memory it is not required that this type of memory components are mounted on the De-RISC board.	

DeRISC-T1.1/T1.4-HW-ID110	The platform shall support use of external SPI non-volatile memory as boot memory.
proposed	
Qualification method: Inspection	
Rationale: High-reliability SPI memory devices exist and can provide smaller form factor compared to parallel versions.	
Comment: Boot from SPI can be implemented either through a SPI memory controller that provides a memory-mapped view of the external device or through use of a boot ROM that makes use of software and a traditional SPI controller. While the platform shall support use of SPI memory it is not required that this type of memory components are mounted on the De-RISC board.	

DeRISC-T1.1/T1.4-HW-ID111	The platform shall support high-capacity non-volatile storage in NAND Flash
proposed	
Qualification method: Inspection	
Rationale: NAND Flash is the only currently used non-vol technology that can offer gigabyte of storage for space applications.	
Comment: While the platform shall support use of NAND Flash memory it is not required that this type of memory components are mounted on the De-RISC board.	

DeRISC-T1.1/T1.4-HW-ID112	The platform shall provide at least two Gigabit Ethernet interfaces
proposed	
Qualification method: Inspection	
Rationale: Ethernet is a convenient debug link and is widely used in automotive and aerospace applications. NASA has through the Lunar missions used Ethernet for new platforms. The European Ariane6 computer makes use of Ethernet.	
Comment: Number of Ethernet interface supported by the FPGA board will be limited by	

FPGA and board space.

DeRISC-T1.1/T1.4-HW-ID113	The platform shall provide at least two SpaceWire interfaces each with a like rate of 200 Mbit/s
proposed	
Qualification method: Demonstration	
Rationale: SpaceWire is widely used in space applications	
Comment: Number of interfaces supported by the FPGA board will be limited by FPGA and board space.	

DeRISC-T1.1/T1.4-HW-ID114	Hardware shall provide routing between the SpaceWire interfaces
proposed	
Qualification method: Demonstration	
Rationale: Routing functionality is provided by existing space-grade devices (GR740)	
Comment: SpW router/switch functionality is intended, as opposed to routing implemented by SW.	

DeRISC-T1.1/T1.4-HW-ID115	The platform shall provide at least two SpaceFibre interfaces with a rate of 2Gbit/s
proposed	
Qualification method: Inspection	
Rationale: High-Speed serial links are being adopted in space applications	
Comment:	

DeRISC-T1.1/T1.4-HW-ID116	The platform shall provide at least two CAN-FD interfaces
proposed	

Qualification method: Inspection
Rationale: CAN is seeing adoption in space and aerospace
Comment:

DeRISC-T1.1/T1.4-HW-ID117	The platform's CAN interfaces need to be connected to at least two separate CAN controllers
proposed	
Qualification method: Inspection	
Rationale: ESA CAN redundancy requirements require separate controllers	
Comment:	

DeRISC-T1.1/T1.4-HW-ID118	The platform shall provide at least two UART interfaces
proposed	
Qualification method: Inspection	
Rationale: UART remains a common interface to avioincs subsystems	
Comment: RS422 or RS485 needs to be selected for board implementation	

DeRISC-T1.1/T1.4-HW-ID119	The platform shall allow extensions to support MIL-STD-1553B
proposed	
Qualification method: Inspection	
Rationale: MIL-STD-1553B is widely used in aerospace and space applications	
Comment: It shall be demonstrated that the system can be extended with MIL-STD-1553B controllers. MIL-STD-1553 transceivers occupy a large amount of board space and it is considered infeasible to support the bus on the current iteration of the De-RISC board.	

DeRISC-T1.1/T1.4-HW-ID120	The platform shall provide at least one general-purpose SPI interface capable of master and slave mode
proposed	
Qualification method: Inspection	
Rationale:	
Comment:	

DeRISC-T1.1/T1.4-HW-ID121	The platform shall provide at least one I2C bus interface
proposed	
Qualification method: Inspection	
Rationale:	
Comment:	

DeRISC-T1.1-HW-ID122	The platform shall include an I/O memory management unit (IOMMU)
proposed	
Qualification method: Inspection	
Rationale:	
Comment:	

5.2. Software platform

DeRISC-T1.1-SW-ID000	The De-RISC platform shall include a partitioning kernel based on a hypervisor and, at partition level a runtime or operating system.
proposed	
Qualification method: Review of the De-RISC documentation.	
Rationale: A partition kernel allows to run multiple software systems in single platform. Currently, it is the approach being used for implementing IMA-based systems in the aerospace sector.	
Comment:	

DeRISC-T1.1-SW-ID001	The De-RISC platform's hypervisor shall be XNG.
proposed	
Qualification method: Review of the De-RISC documentation.	
Rationale: XNG was the selected hypervisor for building the proposed De-RISC architecture.	
Comment: XNG is qualified as ECSS software criticality class B.	

DeRISC-T1.1-SW-ID002	The De-RISC platform's runtimes and operating system, at partition level, shall be LithOS and XRE.
proposed	
Qualification method: Inspection	
Rationale: LithOS and XRE were the selected operating system and runtime for being used at partition level as component of the proposed De-RISC architecture.	
Comment: Both, LithOS and XRE, are qualified as ECSS software criticality class B.	

DeRISC-T1.1-SW-ID003	The De-RISC platform shall allow mixing applications with different criticality level.
proposed	
Qualification method: Inspection	
Rationale:	
Comment:	



5.2. Software platform

DeRISC-T1.1-SW-ID004	The De-RISC platform shall be qualified according to the ECSS E40 [AD01] and Q80 [AD02] standard for the software criticality class B.
proposed	
Qualification method: Verification of the resulting ECSS qualification package.	
Rationale: The DeRISC software platform requires to reach a high software criticality class for being eligible candidate for space missions.	
Comment:	

5.3. Preliminary block diagram

5.3.1. Hardware platform

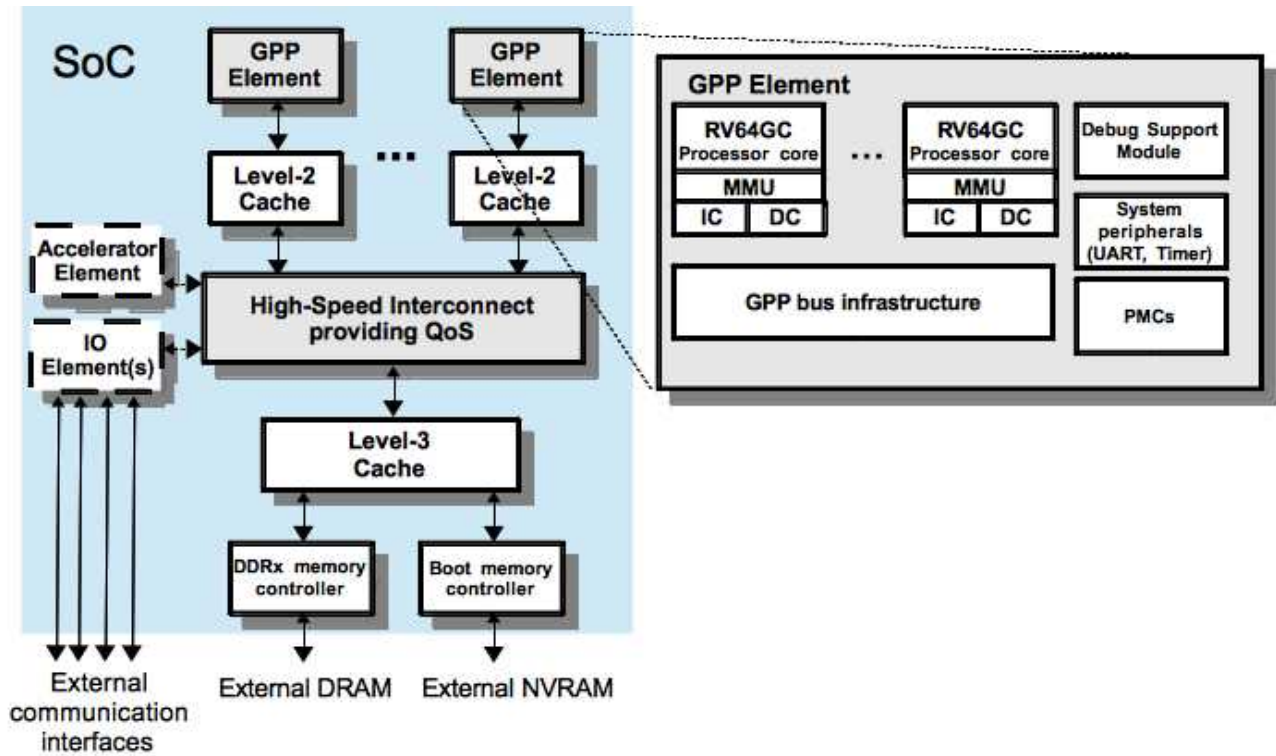


Figure 1: De-RISC hardware platform

5.3.2. Software platform

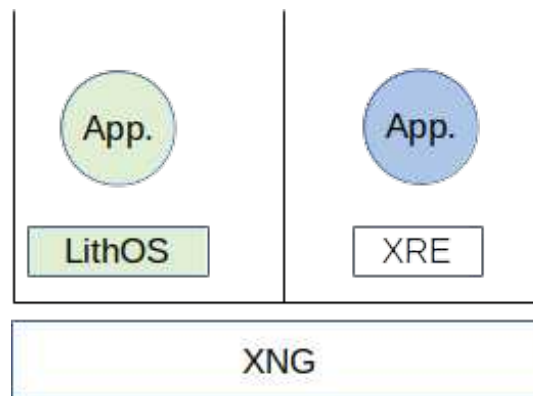


Figure 2: De-RISC software platform

5.4. Physical and Resource constraints

DeRISC-T1.1/T1.4-ME-ID102	The De-RISC FPGA board should have a 3U form factor
proposed	
Qualification method: Inspection	
Rationale: 3U is considered a preferred form factor according to space industry surveys performed by Cobham Gaisler.	
Comment: FPGA power circuitry and transceivers could drive the design to a 6U form factor. PC104 would be relevant for cubesats but is considered unfeasible for the proposed architecture.	

DeRISC-T1.1/T1.4-ME-ID102	The De-RISC FPGA board should a mass of less than 1.2 kg
proposed	
Qualification method: Measurement	
Rationale:	
Comment:	

DeRISC-T1.1/T1.4-ME-ID102	The De-RISC FPGA board should dissipate less than 10 W when operating in worst case configuration and at maximum performance.
proposed	
Qualification method: Analysis	
Rationale:	
Comment:	

5.5. Environment constraints

DeRISC-T1.1/T1.4-ME-ID103	<p>The De-RISC FPGA platform shall be designed in order to ensure that nominal performances are maintained w/o degradation in presence of (limited) radiation effects as:</p> <p>Total Ionizing radiation dose (TID), including NIEL damage,</p> <p>Single Event Latch-up (SEL),</p> <p>Single Event Upset (SEU),</p> <p>Burn-Out,</p> <p>Single Event Gate Rupture (SEGR),</p> <p>Single Event Transient (SET),</p> <p>Single Event Functional Interrupt (SEFI)</p>
proposed	
Qualification method: Analysis	
Rationale: Board shall be usable for flight applications	
Comment:	

DeRISC-T1.1/T1.4-ME-ID104	<p>Total dose: The De-RISC platform shall be able to perform according to functional and performance requirements up to a Total Ionizing Dose (TID) level of TBD krad-Si and a TBD MeV Total Non Ionizing Dose TNID.</p>
proposed	
Qualification method: Analysis	
Rationale:	
Comment: TBD doses to be clarified by M06.	

DeRISC-T1.1/T1.4-ME-ID104	Single event effects: The De-RISC platform shall be immune to destructive events (SEL, SEB, SEGR) with LETth > TBD MeV*cm2/mg.
proposed	
Qualification method: Analysis	
Rationale:	
Comment: TBD limit to be clarified by M06.	

DeRISC-T1.1/T1.4-ME-ID104	<p>Temperature: Operating temperature range for the Flight Model shall be -30 deg C to 70 deg C measured at board Temperature Reference Point</p> <p>Not operating temperature shall be -40 deg C to 80 deg C.</p>
proposed	
Qualification method: Analysis	
Rationale:	
<p>Comment: The TRP (Temperature Reference Point) of the Board shall be the average temperature of the main structure's I/F nodes. It shall represent the average temperature of the main structure face.</p>	

DeRISC-T1.1/T1.4-ME-ID104	<p>Mechanical loads: The hardware shall be designed to sustain the loads generated by the following events:</p> <ul style="list-style-type: none"> • Manufacturing, assembly, ground handling and transportation loads, • Test loads, • Launch\landing loads: quasi static, vibration (including shock), thermal and pressure loads, • Operational loads : including thermal, mechanism induced and pressure loads.
proposed	
Qualification method: Analysis	
Rationale:	
Comment:	

6. Baseline elements description

The subsections below describe existing elements of the DeRISC platform.

6.1. Microprocessor core

6.1.1. Relevant RISC-V standards

Unprivileged Specification:

The RISC-V Instruction Set Manual

Volume I: Unprivileged ISA

Document Version 20191213

Privileged ISA Specification:

The RISC-V Instruction Set Manual

Volume II: Privileged Architecture

Document Version 20190608-Priv-MSU-Ratified

Debug Specification:

RISC-V External Debug Support

Version 0.13.2

6.1.2. Processor core overview

The NOEL-V is a synthesizable VHDL model of a 64-bit processor that implements the RISC-V architecture. The NOEL-V is dual-issue, allowing up to two instructions per cycle to be executed in parallel. To support the instruction issue rate of the pipeline, the NOEL-V has advanced branch prediction capabilities. The cache controller of the NOEL-V supports a store buffer FIFO with one cycle per store sustained throughput, and wide AHB slave support to enable fast stores and fast cache refill.

The NOEL-V is interfaced using the AMBA 2.0 AHB bus (subsystem with Level-2 cache and AXI4 backend is also available) and supports the IP core plug&play method provided in the Cobham Gaisler IP library (GRLIB). The processor can be efficiently implemented on FPGA and ASIC technologies and uses standard synchronous memory cells for caches and register file.

The NOEL-V processor has the following features:

- RISC-V RV64GC
 - 64-bit architecture

- Hardware multiply and divide units
- Compressed (16 bit) instruction support
- Atomic instruction extension
- 32/64 bit floating point extensions using non-pipelined area efficient FPU or high-performance fully pipelined IEEE-754 FPU
- Machine, supervisor and user mode. RISC-V standard MMU with configurable TLB.
- User level interrupts
- RISC-V standard PLIC
- RISC-V standard PMP (physical memory protection)
- RISC-V standard external debug support
- Advanced 7-stage dual-issue in-order pipeline
- Dynamic branch prediction, branch target buffer and return address stack
- Four full ALUs, two of them late in the pipeline to reduce stalls
- Separate instruction and data L1 cache (Harvard architecture) with snooping

High Performance: CoreMark: 4.69 CoreMark/MHz

Additional documentation of the NOEL-V processor can be found in [AD20].

6.1.3. NOEL-V subsystem

The NOEL-V processor is available as a subsystem, with the product name NOELVSUBSYS. A functional block diagram is shown below. The subsystem can be configured to use several processor core instantiation and corresponds to a GPP element in the proposed De-RISC system topology (also shown below)

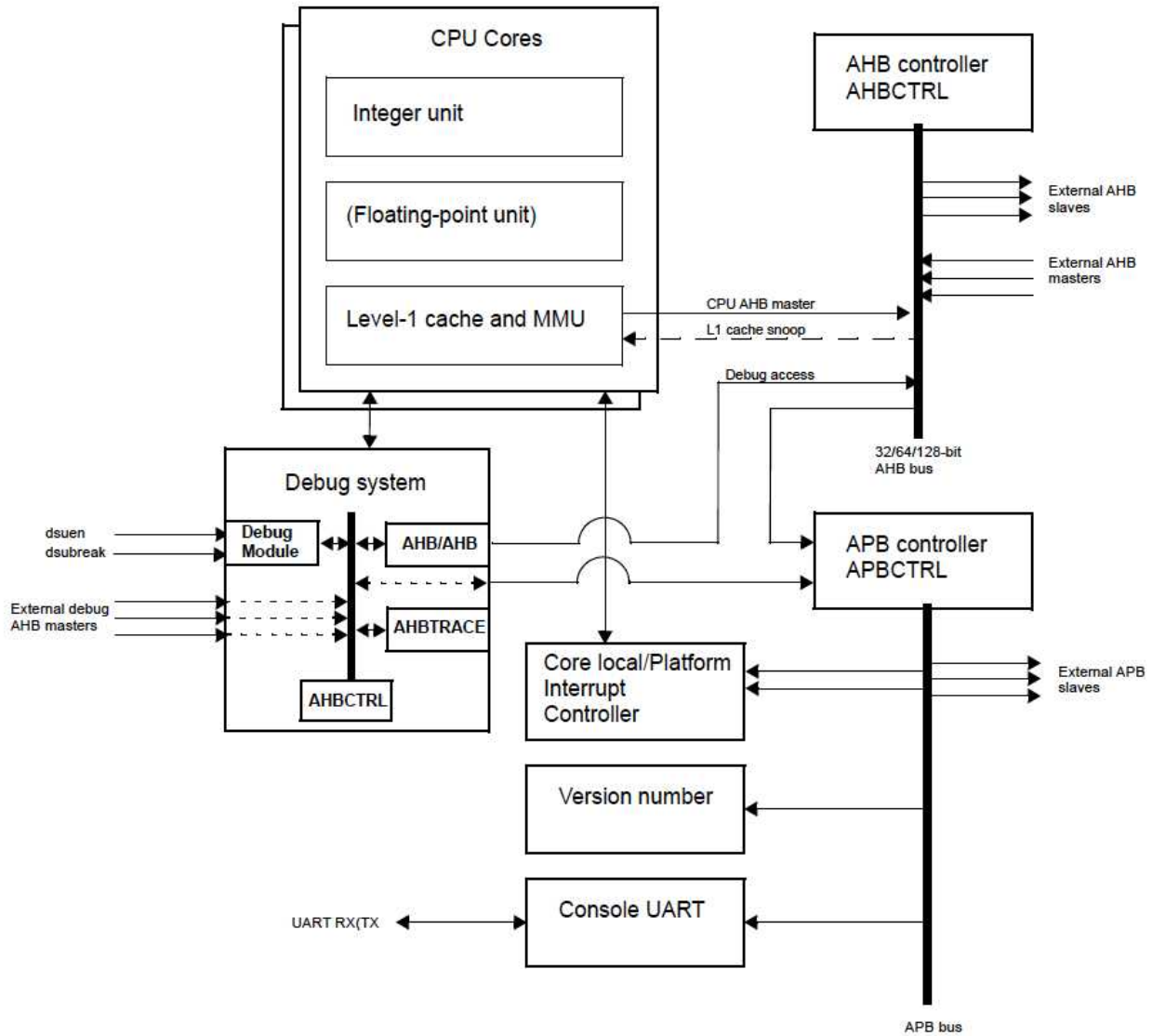


Figure 3: NOEL-V subsystem

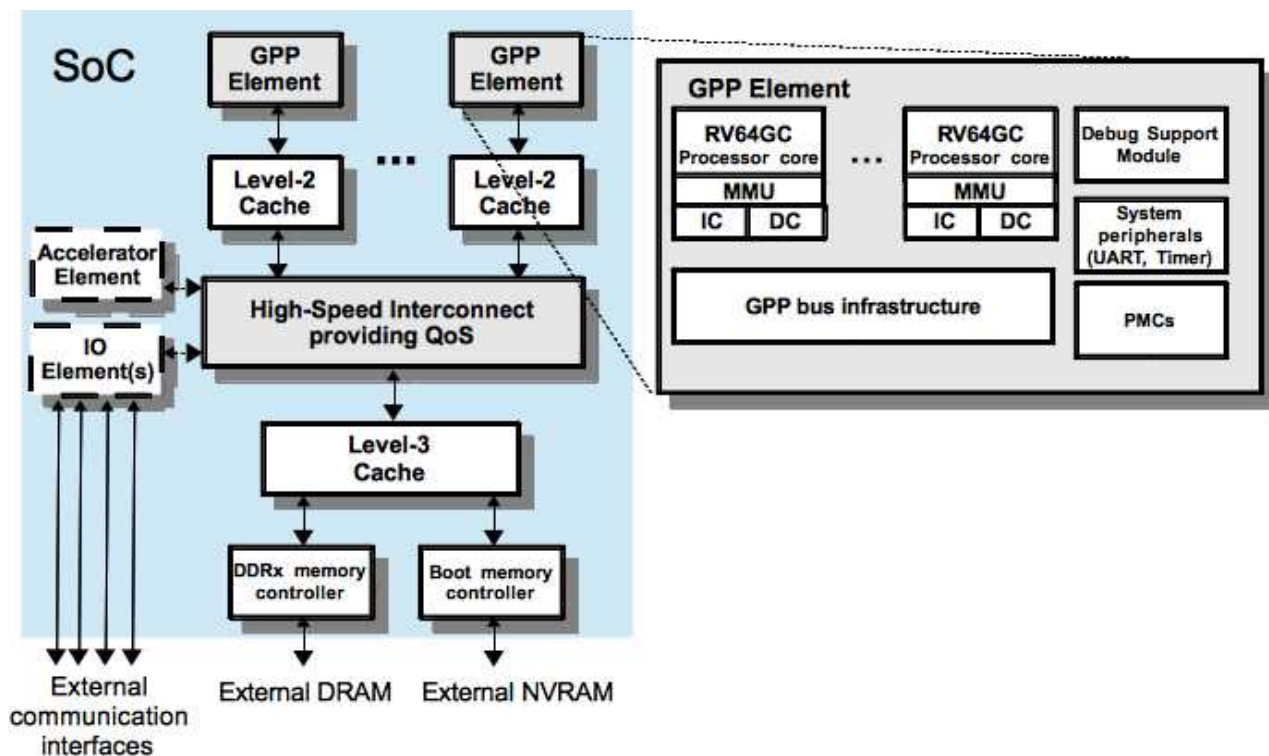


Figure 4: De-RISC hardware platform

The external bus interfaces of the AMBA subsystem adhere to AMBA2 AHB and APB. Interrupt propagation in to the subsystem is performed through a side-band vector to the AMBA bus and are then handled by a standard RISC-V PLIC implementation. The debug interface implements the RISC-V debug specification.

6.1.4. Memory management unit

A Memory Management Unit is integrated into the NOEL-V cache controller. It implements the full RISC-V specification, and provides mapping between multiple 64-bit virtual address spaces and physical memory. A hardware table-walk is implemented, and the MMU has separate 8-entry fully associative TLBs for instruction and data.

[Note: only Sv39 (i.e. 39 bit virtual addressing using a three level page table) is available.]

6.1.5. On-chip debug support

The NOEL-V subsystem includes functionality to allow non-intrusive debugging on target hardware. Through a debug module interface, full access to all processor registers is provided. The debug interfaces also allows single stepping, instruction tracing. An internal trace buffer can monitor and store executed instructions, which can later be read out via the debug interface.

Interfaces to access the debug functionality are connected using a dedicated AHB port that does not

interfere with normal operation.

6.2. GRLIB IPs

Cobham Gaisler's GRLIB IP library is a library of reusable IP cores. The components in the library are inherently portable and include, next to the NOEL-V processor, a wide range of building blocks. The controllers and building blocks that are relevant for the De-RISC system (system architecture to be defined through the final issue of this document) are:

- L2CACHE - Level-2 cache
- FTADDR - Fault-tolerant DDR2/3 SDRAM controller
- GRIOMMU - I/O memory management unit
- GRETH_GBIT - 10/100/1000 Mbit Ethernet MAC
- GRSPWROUTER - SpaceWire router
- GRSPFI - SpaceFibre node interface controller
- GRCANFD - CAN-FD controller
- APBUART - 8-bit UART
- GRGPIO - General Purpose IO unit

Documentation for the above components is available in [AD20].

6.3. XNG hypervisor overview

XtratuM-NextGeneration (XNG) [AD21] is a bare metal hypervisor designed for being used in real-time, safety critical systems. XNG enables the execution of several software systems on a hardware host machine. Each software system is executed in an isolated virtual environment, known as partition. This virtual environment mimics the behaviour of the underlying hardware architecture (virtual processor, virtual interrupt controller, system clock and system timer, etc) easing the portability of existing software systems. The hypervisor also provides partitions with high-level functionalities such as the management of hypervisor, the management of partitions, inter-partition communication based on ARINC653-like sampling and queuing ports [RD01], health monitoring, etc.

Figure 5 illustrates a typical XNG-based system architecture, in this example, three partitions (a LithOS-based partition, a Linux-based partition and a XRE-based partition) run on top of XNG.

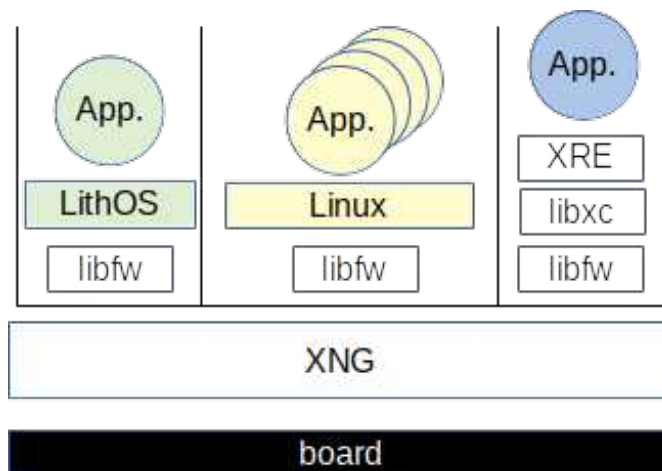


Figure 5: XNG- based system

Figure 1: XNG-based system

As it can be seen in this figure, XNG, at the run-time, is composed by four software components:

- XNG: the hypervisor core.
- libfw: library implementing C wrappers for the services provided by the hypervisor.
- libxc: optional library implementing several C standard functionality.
- xre: minimal runtime, designed for being used to write single threaded application. It is provided as a library.

6.3.1. XNG configuration

The XNG Configuration File (XCF) allows the system integrator to define the amount of resources that will be allocated to each partition and to the hypervisor during the execution of the system.

The configuration of the hypervisor is typically defined during the design of the system and therefore before the execution starts. This allows the system integrator to guarantee a fixed allocation of resources for all the components (hypervisor and partitions).

The *xcparser* tool is provided jointly with XNG to process the user-produced XML formatted XCF to generate the final binary that is used to configure XNG at initialisation.

6.3.2. Partitioning

A partition is a program unit designed to isolate a software application with respect to memory (spatial partitioning) and time (temporal partitioning). XNG defines two types of partitions: *normal* and *system*. Normal partitions can manage and monitor their state, while system partitions can additionally perform these actions over the system and over other partitions.

XNG runs multiple partitions concurrently with different critical levels without affecting one another spatially or temporally.

6.3.3. Partition scheduling

XNG schedules partitions according to a time-based activation scheduling (cyclic scheduling) policy as requested by ARINC653 [RD01] at partition level.

Every schedule defines a set of partition windows (identifying when and how a given partition runs within the schedule) and a major frame (MAF, defining the schedule's re-execution period).

Additionally, XNG supports multiple schedule plans. That is, the system integrator may define several schedules in the XCF, where one is the current (active) schedule at any given time. System partitions are able to change the current schedule at the end of the MAF or at any instant.

Finally, XNG also provides the capability to synchronise the schedule cycle with an external signal, of a periodic nature, that is notified to the hypervisor in the form of an IRQ. At the end of each MAF the hypervisor enters a special state awaiting for the arrival of the IRQ which triggers the start of the next MAF.

6.3.4. Spatial partitioning

Each partition has predetermined memory areas allocated to it. Any access of a partition to a not assigned resource is detected and avoided by the hypervisor.

6.3.5. Partition virtual execution environment (PVEE)

XNG provides partitions with a virtual representation of the underlying hardware. This environment is known as the PVEE.

The PVEE defines the following virtual devices at partition level:

- Virtual CPU (*vCpu*): The *vCpu* represents the underlying hardware CPU.
- System clock (*vClock*): The *vClock* allows a partition to get the current system time (in μ seconds).
- System timer (*sysTimer*): The *sysTimer* allows a partition to arm a virtual timer (in μ seconds) using the system clock as base. The virtual timer accepts two modes:
 - One-shot mode The virtual timer gets disarmed once its count expires.
 - Periodic mode The virtual timer is re-armed when its count expires.
- Execution clock (*execClock*): The *execClock* allows a partition to get the current partition's execution time (in μ seconds). Note that this clock, unlike the system clock, just counts the time of the partition owning it, being suspended while its owning partition does not run.
- Execution timer (*execTimer*): The *execTimer* allows a partition to arm a virtual timer (in μ seconds) using the current partition's execution clock as base. The virtual timer accepts two modes:
 - One-shot mode The virtual timer gets disarmed once its count expires.
 - Periodic mode The virtual timer is re-armed when its count expires.
- Virtual interrupt controller (*vIrqCtrl*): The *vIrqCtrl* allows a partition to manage its virtual IRQs (*vIrqs*). Among other things, a partition is able to mask or unmask the occurrence of a *vIrq*, requested the *vIrq* vector when a *vIrq* occurs, etc.
- Console: The console is a facility provided to enable the partition to display information through a physical serial port (UART) or a buffer stored in hypervisor memory. The console is shared by the hypervisor and all the partitions.

6.3.6. Health monitor

XNG defines a HM service inspired in the ARINC-653 HM [RD01]. The HM is in charge of detecting, reacting and reporting hardware, partitions and XNG faults and failures. Its purpose consists in discovering the errors at an early stage, trying to solve or confine the faulty subsystem to avoid a failure or to reduce its potentially harmful effects. The HM allows the system integrator to define a FDIR policy specific for the system and foreach partition event. This configuration is defined in the XCF.

6.3.7. Inter-partition communication

XNG defines the communication ports as an Inter-Partition Communication (IPC) mechanism inspired in the ARINC-653 standard. This communication is implemented via messages that are defined as contiguous blocks of data of finite length.

The communication is performed through two kind of ports:

- *Sampling port*: A message remains in the port until it is overwritten by a new occurrence of the message. A message is sent from a single source to one or more destinations.
- *Queuing port*: Messages are buffered. A message sent by a partition is stored in the message queue of the port until it is read by another partition. Messages are served in FIFO order by the virtualisation layer. The buffer has a configurable size. A message is sent from a single source to one destination.

6.3.8. Delegated I/O devices

XNG allows the system integrator to delegate a hardware platform I/O device to a partition. The access to IRQs and I/O ports (memory registers) of the device is configured in XCF.

6.4. LithOS overview

LithOS is a Real-Time Operating System (RTOS) designed specifically to be executed inside a XNG partition in an IMA-based system supporting space flight functions of a criticality classification of up to level B1.

Its purpose is to act as a guest operating system providing services compliant with the APEX API defined by the ARINC-653 specification. More precisely, it is compliant with the API described by [RD01] and a subset of [RD02]. In the terminology used by the ARINC-653, jointly LithOS and XM should be considered as the core software of an integrated module [RD01, §2.1]. The architecture of such a system is presented in figure 2, in which the dashed line depicts the boundary of what is considered the core software in the standard.

Note that the figure has been elaborated to resemble the elements depicted in [RD01, Figure 2.1].

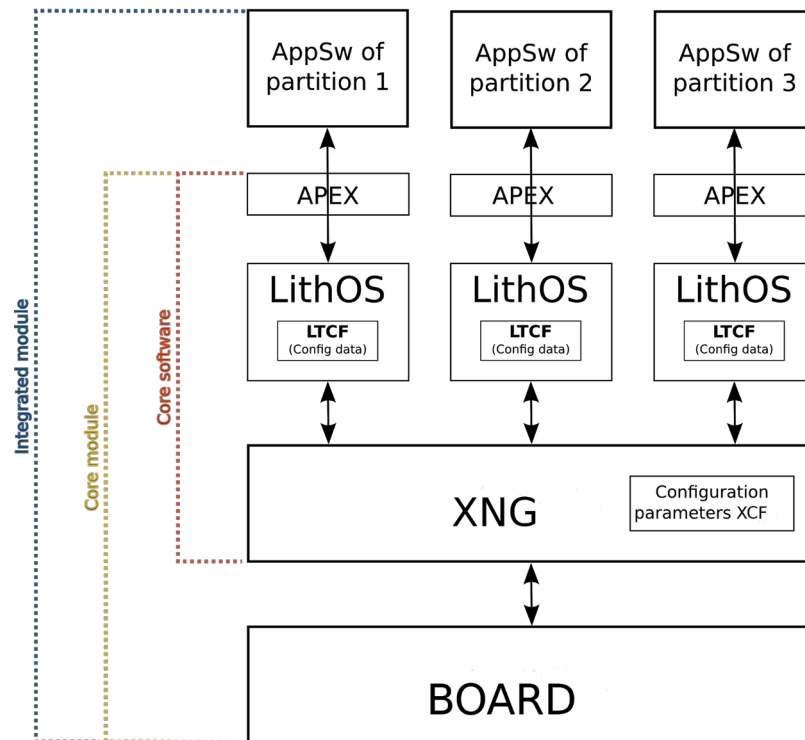


Figure 6: LithOS system architecture

From the point of view of functionalities, the ARINC-653 standard defines:

1. Services and functionalities related to the management of partitioning: Partition's time management, inter-partition communications, and partition-level health monitoring. These resources are configured statically through a standard XML configuration specification.
2. Services and functionalities related to intra-partition resources management: Processes, processes' time management, intra-partition communication, and process-level health monitoring.

While the functions in A are not implemented by LithOS itself but by XNG, this remains transparent to the software (Application Software) requesting services through the APEX interface provided by LithOS.

6.4.1. Configuration

LithOS allows to configure, in the LTCF, the maximum amount of resources that will be allocated to support the services provided by its ARINC-653 API. The configuration parameters accepted by LithOS correspond to the system limits (`SYSTEM_LIMIT_{parameter}`) specified as

implementation dependent in [RD01, Annex E], plus some additional parameters that contribute to further limit the memory footprint of LithOS.

The configuration is performed during the development of the AppSw and the configuration values can be modified without needing to recompile the code of LithOS nor the AppSw. With this approach, the total amount of memory allocated to the partition can be tightly adjusted to the real needs of the AppSw to avoid wasting resources that are usually scarce. Nonetheless, resources are still statically allocated when the software starts the execution; usage of dynamic memory is not necessary.

6.4.2. Partition management

The partition management services of LithOS aim to control the internal states of its own partition, that is, to retrieve its own partition status and set the operation mode.

The services are designed to satisfy the partitioning constraints whereas the hypervisor (i.e. XNG) supports the temporal and spatial partitions isolation.

LithOS implements a set of services that permits to obtain the status of other partitions or set their operation mode. Thus, partitions in charge of the system monitoring and control can make use of these services. These services are not part of the ARINC-653 Standard and some of them are restricted to system partitions.

6.4.3. Process management

LithOS is a real-time operating system that provides execution threads called processes following the [RD01] Standard.

A process is a programming unit contained within a partition (they are not visible outside of the partition) which executes concurrently with other processes of the same partition to achieve the functionality and real-time requirements of the AppSw. Processes are scheduled periodically or aperiodically depending on their configuration at creation time.

LithOS provides services to create and control the execution of these processes.

6.4.4. Time management

Time management is an important feature of the real-time applications. LithOS has been designed to use the global time provided by the virtualization layer (XNG), with a granularity clock of 1 microsecond (μs). This global time is unique and independent of the partition execution within the system. All time values or capacities in a LithOS partition are related to this global time.

LithOS provides time control for process scheduling, deadline, periodicity, delays for process scheduling and time-outs for intra-partition and inter-partition communication.

6.4.5. Inter-partition communication

LithOS provides an interface to the AppSw for the communication between different partitions. This interface is in fact a set of wrapper functions for the XNG interpartition communication services.

The communication is performed through two kind of ports:

Sampling port: A message remains in the port until it is overwritten by a new occurrence of the message.

Queuing port: Messages are buffered. A message sent by a partition is stored in the message queue of the port until it is read by another partition. Messages are served in first in first out (FIFO) order by the virtualization layer. The buffer has a configurable size.

Communication ports allocated to the partition are observable and operable by all the processes within the partition.

6.4.6. Intra-partition communication

Processes within a partition are able to communicate or synchronize each other without the overhead of the inter-partition communication system. To do so, LithOS provides the mechanisms described in [RD01] for intra-partition communication and synchronization:

Semaphore: A synchronization mechanism used to control the access to shared resources.

Event: A synchronization mechanism which permits notification of an occurrence of a condition to processes which may wait for it.

Blackboard: A mechanism to communicate between processes. Any message written to a blackboard remains there until the message is either cleared or overwritten by a new instance of the message. Blackboards only store one message at a time.

Buffer: A mechanism to communicate between processes. Buffers are allowed to store multiple messages using a queuing discipline.

The amount of resources reserved for intra-partition communication and synchronization services is specified in the LTCF.

6.4.7. Health Monitor

XNG defines a HM service inspired in the ARINC-653 HM. The HM allows to define a Failure Detection, Isolation and Recovery (FDIR) policy specific for each AppSw.

Through the configuration file, the module and partition level errors can be associated to predefined actions.

In order to handle the process level errors, some of the events can be propagated to the faulty Partition. LithOS provides the services to install an error handler process which is in charge of handling the process level errors.

Besides, application based exceptions can be defined, raised and managed by using the corresponding services provided by LithOS.

LithOS HM is able to log the occurrences of HM events. The number of occurrences that can be logged is configurable in the LTCF. The LTCF also defines the default HM actions when certain HM events take place. Thus LithOS is able to perform an action if no error handler process has been created.

6.4.8. Multiple module schedule

LithOS implements the multiple module schedule defined in [RD02]. These services permit to extend the single and static module scheduler to several scheduling

6.4.9. Interrupt management

Hardware interrupt managing is hidden through *IrqEvent* facility. These service operate in a similar way as the event synchronisation mechanism does. Any process can request LithOS to suspend its execution in an interrupt request (IRQ) abstract object until it is set in UP state. When a IRQ is delivered to the partition, LithOS sets the abstract object in UP state and resumes the operation of the suspended processes.

6.4.10. System management

LithOS implements a set of services that permits to get the status of the virtualization layer and to perform a system reset or halt.

They are not part of the ARINC-653 Standard and all these services are restricted to system partitions.

6.5. Development tools

6.5.1. Gcc + binutils version (CG)

Cobham Gaisler provides toolchains for the NOEL-V processor and designs including this processor. The toolchain currently available is a GCC9.0 RTEMS 5 based toolchain. Bare-C toolchains will also be available for the introduction of the initial hardware of the De-RISC system.

6.5.2. GRMON debugger

NOEL-V/GRLIB systems are supported by Cobham Gaisler's GRMON3 debug monitor. GRMON3 also provides a socket for the GDB debugger. Since NOEL-V implements the RISC-V debug specification it is theoretically possible to use any RISC-V compliant debugger as long as the physical link to the NOEL-V system can be supported by the debugger.

6.5.3. *xcparser* tool

The configuration of the hypervisor can be seen as a two-steps process:

1. The System Integrator and the Partition's Developers agree on the allocation of resources to the hypervisor and partitions during the design of the system. As a result of this step XML-formatted XCF files are produced and shared among all parties.
2. When the hypervisor starts its execution, during the system development phase or during the operation phase, it expects to have accesses to the configuration values in binary format.

To close the gap between the two steps, the XML-formatted XCF files (represented as `XCF.xml`) can be translated to the a C counterpart (`XCF.c`), by means of the *xcparser*, and then to the final binary (`XCF.bin`) using the GCC compiler.

The *xcparser* tool is provided jointly with XNG to process the user-produced XML formatted XCF to generate the final binary that is used to configure XNG at initialisation.

7. Detailed sub-systems specification

DeRISC-T1.2-SW/SA-ID201	Dynamic memory allocation shall be forbidden
proposed	
Qualification method: operating system specification review (i.e. user manual), code inspection	
Rationale: dynamic features such as memory allocation whose failure (e.g. due to lack of memory available) cannot be explicitly controlled by critical applications may lead those applications to a failure	
Comment: validation tests may fail to reveal cases where sufficient memory exists if allocated dynamically since other applications may behave differently during operation, and coverage for all combinations of behavior across all applications at the same time cannot be reasonably generated with sufficient coverage. Therefore, dynamic memory allocation may jeopardize the reliability of the system and shall be avoided	

DeRISC-T1.5-HW/SA-ID202	The sub-systems shall avoid the use of dynamic features that may alter the functional or non-functional behavior of the platform
proposed	
Qualification method: platform specification review (i.e. user manual and technical reference manual)	
Rationale: Functional safety V&V processes rely on deterministic and repeatable processes, which implies that operation conditions can be enforced during V&V. Dynamic features that cannot be controlled or disabled explicitly by software means may behave differently at analysis and during operation, thus defeating the reliability of the V&V process	
Comment: Some such features that must be avoided are, for instance, dynamic frequency scaling (either with or without voltage scaling) if it is not explicitly controlled by software, as well as performance throttling upon a temperature alarm.	

DeRISC-T1.5-HW/SA-ID203	The platform shall have at least one critical configuration setting where all safety, security and performance requirements can be reasonably achieved
proposed	
Qualification method: platform specification review (i.e. user manual and technical reference manual), and testing campaigns with targeted software tests	
Rationale: The platform may allow some configuration settings where meeting other requirements is not possible, but at least one critical configuration setting shall allow meeting the other requirements. One such critical configuration setting shall be set for the platform if at least one application with some form of criticality needs to be executed	
Comment: The platform may offer, for instance, using shared caches with way partitioning across cores or fully shared. While the latter challenges bounding multicore interference, the use of way partitioning removes a high performance variability interference channel, thus easing the achievement of real-time requirements that might not be achievable otherwise	

DeRISC-T1.5-HW/SA-ID204	Critical configuration settings shall be configurable only in the appropriate privileged mode
proposed	
Qualification method: platform specification review (i.e. user manual and technical reference manual), and testing campaigns with targeted software tests	
Rationale: Since the fulfilment of some requirements directly depends on the critical configuration setting for the platform, such configuration must be set only with the appropriate privileges (e.g. by the hypervisor at boot time) avoiding any undesired modification that could easily create a mixed-criticality concern (e.g. a low-criticality application changing some configuration settings impacting the safety or security requirements of a higher criticality application)	
Comment: The critical configuration setting is typically expected to be set by the hypervisor at boot time so that, for instance, shared caches are space-partitioned across cores. During operation, applications shall not be allowed to alter such configuration and, in fact, it is very likely that even the hypervisor does not change those settings anymore	

7.1. Processor core

DeRISC-T1.2-HW/SW/PR-ID205	Core-local activities should not suffer interference from non-privileged applications running in other cores
proposed	
Qualification method: platform specification review (i.e. user manual and technical reference manual), and testing campaigns with targeted software tests	
Rationale: safety or security-related applications, particularly if they have any real-time or performance requirement, need to be proven to complete by specific deadlines. Interference is expected to occur in shared resources, but not in core-local ones. Thus, non-privileged applications (i.e. any application except the RTOS and the hypervisor) being executed in other cores should not have means to interfere execution directly in core-local resources of another application.	
<p>Comment: if not implemented properly, cache coherence mechanisms, cache inclusivity policies, etc, may alter core-local resources in a given core due to activities occurring in another core (e.g. cache line invalidations) despite those resources are intended to be managed only by the application executing in such core. Therefore, a combination of hardware support and software support is needed so that:</p> <p>(a) Remote cores cannot cause local cache evictions due to inclusivity. For instance, with inclusive shared L2 caches, core 1 could evict a line of core 2 in L2, thus causing a cascade eviction in an L1 cache of core 2.</p> <p>(b) Remote cores cannot cause local cache evictions due to coherence. This implies that data are not shared simultaneously across different applications. It may be shared simultaneously across threads of the same application, where joint timing analysis is a reasonable choice, or serially across different applications so that the one running later can cause remote evictions of already unused cache lines, thus with no effect on other applications.</p> <p>(c) Any other similar circumstance where core-local resources are interfered by remote unprivileged activity.</p> <p>Note that we may need both, hardware and software support to partition cache space, schedule applications, set appropriate communication mechanisms across applications, and the like.</p>	

DeRISC-T1.1-HW-ID1xx	The platform shall implement NOEL-V with GCN RISC-V extensions
proposed	
Qualification method:	
Rationale:	
Comment:	

DeRISC-T1.1-HW-ID1xx	The NOEL-V instantiation should implement the RISC-V B extension
proposed	
Qualification method:	
Rationale:	
Comment:	

7.1.1. MMU & privilege levels

7.1.1.1. Hypervisor support

Note from CG: There needs to be a discussion about the state of the H extension here.

DeRISC-T1.1-HW-ID005	The MMU shall allow the partitioning kernel to define the translation between 39 bit virtual memory addresses into 56 bit physical memory addresses.
proposed	
Qualification method: Review of the DeRISC hardware platform's documentation.	
Rationale: The hypervisor requires either a MPU or a MMU for implementing spatial partitioning.	
Comment: Supported virtual and physical memory address length defined according to SV39.	

DeRISC-T1.1-HW-ID006	<p>The MMU shall allow the partitioning kernel to configure for each page the attributes listed hereafter:</p> <ol style="list-style-type: none"> 1. Access privileged level permissions: supervisor or user privileged level. 2. Write permissions: read-only or read-write. 3. Execution permissions: execute or non-execute. 4. L1 cache policy: non-cached and write-through. 5. L2 cache policy: non-cached, write-through and write-back.
proposed	
Qualification method: Testing.	
Rationale: The hypervisor runs partitions in processor user mode so the pages assigned to partitions require to be accessible from such a mode. On the other hand, for avoiding a partition from accessing to hypervisor space, those pages assigned to the hypervisor are mapped in user mode.	
Comment:	

DeRISC-T1.1-HW-ID007	The MMU shall support pages of 4KB, 2MB and 1GB.
proposed	
Qualification method: Testing.	
Rationale: For saving TLB's entries, contiguous physical memory areas should be mapped with the largest supported page.	
Comment: Supported page sizes defined according to SV39.	

7.1.2. Core interface

7.1.2.1. Hypervisor and RTOS considerations (FEN)

DeRISC-T1.1-HW-ID008	The core shall conform to the RV64GC specification, including atomic, hardware multiplication and divide instructions.
proposed	
Qualification method: Testing.	
Rationale: Hardware architecture targeted by the DeRISC hardware platform. Note that atomic instructions is explicitly required by the software platform for supporting the SMP architecture.	
Comment:	

DeRISC-T1.1-HW-ID009	The core shall conform to the FDQ extensions from the RV specification.
proposed	
Qualification method: Testing	
Rationale: Partitions may require a FPU.	
Comment:	

7.2. Memory subsystem

DeRISC-T1.2-HW/PR-ID206	The memory subsystem shall avoid the use of arbitration policies with unbounded latencies for time-critical applications
proposed	
Qualification method: platform specification review (i.e. user manual and technical reference manual), and testing campaigns with targeted software tests	
Rationale: safety or security-related applications, particularly if they have any real-time or performance requirement, need to be proven to complete by specific deadlines. The existence of arbitration policies in the memory subsystem that may defer the processing of a request indefinitely may make those applications fail their requirements	
Comment: arbitration policies that may systematically defer the processing of any request in a shared cache or memory controller such as, for instance, priority-based arbitration, may systematically prioritize requests of a given application, thus postponing the processing of requests from other applications indefinitely. If those other applications may have any type of criticality, this may break the safety or security requirements of the system.	

DeRISC-T1.2-HW/PR-ID207	The memory subsystem should allow reasonably segregating space across applications
proposed	
Qualification method: platform specification review (i.e. user manual and technical reference manual), and testing campaigns with targeted software tests	
Rationale: safety or security-related applications, particularly if they have any real-time or performance requirement, need to be proven to complete by specific deadlines. Shared (cache) space across applications leads to mutual content evictions with arbitrarily high impact in execution time, which may make applications fail their timing requirements.	
Comment: cache partitioning across cores and comparable solutions are required for shared caches to avoid arbitrarily high interference. The alternative (shared space) imposes the need of joint timing analysis of applications sharing the cache, which in general may be too costly, and impose high V&V costs upon an update of a single application.	

7.2.1. Tightly coupled memory

DeRISC-T1.1-HW-ID300	A fast addressable on-chip memory shall be privately accessible for each core.
proposed	This may be a part of the L1 cache memory space, configured as addressable memory.
Qualification method: Inspection	
<p>Rationale: This allows decoupling the tasks prologue and epilogues, rich in memory accesses and inter-partition communications, and the main computation loop that could execute in isolation in a private memory. A global scheduling of the former minimises interference, while a local scheduling of the latter maximises parallel execution.</p> <p>cf. Girbal, S., Pérez, D. G., Le Rhun, J., Faugère, M., Pagetti, C., & Durrieu, G. (2015, September). A complete toolchain for an interference-free deployment of avionic applications on multi-core systems. In 2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC) (pp. 7A2-1). IEEE.</p>	
Comment: The size of this memory would ideally be large enough to contain all critical code.	

7.2.2. L1 cache memory

DeRISC-T1.1-HW-ID1xx	The L1 cache shall be protected against single-event upsets by using parity on cache data and tags.
proposed	
Qualification method:	
<p>Rationale: The L1 cache needs only error detection capabilities. Correct data can be fetched from L2 or main memory.</p>	
Comment:	

DeRISC-T1.1-HW-ID010	The L1 cache shall notify the software about parity errors through a synchronous event (i.e. processor exception).
proposed	
Qualification method: Testing.	
Rationale: The software platform must be aware of any L1 parity error so a mitigation action can be performed before such corrupted data or instruction could be processed.	
Comment:	

DeRISC-T1.1-HW-ID011	The L1 cache shall support the write-through cache policy.
proposed	
Qualification method: Testing.	
Rationale: Although the write-back cache policy may achieve better overall performance, it presents two drawbacks for critical systems: <ul style="list-style-type: none"> 1. The partition's context switch WCET is impacted by cache line evictions. 2. A parity error in a word not evicted yet can cause loss of information. 	
Comment:	

DeRISC-T1.1-HW-ID012	The L1 cache shall provide the partitioning kernel with an operation for invalidating the whole cache content.
proposed	
Qualification method: Testing.	
Rationale: The DeRISC software platform requires such an operation for guaranteeing cache coherency, specially during the cache initialisation and after switching the currently active MMU page table.	
Comment:	

DeRISC-T1.1-HW-ID013	The L1 cache shall provide the partitioning kernel with an operation for invalidating a specific cache line.
proposed	
Qualification method: Testing.	
Rationale: The DeRISC software platform requires such an operation for guaranteeing cache coherency, specially during the cache initialisation and after switching the currently active MMU page table.	
Comment:	

7.2.3. L2 cache memory

A L2 cache shared among several processor cores is usually the main shared resource causing timing interferences. The ability to partition the L2 cache per core is beneficial, but conflicts can still happen on the access port if it is also shared.

DeRISC-T1.1-HW-ID1xx	The L2 cache shall be protected against single-event upsets by using SECDED BCH code on internal data storage
proposed	
Qualification method:	
Rationale: The L2 cache must have error detection and correction abilities.	
Comment:	

DeRISC-T1.1-HW-ID014	The L2 cache shall notify the software about the ECC uncorrectable errors through a synchronous event (i.e. processor exception).
proposed	
Qualification method: Testing.	
Rationale: The software platform must be aware of any L2 ECC uncorrectable error so a mitigation action can be performed before such corrupted data or instruction could be processed.	
Comment:	

DeRISC-T1.1-HW-ID015	The L2 cache shall allow to configure whether ECC correctable errors are reported or not to the software.
proposed	
Qualification method: Testing.	
Rationale: Critical partitioned system usually implements a partition performing memory scrubbing. This way, the hypervisor is discharged from this overhead.	
Comment:	

DeRISC-T1.1-HW-ID016	The L2 cache should be able to self-correct ECC correctable errors without the intervention of the software.
proposed	
Qualification method: Testing.	
Rationale:	
Comment:	

DeRISC-T1.1-HW-ID016	The L2 cache shall support the write-through and the write-back cache policies.
proposed	
Qualification method: Testing.	
Rationale: The DeRISC software platform requires such an operation for guaranteeing cache coherency, specially during the cache initialisation and after switching the currently active MMU page table.	
Comment:	

DeRISC-T1.1-HW-ID017	The L2 cache shall provide the partitioning kernel with an operation for invalidating the whole cache content.
proposed	
Qualification method: Testing.	
Rationale: The DeRISC software platform requires such an operation for guaranteeing cache coherency, specially during the cache initialisation and after switching the currently active MMU page table.	
Comment:	

DeRISC-T1.1-HW-ID018	The L2 cache shall provide the partitioning kernel with an operation for invalidating a specific cache line.
proposed	
Qualification method: Testing.	
Rationale: The DeRISC software platform requires such an operation for guaranteeing cache coherency, specially during the cache initialisation and after switching the currently active MMU page table.	
Comment:	

DeRISC-T1.1-HW-ID019	The L2 cache shall provide the partitioning kernel with an operation for cleaning the whole cache content.
proposed	
Qualification method: Testing.	
Rationale: The DeRISC software platform requires such an operation for guaranteeing cache coherency, specially during the cache initialisation and after switching the currently active MMU page table.	
Comment:	

DeRISC-T1.1-HW-ID020	The L2 cache shall provide the partitioning kernel with an operation for cleaning a specific cache line.
proposed	
Qualification method: Testing.	
Rationale: The DeRISC software platform requires such an operation for guaranteeing cache coherency, specially during the cache initialisation and after switching the currently active MMU page table.	
Comment:	

7.2.4. External memory controller

7.2.4.1. Fault tolerance properties

DeRISC-T1.1-HW-ID1xx	The DDR2/3 SDRAM controller shall tolerate (maintain correct operation in the event of) loss of one external 8-bit wide memory device
proposed	
Qualification method: Demonstration	
Rationale: DRAM memory devices are often susceptible to SEFIs causing loss of one full memory device.	
Comment:	

7.2.4.2. Request arbitration

In order to maximize average performance, most non-critical external memory controllers (DDR-SDRAM) perform a re-ordering of access requests based on currently open memory page(s). This introduces an unpredictable latency in memory accesses.

7.2.5. Global memory mapping

TBD

7.3. Interconnect

DeRISC-T1.2-HW/PR-ID208	The interconnects shall avoid the use of arbitration policies with unbounded latencies for time-critical applications
proposed	
Qualification method: platform specification review (i.e. user manual and technical reference manual), and testing campaigns with targeted software tests	
Rationale: safety or security-related applications, particularly if they have any real-time or performance requirement, need to be proven to complete by specific deadlines. The existence of arbitration policies in the interconnects that may defer the processing of a request indefinitely may make those applications fail their requirements	
Comment: arbitration policies that may systematically defer the processing of any request in a shared bus or interface such as, for instance, priority-based arbitration, may systematically prioritize requests of a given application, thus postponing the processing of requests from other applications indefinitely. If those other applications may have any type of criticality, this may break the safety or security requirements of the system.	

7.3.1. L1 to L2

To be expanded: Multicore considerations, minimizing conflicts on shared cache level(s)

7.3.2. Main interconnect

In order to minimize timing interference, the main interconnect in the platform, linking cores with the main memory and high-bandwidth peripherals, should minimize interference channels, i.e. shared paths between different data transaction initiators and targets. A single bus is a major shared resource, as only one transaction is performed at a given time.

DeRISC-T1.5-HW-ID301	The main interconnect should provide quality of service mechanisms
proposed	
Qualification method: inspection	
Rationale: Depending on the application, different data paths in the SoC may be subject to a large traffic, while others may not. A configuration of quality of service on the main interconnect will allow adaptation to various scenarios.	
Comment:	

7.3.3. Cache coherency

DeRISC-T1.1-HW-ID021	The De-RISC hardware platform shall provide a mechanism (e.g. cache snoopers) for guaranteeing L2 cache coherence among cores.
proposed	
Qualification method: Testing.	
Rationale: In MP systems, it is required a mechanism for guaranteeing the coherence of all the L1 caches with respect to the L2 cache.	
Following scenario must be prevented:	
<ol style="list-style-type: none"> 1. Core 0 reads from Address0 => Content(Address0) loaded into core 0's L1 cache and L2 caches. 2. Core 1 writes to Address0 => Content(Address0) updated in core 1's L1 cache and L2 cache. 	
Core 0's L1 cache content(Address0) is invalid => cache coherency issue.	
Comment:	

7.3.3.1. Interference channels and isolation properties

To be expanded.

7.3.3.2. IPC (inter-partition communication) support

DeRISC-T1.2-HW-ID302	The platform shall provide inter-core interrupt notification mechanism
proposed	
Qualification method: demonstration	
Rationale: Inter-core “doorbell” interrupts are commonly used as a fast notification mechanism.	
Comment:	

7.3.4. IOMMU

7.3.4.1. Hypervisor support

DeRISC-T1.1-IO-ID022	The De-RISC hardware platform should provide an IOMMU.
proposed	
Qualification method: Testing.	
Rationale: An IOMMU may be needed for spatial partitioning at bus level since a partition may use DMA for accessing to physical memory areas without having access permissions.	
Comment:	

7.4. IO Peripherals

7.4.1. UARTs

DeRISC-T1.1-IO-ID023	The De-RISC hardware platform shall provide a configurable number of APB-UARTs.
proposed	
Qualification method: Testing.	
Rationale: Needed for easing the debugging and development of a system.	
Comment:	

7.4.2. Timers

DeRISC-T1.1-IO-ID024	The De-RISC hardware platform shall provide one GPTimer per core plus one additional GPTimer (i.e., at least, 5 GPTimers).
proposed	
Qualification method: Testing	
Rationale: XNG requires the GPTimers for: <ul style="list-style-type: none"> 1. Implementing a global system clock. 2. Implementing a system timer per core. 	
Comment:	

DeRISC-T1.1-IO-ID025	The GPTimer's counters shall support to be configured in cascade mode for implementing a 64 bits counter.
proposed	
Qualification method: Testing	
Rationale: Supporting a 64 bits counter discharges the hypervisor from maintaining a software system clock.	
Comment:	

DeRISC-T1.1-IO-ID026	The GPTimer's resolution shall be one μ second or lower.
proposed	
Qualification method: Testing.	
Rationale: XNG requires a clock resolution of one μ second or lower.	
Comment:	

7.4.3. Interrupt controller

DeRISC-T1.1-IO-ID027	The De-RISC hardware platform shall provide one interrupt controller per core (i.e., at least 5 interrupt controllers).
proposed	
Qualification method: Testing.	
Rationale: Each core must be able to process its own interrupts.	
Comment:	

DeRISC-T1.1-IO-ID028	The interrupt controller shall provide the partitioning kernel with the capability of generating inter-processor interrupts (IPIs).
proposed	
Qualification method: Testing.	
Rationale: XNG requires IPIs for synchronising multiple cores in an efficient way.	
Comment:	

DeRISC-T1.1-IFC-ID029	The interrupt controller shall provide the partitioning kernel with the capability of routing IRQ lines to a given core.
proposed	
Qualification method: Testing.	
Rationale: Each core should be able to just processing its own assigned IRQ lines.	
Comment:	

7.4.4. Ethernet

To be expanded.

7.4.5. DMAs

To be expanded.

7.4.6. SpaceWire

To be expanded.

7.5. Software architecture

7.5.1. Time and space partitioning

DeRISC-T1.5-SW-ID303	The platform shall support robust time and space partitioning of applications.
proposed	
Qualification method: TBD	
Rationale: Safety standards, notably in the avionics domain, require that multiple applications running on the same computer do not interfere with one another. A common means to attain this goal is the strict partitioning of computing resources between applications, in time, in memory-space and in processor-space.	
Comment: Memory-space partitioning is traditionally ensured by MPU or MMU modules. In a multicore context, processor-space partitioning is a new concept, where affinity of a task to a processor has to be enforced. This also encompasses the usage of other shared resources in the architecture, notably parts of the memory hierarchy and on-chip interconnect. Time partitioning is traditionally ensured by deterministic scheduling at the RTOS/Hypervisor level(s), but is also sensitive to delays caused by arbitration of concurrent access to shared resources (a.k.a. timing interferences).	

7.5.2. Deterministic partition scheduling

DeRISC-T1.2-SW-ID304	The hypervisor shall provide a deterministic scheduling policy at partition level.
proposed	
Qualification method: TBD	
Rationale:	
<comment>	

7.5.3. Explicit inter-partition communications

DeRISC-T1.2-SW-ID305	The hypervisor shall provide explicit communication mechanisms between partition
proposed	
Qualification method: inspection	
Rationale: typically ARINC653-like Queuing Ports and Sampling Ports	
Comment:	

DeRISC-T1.2-SW-ID306	The hypervisor shall provide a mechanism for quick processing of asynchronous events
proposed	
Qualification method: TBD	
Rationale: a cyclic executive scheme may present unacceptable latency for highly-critical asynchronous events.	
Comment:	

7.6. Debug and monitoring

7.6.1. Debug interface

To be expanded.

7.6.2. Performance and interference monitoring

DeRISC-T1.5-HW/SA-ID209	The monitoring features shall offer specific means to observe interference channels for validation and monitoring purposes
proposed	
Qualification method: platform specification review (i.e. user manual and technical reference manual), and testing campaigns with targeted software tests	
Rationale: safety or security-related applications, particularly if they have any real-time or performance requirement, need to be proven to complete by specific deadlines. Interference channels may exist in hardware shared resources. While appropriate system design and verification are mandatory to bound interference, tests are needed to validate that observed interference does not exceed expected levels. Therefore, appropriate observation means are needed to measure interference with sufficient detail to undergo certification and qualification processes needed	
Comment: performance monitoring units are key components to meet this objective. They shall offer appropriate means to be configured and deliver sufficient evidence to undergo the certification or qualification process within affordable cost bounds. For instance, providing information on how much interference each application causes on each other in each shared resource offers a sufficiently complete set of data for a successful validation process	

7.7. Reset and Boot process

The consortium plans to add a managed boot process to the De-RISC platform with support for secure boot (only authenticated software images will be started by the system). Requirements for this functionality is yet to be formulated.

8. Documentation

DeRISC-T1.2-HW/SA-ID210	Processor documentation shall be complete and precise enough to allow setting a critical configuration setting with reasonably high performance
proposed	
Qualification method: technical reference manual review	
Rationale: While hardware support is mandatory to configure the processor and allow its use with limited interference, high performance and sufficient means to allow V&V activities, appropriate processor documentation is needed to allow using those processor features.	
Comment: poorly documented features may remain unused, or may be used inappropriately, which overall defeats the whole purpose of setting them up.	

DeRISC-T1.2-SW/SA-ID211	Hypervisor documentation shall be complete and precise enough to allow exercising processor control knobs and features in a reliable and verifiable way
proposed	
Qualification method: user manual review	
Rationale: While hardware support is mandatory to configure the processor and allow its use with limited interference, high performance and sufficient means to allow V&V activities, appropriate hypervisor documentation is needed to allow configuring and using those processor features reliably and efficiently.	
Comment: poorly documented features may remain unused, or may be used inappropriately, which overall defeats the whole purpose of setting them up.	