De-RISC: A Complete RISC-V Based Space-Grade Platform

Nils-Johan Wessman[§], Fabio Malatesta[§], Stefano Ribes[§], Jan Andersson[§], Antonio García-Vilanova[¶], Miguel Masmano[¶], Vicente Nicolau[¶], Paco Gomez[¶], Jimmy Le Rhun[⋆], Sergi Alcaide[†], Guillem Cabo[†], Francisco Bas[†,‡], Pedro Benedicte[†], Fabio Mazzocchetti[†], Jaume Abella[†]

[§]CAES Gaisler, Sweden [¶]fentISS, Spain [⋆]Thales Research and Technology, France
[†]Barcelona Supercomputing Center (BSC), Spain [‡]Universitat Politecnica de Catalunya (UPC), Spain

***Abstract*—The H2020 EIC-FTI De-RISC project develops a RISC-V space-grade platform to jointly respond to several emerging, as well as longstanding needs in the space domain such as: (1) higher performance than that of monocore and basic multicore space-grade processors in the market; (2) access to an increasingly rich software ecosystem rather than sticking to the slowly fading SPARC and PowerPC-based ones; (3) freedom (or drastic reduction) of export and license restrictions imposed by commercial ISAs such as Arm; and (4) improved support for the design and validation of safety-related real-time applications, (5) being the platform with software qualified and hardware designed per established space industry standards.**

De-RISC partners have set up the different layers of the platform during the first phases of the project. However, they have recently boosted integration and assessment activities. This paper introduces the De-RISC space platform, presents recent progress such as enabling virtualization and software qualification, new MPSoC features, and use case deployment and evaluation, including a comparison against other commercial platforms. Finally, this paper introduces the ongoing activities that will lead to the hardware and fully qualified software platform at TRL8 on FPGA by September 2022.

I. Introduction

Increasing automation and autonomy of spacecraft demands higher performance for safety and mission-critical systems. Those systems must adhere to specific development processes to reach qualification for the highest integrity levels for space operation. So far, space industry has focused on developing products based on SPARC, Arm and PowerPC Instruction Set Architectures (ISAs). However, those ISAs bring either a continuous decrease in terms of software support, expensive licenses to design MPSoCs implementing those ISAs, and/or export restrictions challenging commercialization.

H2020 EIC-FTI De-RISC project [10] tackles these challenges holistically by developing a high-performance RISC-V multicore platform for the space domain building on the Xtratum New Generation (XNG) hypervisor and LithOS real-time operating system (RTOS) by fentISS, and the NOEL-V based MPSoC by CAES Gaisler, both of them based on RISC-V ISA [16]. Those key building blocks, which will be reaching TRL8 on FPGA by the end of 2022, are complemented by a multicore interference aware statistics unit by BSC, and key requirements and use case evaluation by Thales, a principal space end user and technology developer.

The De-RISC platform, which stands for **De**pendable **R**eal-time **I**nfrastructure for **S**afety-critical **C**omputer Systems, is, to the best of our knowledge, the first RISC-V based space-grade hardware and qualified software platform for space operation.

In this paper, we review the main features of the De-RISC hardware and software components, whose full details can be found in [19], and introduce the latest advances, which, when fully complete and validated, will allow the H2020 De-RISC platform reach commercial maturity. In particular, this paper introduces the following features:

- ECSS level B qualification of the XNG hypervisor;
- development kit for application design, integration and debug on the De-RISC platform;
- a powerful and configurable MPSoC designed per established space industry standards;
- extensions to the SafeSU statistics unit for multicore interference management;
- an evaluation with a space use case including a comparison against the GR740 space-grade MPSoC.

The rest of the paper is as follows. Section II presents relevant state-of-the-art. Section III introduces XNG and LithOS, the progress towards ECSS qualification, and the De-RISC development kit. Section IV presents the MPSoC and its latest features, including the SafeSU extensions. Section V introduces the use case and its evaluation. Finally, Section VI summarizes this paper.

II. State of the Art

This section reviews the state-of-the-art on relevant MPSoCs and hypervisors/RTOSs.

A. MPSoCs

The main European MPSoC providers for deep space missions – CAES Gaisler and Microchip (formerly Atmel) – build on the SPARC ISA. Gaisler produced the popular SPARC V7 ERC32 single core device. Atmel developed the AT697F device building on a LEON2FT core, but it lacked some key interfaces for the space domain such as SpaceWire and MIL-STD-1553B. Gaisler's GR712RC device includes both, a dual-core LEON3FT processor [6] more powerful than AT697F's LEON2FT one, as well as SpaceWire and MIL-STD-1553B interfaces. Later, Gaisler introduced the GR740 device, which includes a quad-core LEON4FT processor [7] and additional devices, hence outperforming previous devices.

US space-grade devices generally build on the PowerPC ISA such as BAE Systems RAD750 and RAD5545, and the DDC/Maxwell SCS750 board. Some of those devices have been used for the GAIA (Global Astrometric Interferometer for Astrophysics) mission from the European Space Agency (ESA). As shown in [9], the GR740 device also offers

978-3-9819263-6-1/DATE22/©2022 EDAA

the performance needed by GAIA. The De-RISC MPSoC inherits GR740 technology and improves it. Moreover, differently to the other space devices, it is the first one building on the RISC-V ISA.

There is a plethora of RISC-V cores available in the RISC-V International web portal [16]. However, those are not space-graded devices. The only relevant components to our knowledge are the SiFive's E76-MC embedded processor [18] and Gaisler's NOEL-V core [8]. The E76-MC includes some safety features, but misses many safety and reliability features needed for the space domain (e.g. watchdogs, domain-specific interfaces, etc.). The NOEL-V core, instead, is particularly devised for the space domain and is the core integrated in the De-RISC MPSoC.

B. Hypervisors and RTOSs

European space missions often rely on fentISS' XtratuM hypervisor [11], which provides temporal and spatial isolation, as needed for mixed-criticality applications in safety-related real-time systems. For instance, XtratuM has already been deployed in tens of satellites across different missions, and is also considered for *NewSpace* missions [20], which rely on hundreds or even thousands of light satellites, as opposed to conventional missions. The Air hypervisor, by GMV-Portugal has been proven adequate for space SoCs [13], is expected to be deployed as part of the INFANTE project, but there are no commercialization plans as per today. Sysgo's PikeOS [3], while also targeting the space domain, has not been deployed in any space mission so far to our knowledge.

US space missions may rely on LynxSecure hypervisor by Lynx Software Technologies, Wind River's hypervisor, and Green Hill's Integrity RTOS. Those hypervisors and RTOSs meet the main safety and security requirements of safety-related space applications.

However, only XtratuM has reached commercial maturity on RISC-V as part of De-RISC and is currently undergoing validation and qualification processes.

III. De-RISC Hypervisor

Within the software stack of De-RISC, the XtratuM Next Generation (XNG) hypervisor [11] and the ARINC-653 compatible LithOS run-time, both by fentISS, have been ported successfully to the hardware RISC-V architecture.

Making use of the mechanisms provided by the hardware, XNG provides time and space isolated execution environments, also known as partitions, and minimizes the interference between cores caused by the access to common resources.

Currently, XNG is being ported to support full virtualization, which replicates the underlying hardware behaviour to the runtime environment. The purpose of this is, for any operating system intended for stand-alone use, to successfully run inside a partition without needing to be modified. Implementation of full virtualization on XNG intends to take advantage of RISC-V's Hypervisor Standard Extension (H extension), which virtualizes the supervisor-level architecture to support the efficient hosting of guest operating systems atop a type-1 or type-2 hypervisor.

A. XtratuM Next Generation

The XtratuM Next Generation building blocks provide the services needed by safety-critical systems such as hypervisor and partition management, support for normal and privileged partitions, resource virtualization through the Partition Virtual Execution Environment (PVEE), temporal partitioning, spatial partitioning, inter-partition communication (IPC) mechanisms, a Health Monitor (HM) service which detects faults in the hardware and in XNG, observability of the system and the XNG Configuration Files (XCF), which are a set of XML files to allow the system integrator to configure the system.

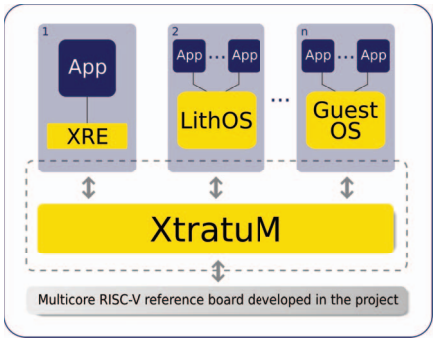

Fig. 1. XNG running on top of the De-RISC hardware platform.

Figure 1 shows a typical configuration with XNG running on top of the hardware platform. The block diagram illustrates a XNG-based system architecture with three partitions running on it. Partition 1 contains the XNG Run-time Environment (XRE), which allows the execution of bare metal applications over XNG without the need of any guest operating system, together with an application. The other two partitions (2 & 3) are managed by their corresponding guest operating systems (which can be LithOS or a third-party OS), executing their respective application(s).

B. LithOS

LithOS is an RTOS designed specifically to be executed inside an XNG partition in an IMA-based (Integrated Modular Avionics) system supporting space flight functions of a criticality classification of up to level B of ECSS.

Its purpose is to act as a guest OS providing services compliant with the APEX API defined by the ARINC-653 specification [1] and a subset of [2]. In the terminology used by the ARINC-653, LithOS and XNG should jointly be considered as the core software of an integrated module [1]. Some of the main LithOS features are:

- configuration of the maximum amount of resources that will be allocated to support the services provided by its ARINC-653 API;
- partition management services that aim at controlling the internal state of its own partition or other partitions;
- execution threads (called processes) which execute concurrently with other processes of the same partition in a periodic or an aperiodic way;
- time control for process management by using of the global time provided by the virtualization layer (XNG);
- intra-partition communication and synchronization mechanisms;
- inter-partition communication services;
- a Health Monitor (HM) service inspired in the ARINC-653 HM;
- multiple module schedule as defined in [2];
- hardware interrupt management;
- and system management services.

C. Testing automation

Since both XNG and LithOS are feature-rich and offer several functional capabilities, it is necessary to facilitate the conduction of test campaigns. Therefore, a highly automated validation environment has been developed for both products in the scope of De-RISC. Such environment provides several functionalities:

- compilation system with multi-architectural support;
- possibility of carrying out a multi-core execution whenever it is supported by the architecture;
- scripts for automatically adapting the hypervisor configurations for each target;
- scripts for automating the connection to debuggers and managing execution, as well as exception handling;
- and test suites under development composed by common and architectural-specific tests that will allow scalability to support further architectures in the future.

Several efforts are currently being made to implement continuous integration (CI) on both XNG and LithOS, thus facilitating the integration of source code changes by detecting potential defects at an early stage thanks to such automation of the software testing.

D. XNG development kit

An XNG development kit has also been ported to support the De-RISC architecture. It is divided in software development tools and integration tools. The development tools include:

- xcparser: A configuration tool to translate the XCF files containing the system resources description into a C file, used for configuring the hypervisor at runtime;
- elfbdr: A configuration tool for embedding multiple binary images in a single ELF (Executable and Linkable Format) file;
- xci: An observability tool for displaying the state of the hypervisor and the partitions when the system is suspended;
- xcon: An observability tool for displaying the contents of the hypervisor's console when a memory dump file containing the hypervisor's console data area is obtained;
- and xtraceviewer: An observability tool to filter the traces generated by the instrumented version of XNG from the output of the hypervisor's console and to translate them into human-readable traces.

Regarding the integration tools:

- XtratuM Project Manager (XPM): A framework for Eclipse that helps dealing with all the required files to build the partitions as well as for creating the XCF and the final system image;
- and Xoncrete: A tool to analyse the system schedulability and produce feasible, static schedules.

E. ECSS qualification

XNG porting has been carried out following the ECSS development process to ease the future space qualification at ECSS level B (critical severity). This will make the hypervisor ready for the aerospace market almost right after the project completion.

The documentation required for the ECSS qualification data package reflects the V-model development cycle that is being followed. It begins with the software development plan, followed by the software requirements and interfaces definition, the architectural and detailed design, and the software source code, which corresponds to the left side of the V-model development cycle.

The right side starts with the unit testing –against the source code–, the integration testing –against the design– and the validation testing[1] campaign – against software requirements. Unit, integration and validation test suite source code is also included in the qualification data package, as well as the corresponding plans and reports.

In parallel to the development process, the associated verification and quality assurance documents (plans and reports) are also included. At the end of the development, data package release information and a user manual to be provided to the customer are also generated.

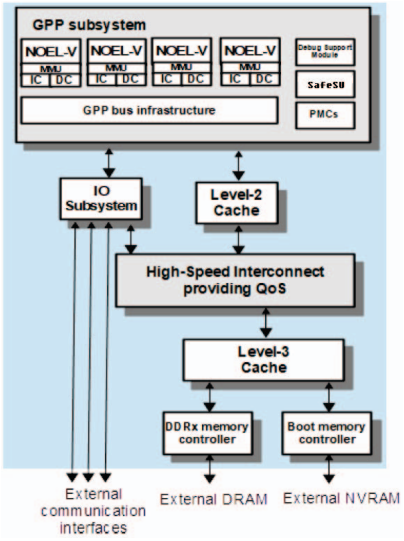

Fig. 2. De-RISC MPSoC

IV. De-RISC MPSoC

The De-RISC MPSoC architecture has been designed with a topology similar to system-on-chip architectures found in the commercial domain. The architecture has been defined to allow management of the MPSoC using an off-the-shelf operating system to control all processor clusters in the design. The architecture also allows partitioning of software instances onto individual clusters and separation of instances within one cluster. The intent is to provide a solution for general purpose payload processing applications and for centralized architectures where users may apply the SoC with mixed-criticality workloads such as platform functions, combined with a software GNSS receiver, combined with sensor data processing.

Applications will apply software, such as the XNG hypervisor, that runs in hypervisor mode on the processors. The hypervisor will host guest operating systems that can run on one or several processor cores and may also have a schedule where different partitions are active on the system over time. The controlling software will rely on hardware support to enforce partitioning between software instances. This hardware support consists of:

[1]Here, the term *validation testing* is used following ECSS terminology.

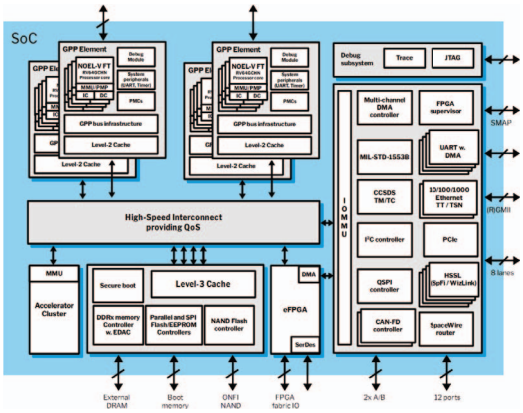

Fig. 3. De-RISC MPSoC

- Processor Physical Memory Protection registers – providing access control to memory areas without using the full functionality of a Memory Management Unit.
- Processor Memory Management Unit – Providing a virtual memory view to hypervisor guests and subprocesses
- IO Memory Management Unit – Providing access control and address translation for communication controllers that are capable of direct memory access

The architecture is built up from several building blocks:

- GPP Elements, consisting of general purpose processors, the standard configuration is to have four general purpose processing cores and system peripherals.
- IO subsystem, connecting bus interfaces of communication controllers that are capable of direct memory access, and putting these under control of the IO memory management unit.
- Memory subsystem, providing memory controllers behind a larger cache controller.

The De-RISC MPSoC includes one General-Purpose Processing (GPP) element, although the MPSoC is ready to include additional GPP elements if logic resources on the target technology allows for this. A GPP is a multicore in itself and, in the case of the De-RISC MPSoC, its GPP includes four NOEL-V RISC-V RV64GCH cores. Varying the number of GPPs and cores per GPP is possible, and ultimately the choice of the most efficient tradeoff depends on the target technology and expected application needs. The schematic of the De-RISC MPSoC is shown in Figure 2. Another application of the De-RISC SoC architecture is the GR7xV development, Figure 3, that targets an ASIC implementation, while the realization within the De-RISC project targets FPGA implementations.

As shown, the GPP cluster includes, apart from the four NOEL-V cores with their respective per-core caches (see Section IV-A), a bus infrastructure interfacing cores with components external to the cluster, such as the shared L2 cache and the IO subsystem (see Section IV-B).

Beyond the L2 cache, the De-RISC MPSoC includes a high-speed interconnect offering quality-of-service (QoS) support. This interconnect is intended to ease the extension of the MPSoC by attaching further GPP clusters and accelerators to it.

Finally, the aforementioned interconnect is attached to a shared L3 cache which, in turn, is connected to the DDR memory controller and to the boot memory controller.

Note that the MPSoC (mainly its peripherals) as well as the NOEL-V cores inherit the fault-tolerance support from the LEON processor family, thus allowing for seamless correct operation despite faults by correcting errors by hardware means. Such fault-tolerance support generally includes error detection and correction capabilities for all on-chip RAM memories, although specific details may be sensitive and need being directly requested to Gaisler. Upon the detection of an uncorrectable error, execution stops to avoid propagating it beyond the actual component affected.

A. NOEL-V RV64 processor core

The NOEL-V processor core is a 64-bit processor implementing the RISC-V ISA. Its pipeline is dual-issue in-order, and includes floating point units (supporting floats and doubles), four fully pipelined integer units (two of them in late stages to minimize stalls), support for integer multiplications and divisions, support for atomics, a Memory Management Unit (MMU), Physical Memory Protection (PMP), advanced branch prediction units, return address stack, and separate data (DL1) and instructions (IL1) first level cache memories whose size is configurable.

The cache controller includes a store buffer allowing back-to-back execution of store instructions, thus reaching a sustained throughput of one store instruction per cycle. The Advanced High-performance Bus (AHB) interface to connect to the GPP bus supports wide data transactions to allow for fast store data transmissions, and fast cache line refill.

Note that, apart from being integrated as part of the De-RISC MPSoC, the NOEL-V processor model is also provided in the Gaisler IP library (GRLIB). The GRLIB is an integrated set of IP cores that can be connected to the on-chip bus with an appropriate plug&play method and is available in a free open-source version.

B. Communication interfaces and peripherals

Usual peripherals such as timer units, system UARTs and interrupt controllers are included in the De-RISC MPSoC. Those follow standard specifications to guarantee portability across SoCs and software compatibility with drivers matching specifications.

The IO subsystem is architected in a modular way so that the particular IO interfaces implemented can be tailored to match application needs, thus varying the type and number of interfaces integrated. Those include standard interfaces, but also those specific for the space domain, and include the following ones:

- High-Speed Serial Link support through SpaceFibre controllers.
- SpaceWire communication links, connected to an on-chip router.
- 10/100/1000 Mbit Ethernet interfaces.
- MIL-STD-1553B support.
- Controller Area Network Flexible Data-Rate (CAN-FD) interface.
- UART interfaces with DMA support.
- SPI and I^2C master/slave, and GPIO interface.

Note that, except the parallel PCI interface, the De-RISC MPSoC includes all IO interfaces available in the GR740 microprocessor, including those that would be typically implemented in an external FPGA. In the case of the De-RISC MPSoC, all of them are included on-chip.

Finally, regarding the memory interface, the De-RISC MPSoC supports DDR3 SDRAM with a strong Error Detection and Correction (EDAC) code that tolerates even failures of complete external memory components. NOR and MRAM flash memory devices are supported for boot, which is also possible through the SPI interface. Additionally, NAND Flash memory is also supported as a means of having non-volatile memory storage, which is of particular importance in the space domain.

C. Extended SafeSU

The De-RISC MPSoC includes the SafeSU [4], a statistics unit implementing several features related to multicore interference validation, diagnostics and safety measures support. The SafeSU is conceived as an AMBA AHB-compatible module that, based on the signals observed from the different masters and slaves, can determine what master is using the shared interconnect and what others are made to wait, hence experiencing multicore contention. This information allows implementing features such as:

- Measuring the contention experienced by each master due to each other master [14], as needed for diagnostics in case of a deadline overrun.
- Measuring the maximum latency per request type [5], as needed for Worst-Case Execution Time (WCET) estimation.
- Setting multicore contention quotas [5], as needed to implement safety measures to avoid mixed-criticality concerns due to timing interference.

However, SafeSU's features are effective as long as contention occurs in the AHB bus. If the corresponding slave accepts requests and those are delayed in internal queues without keeping the AHB bus busy, then such contention is not exposed to the SafeSU. This is the case for part of the contention in the De-RISC MPSoC whenever write requests keep DRAM memory busy delaying further write requests. Hence, we are in the process of extending the SafeSU to monitor the L2-to-DRAM bus connecting the L2 cache with the DRAM controller, and detect when a request takes longer than the minimum service time to be served to report such interference.

V. Use Cases

The validation of the complete computing platform involves multiple steps, focusing on successively larger sets of features. Representative algorithms and applications are used to apply typical workload, and exert the dependability-oriented mechanisms notably for a space-grade context.

A. Performance Benchmarks

A first category of evaluations uses synthetic benchmarks and compute kernels, in order to assess the basic performance of the platform. Standard processor benchmarks such as Dhrystone and EEMBC Coremark have been ported on the NOEL-V core, providing figures of 2.82 DMIPS/MHz and 4.41 Coremark/MHz respectively. These standard benchmarks are only useful to assess the core performance, as they fit in the first level of cache memory.

Several compute kernels have also been ported in both bare-metal and with XNG hypervisor using the simple XRE execution environment. For example, the matrix multiplication benchmark will make use of the different levels of the memory hierarchy depending on the size of the matrices. The results are shown in the table below. Please note that the current platform uses the area-efficient NanoFPU, and a fully-pipelined FPU (currently in validation) is likely to improve results on benchmarks such as FFT. Results for an Arm A53 core have been included normalized to the operating frequency of the Xilinx VCU118 FPGA where the De-RISC MPSoC has been synthesized. Details on matmul trends are currently under investigation (i.e. significant execution time increase for matmul small and no increase for matmul large). Our current hypotheses relate to cache interference altering access ordering.

Execution time in ms			
Kernel	NOEL-V bare-metal	NOEL-V XNG	ARM A53 (normalized)
matmul small	3104	6930	2572
matmul large	83819	83668	79917
fft	N/A	8522	1229
sort	7387	8015	7311
sha-1	1115	1775	1038

B. Command and Data Handling platform

This use-case is representative of an on-board satellite subsystem with multiple communicating partitions. It is derived from the application used in the ECSEL EMC^2 project to evaluate the LEON4FT space-grade microprocessor together with XtratuM hypervisor [15]. The focus is on the inter-partition communication in the XNG environment. Telecommand and telemeasure messages are exchanged between partitions, using a standard protocol, over Queuing Ports provided by XNG. In addition, floating-point computations representative of satellite attitude control are executed, involving quaternion computation.

Preliminary results comparing NOEL-V with XNG and LEON4 with XtratuM show a similar telecommand message de-queueing time (with an average measurement of 57µs), whereas enqueuing took 30% shorter time on the De-RISC platform. Further experiments are needed to obtain better statistical significance, and assess the floating-point computation.

C. Flight software

For higher level of validation, a more complete use-case is currently being integrated. It is based on the LVCUGEN framework and the CCSDS-123 hyperspectral compression algorithm, forming a fully mixed-critical space system.

LVCUGEN (Logiciel de Vol Charge Utile GENérique, or generic payload flight software) [12] is developed by CNES, the French space agency, to facilitate the integration of payload software by providing time and space partitioning based on XNG and LithOS, system services such as IO server, mode management, data loading health monitoring and a telemeasure/telecommand communication library. This ensures a good representativeness of the use-case, in terms of partitioning, resource usage, scheduling, and space-grade coding practices.

As a representative data-intensive application, we chose the most recent standard for space-grade image compression, CCSDS-123 [17]. This algorithm performs lossless compression of hyper-spectral images, i.e. with a large number of wavelength bands. The raw data can therefore be quite large and with a three-dimensional organization. The algorithm is optimized for this multi-dimensional locality

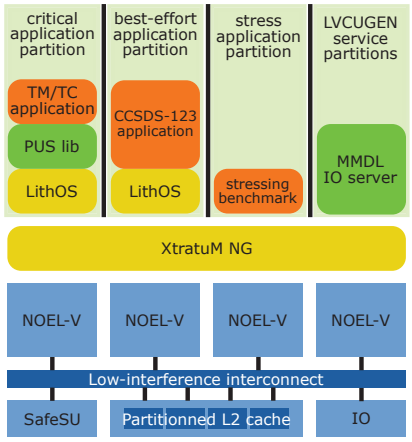

Fig. 4. Flight Software use-case configuration

with a special adaptive predictor, and performs a propagation of intermediate results. The data throughput of the application scales with the image size, making it a good candidate to exert the De-RISC platform with a typical data-intensive workload.

Several deployments will be possible on the multi-core platform, one example is illustrated in Figure 4. One partition, based on LithOS, hosts a critical application, sending and receiving telemeasures and telecommands. Another partition also based on LithOS hosts the CCSDS-123 algorithm, acting as a representative compute-intensive load on the platform resources, including the memory hierarchy. A system partition is dedicated to framework services such as Mode Management and Data Loading (MMDL) and IO server. In addition, supplementary stressing benchmarks can be used in some configurations to put pressure on specific resources.

To assess the effectiveness of time and space partitioning and the associated interference-mitigation mechanisms, we will measure the (lack of) effect of the compute-intensive and stress partitions upon the behaviour of the critical partition, using the capabilities of the SafeSU module.

VI. Summary

The increasing need for performance, reconfigurability, and a richer software ecosystem in the space domain brings new opportunities for the development of appropriate platforms. H2020 De-RISC tackles all those challenges holistically by developing the first space-grade MPSoC and qualified hypervisor based on the RISC-V ISA. In particular, the De-RISC platform consists of (1) fentISS' XNG ECSS level B qualified hypervisor and LithOS RTOS, (2) a high-performance NOEL-V based MPSoC by CAES Gaisler, (3) including a multicore interference aware statistics unit (SafeSU) by BSC, being all of them designed meeting (4) Thales requirements and use case KPIs.

In this paper, we have presented the latest progress, which includes XNG qualification efforts, development kit deployed, MPSoC features, SafeSU extensions, and use case evaluation and comparison against other products in the market. The completion of those items and their validation will allow reaching TRL8 by the end of 2022. The hypervisor and MPSoC will be available under commercial licenses. A number of MPSoC components will be part of Gaisler's GRLIB offered as open source under GPL license. Finally, the SafeSU, which is already offered as open source (MIT license), will be conveniently upgraded.

Acknowledgements

This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement EIC-FTI 869945. BSC work has also been partially supported by the Spanish Ministry of Science and Innovation under grant PID2019-107255GB-C21/AEI/10.13039/501100011033.

References

[1] Aeronautic Radio, Inc. *Avionics Application Software Standard Interface. Part 1 - Required Services. ARINC Specification 653P1-3*, 2010.

[2] Aeronautic Radio, Inc. *Avionics Application Software Standard Interface. Part 2 - Extended Services. ARINC Specification 653P2-2*, 2010.

[3] J. Bredereke. A survey of time and space partitioning for space avionics. *Technical Report, City University of Applied Sciences Bremen*, 2017.

[4] G. Cabo, F. Bas, R. Lorenzo, D. Trilla, S. Alcaide, M. Moretó, C. Hernández, and J. Abella. Safesu: an extended statistics unit for multicore timing interference. In *2021 IEEE European Test Symposium (ETS)*, 2021.

[5] J. Cardona, C. Hernandez, J. Abella, and F. J. Cazorla. Maximum-contention control unit (mccu): Resource access count and contention time enforcement. In *2019 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 710–715, March 2019.

[6] Cobham Gaisler. LEON3FT Fault-tolerant processor. https://www.gaisler.com/index.php/products/processors/leon3ft (accessed Feb-2021).

[7] Cobham Gaisler. LEON4 processor. https://www.gaisler.com/index.php/products/processors/leon4ft (accessed Feb-2021).

[8] Cobham Gaisler. NOEL-V processor. https://www.gaisler.com/index.php/products/processors/noel-v (accessed Feb-2021).

[9] Cobham Gaisler. RTEMS SMP executive summary, development environment for future leon multi-core. *RTEMSSMP-ES-001*, 2, 2015. http://microelectronics.esa.int/gr740/RTEMS-SMP-ExecSummary-CGAislerASD-OAR.pdf.

[10] De-RISC Consortium. De-RISC website, 2021. https://www.derisc-project.eu/ (accessed Feb-2021).

[11] fentISS. XtratuM Hypervisor. https://fentiss.com/products/hypervisor/ (accessed Feb-2021).

[12] J. Galizzi, P. Arberet, J.C. Damery, C. Guy, A. Crespo, M. Masmano, and F. Roubert. LVCUGEN- Ready for Flight? In L. Ouwehand, editor, *DASIA 2015 - DAta Systems in Aerospace*, volume 732 of *ESA Special Publication*, September 2015.

[13] B. Gomes, D. Silveira, L. Gouveia, and L. Mendes. Air hypervisor using RTEMS SMP. In *European Workshop on On-Board Data Processing (OBDP2019), ESTEC (ESA)*, 2019.

[14] J. Jalle et al. Contention-aware performance monitoring counter support for real-time MPSoCs. In *IEEE Symposium on Industrial Embedded Systems (SIES)*, 2016.

[15] L. Pomante, D. Andreetti, F. Federici, V. Muttillo, and D. Pascucci. Analysis and design of a command & data handling platform based on the leon4 multicore processor and pikeos hypervisor. *DAta Systems In Aerospace (DASIA)*, 2017.

[16] RISC-V International. RISC-V International website. https://riscv.org/.

[17] L. Santos, A. Gomez, and R. Sarmiento. Implementation of ccsds standards for lossless multispectral and hyperspectral satellite image compression. *IEEE Transactions on Aerospace and Electronic Systems*, 07 2019.

[18] SiFive Inc. SiFive E76-MC Manual v19.08p0, 2019. https://sifive.cdn.prismic.io/sifive%2F08e49813-ffcc-4a6c-9d70-ba2add7ebbe6_sifive+e76-mc+manual+v19.08.pdf (accessed Feb-2021).

[19] N.J. Wessman, F. Malatesta, J. Andersson, P. Gomez, M. Masmano, V. Nicolau, J. Le Rhun, G. Cabo, F. Bas, R. Lorenzo, O. Sala, D. Trilla, and J. Abella. De-risc: the first risc-v space-grade platform for safety-critical systems. In *2021 IEEE Space Computing Conference (SCC)*, 2021.

[20] Wikipedia. Small satellite, 2021. https://en.wikipedia.org/wiki/Small_satellite (accessed Feb-2021).