EUROPEAN WORKSHOP ON ON-BOARD DATA PROCESSING (OBDP2021), 14-17 JUNE 2021

DE-RISC: LAUNCHING RISC-V INTO SPACE

Jimmy Le Rhun[1], Vicente Nicolau[2], Antonio Garcia-Vilanova[2], Jan Andersson[3], and Sergi Alcaide[4]

[1]*Thales Research & Technology, Palaiseau, France*
[2]*fentISS, Valencia, Spain*
[3]*Cobham Gaisler, Goteborg, Sweden*
[4]*Barcelona Supercomputing Center, Barcelona, Spain*

ABSTRACT

An key challenge faced by mission-critical computers is the ability to scale the processing performance, while maintaining a high level of dependability in a harsh environment. The adoption of COTS multi-core processors poses difficulties both in terms of timing interference and reliability under thermal and radiation stress. Specific dependability-related features are thus required for space computers. The recent rise in popularity of open-source hardware is a great opportunity to push the concept of fault-tolerant, deterministic computing platform. The De-RISC project aims at providing the first complete processing platform for aerospace, leveraging RISC-V cores, state-of-the-art hypervisor technology and advanced monitoring techniques. The current status of the project is in line with the plans: the prototype platform is already functional and almost complete, with new features added in scheduled internal releases. The validation phase has started, and will proceed incrementally until the end of the project. The commercial release of the platform is expected for Q2 2022.

Key words: open hardware; RISC-V; aerospace; dependability; hypervisor; realtime monitoring.

1. INTRODUCTION

In the domain of mission- and safety-critical computers, multicore COTS processors have been recently considered as an interesting evolution, as they could match the ever increasing performance requirements of next-generation avionics and space systems.

However, these architectures are characterized by shared hardware resources, notably within the memory hierarchy and interconnect, that are subject to the phenomenon of timing interference. As a core performs a transaction with a shared resource, hardware arbitration mechanisms delay any concurrent access from other cores to the same resource, resulting in hard-to-predict worst-case execution times.

On the other hand, the strict requirements for timeliness of critical applications, and the associated safety standards, mandate the identification and mitigation of interference channels. This task can be difficult for COTS processors, as the documentation is often not precise enough. Most generic COTS platforms also lack the specific mechanisms for fault-tolerance and robustness in a harsh environment.

The recent rise in popularity of open-source hardware systems, notably the RISC-V processor architecture, offers the opportunity to "open the box", and tailor the hardware and software components of a computing platform to improve the support of dependability requirements.

In this paper, we present the progress of the De-RISC (Dependable Real-time Infrastructure for Safety-critical Computers) H2020 Fast Track to Innovation project. The goal is to provide a high-TRL dependable computing platform composed of a fault-tolerant multicore system-on-chip design on FPGA and a space-grade hypervisor and RTOS, with specific features for interference monitoring and mitigation.

In the next section we cover a brief state-of-the-art, and in section 3 we detail the elements of the De-RISC platform. Section 4 gives an overview of the validation activities, and finally we discuss in section 5 the current status of the project.

2. STATE OF THE ART

Mission-critical computers, especially in the aerospace domain, require specific techniques to meet their very stringent dependability requirements, including robustness to harsh environment (extreme temperatures, vibrations, cosmic radiation, etc.) and real-time determinism (in order to guarantee that the deadline of critical tasks will be met).

Most commercial off-the-shelf components are optimized for best average performance, in relatively mild environments. To reach the required performance, they rely on multi-core architectures. However, this average performance gain often comes from speculative or generally non-deterministic mechanisms, that negatively impact the worst-case performance that matters in critical systems.

Radiation tolerance is a defining characteristic of processor destined for deep space missions, and to a lesser extent of those targeting low earth orbit and aviation.

2.1. Processor architectures for aerospace

The PowerPC architecture is widely used in avionics, and in the US space market. For the former, products designed for the communication market are popular, but those lack strong fault-tolerance mechanisms and multicore versions exhibit timing interference issues [11]. For the latter, devices such as the RAD750 and RAD5545 from BAE Systems represent the high performance devices currently available. In addition to being a proprietary instruction set architecture, space-grade PowerPC devices are subject of US export control restrictions.

As an alternative, the SPARC architecture was used to develop the European success story of space-grade processors, the series of LEON cores [1] by Cobham Gaisler. The relative openness of the SPARC instruction set architecture allowed the LEON cores to be proposed both under an open-source licence, and the fault-tolerant version under a commercial licence. In addition, radiation-hardened System-on-Chip devices are available, including the GR712RC based on two LEON3FT cores and the GR740 [4] using four LEON4FT cores. The open-source implementations allowed a validation by a wide user base, which proved beneficial for improved reliability.

There are also efforts to develop ARM-based System-on-Chip for the space domain, such as the H2020 DAHLIA European project, but the proprietary instruction set may limit the ability to support novel dependability mechanisms.

Recently, the RISC-V [2] open-source instruction set architecture gained a great momentum both in research institutions and industry, as it proposes modularity, good efficiency, a common software ecosystem, and opportunity for customization. There are many implementations, either open-source or commercial, targeting different classes of computers, from small IoT devices to large supercomputers. For example, popular implementations include Rocket from UC Berkeley and SiFive, CV32E40/Ri5cy and CV64A6/Ariane from ETH Zürich and the Open Hardware Group.

However, there was no implementation specifically targeting safety-critical, fault-tolerant systems at the start of the De-RISC project. Thanks to the experience with LEON-FT, the RISC-V implementation by Cobham Gaisler named NOEL-V is the best candidate to be the first to integrate these features.

2.2. Space-grade hypervisors

The increase in complexity of critical systems, notably the integration of multiple independant applications on the same computer, led to the adoption of hypervisors or partitioning kernels to manage time and space partitioning of applications. This is particularly the case in aeronautics, with the Integrated Modular Avionics (IMA) [9] concept along with the ARINC-653 standard for RTOS API.

In Europe, the main providers of suitable hypervisors for aero and space systems are Sysgo with PikeOS, both an hypervisor and RTOS with optional ARINC-653 interface, and FentISS with XtratuM hypervisor and LithOS operating system. PikeOS has been deployed on the first ANGELS (Argos Neo on a Generic Economical and Light Satellite) mission [10], whereas XtratuM is already onboard 76 satellites of three different missions. US-based hypervisor technolgies are also used in space, such as WindRiver hypervisor, LynxSecure separation kernel, and Green Hill's Integrity RTOS.

In terms of support for RISC-V architecture, only XtratuM is already attaining commercial maturity as part of the De-RISC project presented in this paper.

3. DE-RISC PROJECT

De-RISC [7] is an H2020 project from the EIC-FTI initiative (project EIC-FTI 869945) aimed at developing a space-grade hardware and software platform based on the RISC-V open architecture (hardware) and an Integrated Modular Avionics architecture (software) based on strict time and space partitioning. The hardware architecture is developed by Cobham Gaisler using their NOEL-V open source multicore design whereas the software architecture is developed by fentISS using their XtratuM Next Generation (XNG) hypervisor. Advanced hardware features such as performance monitoring support are designed and integrated in the platform by the Barcelona Supercomputing Center (BSC). The hardware design will be completed by providing a space-grade development board with a compact PCI serial space interface and featuring a Xilinx XCKU060 FPGA together with GR716B rad-hard microcontroller. Finally, Thales Research validates the technology developed in the project using two use cases, a Command and Data-Handling Application previously used in the EMC2 project [8], [12]; and a generic on-board data handling framework (LVCUGEN [14]) provided by CNES to ease the development of complex instruments and of nanosatellites.

3.1. Hardware architecture

The architecture of the De-RISC SoC platform is based on lessons learned and feedback from users of the radiation-hard fault-tolerant GR740 multicore processor

[4] that has been developed in a series of activities initiated by ESA within the Next Generation Microprocessor development.

Figure 1 provides an overview of the De-RISC SoC architecture. NOEL-V is Cobham Gaisler processor implementation of RISC-V [3]. Four processors are organized in a multicore cluster (GPP element) connected to a shared Level-2 cache. Each GPP element is a multi-core processor system that also includes standard peripherals such as interrupt controller and a console UART.

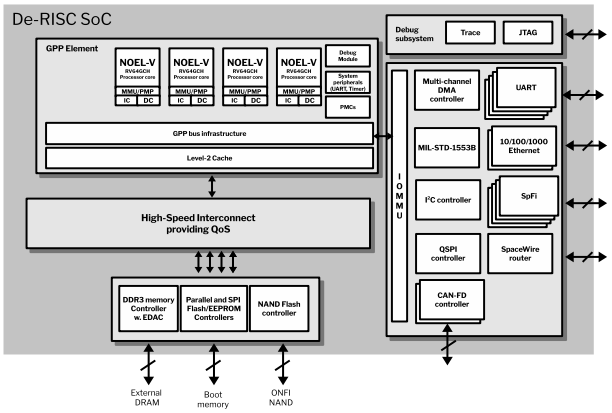

Figure 1. De-RISC SoC Design

The processor configuration applied in De-RISC is an RISC-V RV64GCH instruction set architecture. This means that the architecture has RISC-V 64-bit core implementing the following features:

- RV64I 64-bit Base integer instructions (RV64I)
- MUL/DIV (M)
- Atomics (A)
- Single/Double Float (FD)
- Compressed instructions (C)
- Hardware hypervisor support (H)
- Supported modes: M/S/U
- MMU (Sv39/Sv32)
- Physical Memory Protection (PMP)
- 16 KiB I-cache, 16 KiB D-cache
- Optional Tightly Coupled Memory

The SoC contains an IO subsystem with communication controllers capable of direct memory access (DMA). These are connected to the larger system through an IOMMU that provides address translation and access protection. The default set of communication controllers include SpaceFibre links, dual gigabit Ethernet MACs, dual redundant CAN-FD interfaces, SpaceWire links and other lower speed interfaces.

The memory subsystem contains a DDR3 SDRAM controller providing primary memory and memory controllers to provide access to non-volatile memory via NAND, SPI and legacy parallel interfaces.

Figure 1 and the description above shows the default configuration of the SoC architecture applied within De-RISC. The architecture is scalable and the number of processors within each processor cluster is configurable both in terms of number of processors and also processor configuration. In addition to this, the number of clusters is also configurable. Applying one cluster with four processor cores is possible within using the project's target FPGA which is the Xilinx Kintex Ultrascale 060. Subject to mission needs, radiation mitigation measures may need to be applied to the design leading to a shortage of FPGA resources and the number of processor cores, or the feature set implemented by each processor core, can then need to be reduced.

Figure 2 provides a general view of the scalable SoC in a configuration foreseen for future application specific standard product implementations. In this configuration several processor clusters are implemented and a third cache hierarchy is introduced where the processing elements are connected with the input/output elements through an IOMMU. An accelerator cluster and an embedded FPGA are foreseen as part of a standard product implementation.

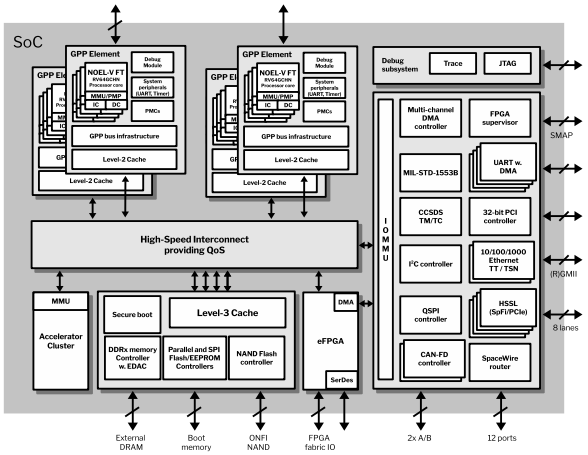

Figure 2. Standard product configuration of SoC platform

Regardless of the SoC topology applied, several hardware features are available to support mixed-criticality workloads. The approach taken in De-RISC is to support safety-critical applications using the general-purpose processing core instead of dedicating real-time cores in a real-time subsystem. This is achieved by introducing features such as tightly coupled memory to the processor cores, supporting configurations of the GPP elements where isolated communication paths are created between

the processors and Level-2 cache, and updates to cache replacement policies and coherency protocols to avoid interference from other parts of the system on core local activities.

The De-RISC project also includes the development of a PCB board that has been designed to be compatible with the CompactPCI Serial Space backplane standard (CPCI-S.1 R1.0) and will be available as the product GR-CPCIS-XCKU from Cobham Gaisler towards the end of 2021. The board applies a Xilinx XCKU060 FPGA and supports use of all the communication interfaces of the De-RISC SoC with the exception of MIL-STD-1553B.

3.2. Software infrastructure

The system software controlling the platform is based on the XtratuM (XNG) hypervisor [5] and the guest OS LithOS [6]. Other guest operating systems for XNG such as RTEMS and Linux will also be ported to the De-RISC platform. The figure 3 shows a block diagram of XNG and three partitions running a bare-metal partition, LithOS and some other guest operating system, such as RTEMS or Linux.

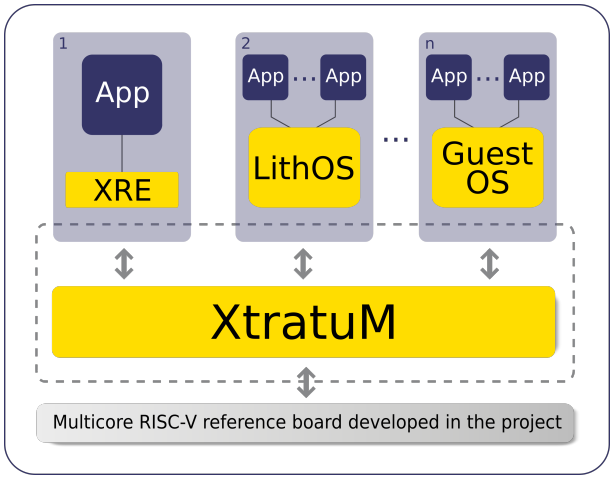

Figure 3. XtratuM-based system with XRE, LithOS and other guest OS as partitions run-time environments

XNG internal software architecture is divided in three parts. The common part implements the architecture independent functions of the hypervisor. A second architecture-dependant part implements the functions which depend on the underlying processor architecture. Finally, a board dependent part implements the details related to the target board and I/O peripherals in which the system is running.

XNG partitions are the basic space segregation units. Each partition is presented with a partition virtual execution environment (PVEE) per core. A PVEE implements the elements providing the functionality equivalent to their hardware counterparts. XNG virtualizes the following elements for each processor core: the vCpu which represents the executing processor; the vFpu which represents the floating point unit; the vIrqLCtrl representing each processor local interrupt controller; the sysTimer and execTimer provide each vCpu of a partition with real-time timers (incremented with the system clock) and execution timers (incremented with the execution clock). Other virtual devices such the virtual interrupt distributor, the system clock and the execution clock are global to a partition (figure 4).

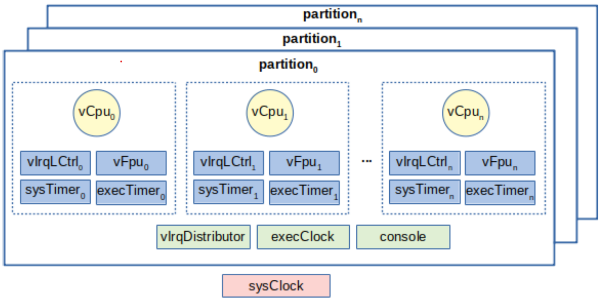

Figure 4. PVEE virtual devices

The initial version of XNG used for the porting activity was the monocore version running on the Xilinx Zynq7000 MPSoCs and qualified to ECSS level B on a previous CNES project [5]. The first step was to extend the version to multicore by providing spin-locks for atomic operations and other required mechanisms for multicore architectures. The new XNG multicore (XNG/SMP) was ported to the GR740 which shares peripherals and development environment with the NOEL-V platform. This strategy proved to be successful and allowed a fast porting to the NOEL-V platform. Some iterations have also taken place to refine the implementation and have it ready for validation. In the next project phase, XNG will be extended to support full virtualization using the RISC-V H extension developed by Cobham Gaisler for NOEL-V.

3.3. Monitoring

The De-RISC platform implements advanced monitoring techniques that help to ensure the real-time behaviour in a multicore context. More precisely, the novel Safe Statistics Unit (SafeSU for short [18]) is included in the SoC. This module provides monitoring and control of the multicore interference inside the platform, which is of prominent importance for the adherence to real-time safety requirements and for their verification and validation.

The SafeSU is composed of 3 main components:

- Request Duration Counter [16]: Provides end users with an observability channel to bound multicore interference during verification stages.
- Cycle Contention Stack [17]: Offers observability features by measuring multicore interference during testing, as part of the validation stages.

- Maximum-Contention Control Unit [16]: This controllability feature allows monitoring and controlling multicore interference by means of quotas during operation.

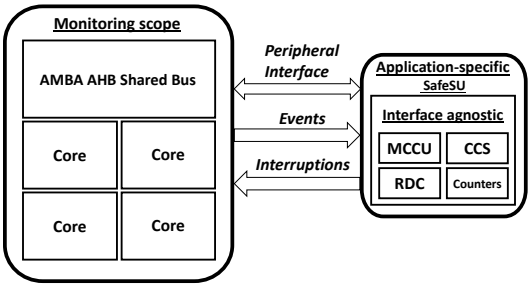

Figure 5. SafeSU integration in the De-RISC Platform

Figure 5 shows the integration of the SafeSU into the De-RISC SoC. The SafeSU is connected to the AMBA AHB shared bus as a slave and monitors the transactions and the owner of the bus each cycle. Events such as cache misses or retired instructions are passed to the SafeSU by the cores, and the SafeSU sends back interrupts.

Request Duration Counter (RDC) [16]: The RDC is in charge of measuring the highest latency occurred for different types of accesses inside the AHB component (e.g. a bus) it is attached to. If combined with a test campaign of specifically designed stressing tests, the RDC can allow estimating the highest latency for each type of event. For example, in the De-RISC platform, maximum read or write latency operations can be bound after cache accesses missing and requiring to access main memory. These highest latencies can be later used for the estimation of the Worst-Case Execution Time (WCET) to obtain reliable upper bounds to the execution time of real-time tasks during the verification stage. For instance, execution time measurements can be padded with the RDC latencies per bus access to account for the potential impact of interference during operation.

Furthermore, the RDC module has been extended to include default programmed values (e.g., the highest latencies observed per each event type) and be used during operation as a safety monitor that can trigger an interrupt if an event latency exceeds the pre-defined programmed value. This is relevant during operation to identify any risk of violating a deadline.

Cycle Contention Stack (CCS) [17]: The CCS is used as a "blaming" mechanism since it measures the interference experienced by each core and breaks it down across the actors (e.g. cores) that have caused it. The CSS is implemented as an NxN table of counters, where N is the number of actors in the bus it is attached to (4 cores in the De-RISC platform, 16 counters).

The CCS can be useful during both, operation and validation. During validation stages, the CCS can assess whether pairwise interference is within expected bounds. Without the breakdown, high interference from one core could go unnoticed if the other cores produced low interference, letting undesired behavior without detection. During operation, can be of great help to implement safety measures since, when a deadline overrun happens, it allows identifying the cores responsible of causing the disproportionate interference, and allows further correction actions to be taken appropriately (e.g. penalizing the "blamed" core in the scheduling algorithm).

Maximum-Contention Control Unit (MCCU) [16]: The MCCU organization, similarly to that of the CCS, contains NxN counters. Instead, the MCCU mission is not to measure interference but rather only monitoring that interference does not exceeds predefined bounds. By modifying its configuration registers, users can define the maximum interference quota that each core can induce into each other. Once a particular quota has been exceeded (or it is about to), the MCCU raises an interrupt, which allows software layers to take appropriate corrective actions. In mixed-criticality scenarios, this is a very relevant feature since tasks with different levels of criticality compete for resources, and the most critical tasks must have more advantages with respect to lower criticality tasks to guarantee their completion.

The MCCU can be useful into different modes, each one arranged for a different point of the product lifecycle. The first one, upon an event from a core c_i that creates interference into core c_j, the MCCU counter specifics to the "offending" core is updated subtracting the maximum latency that this type of event have been recorded (e.g. given by the RDC). This case can be employed at design stage since interference events are assumed to create the maximum but still realistic interference possible. Thus, this use of the MCCU can help indicating whether deadline violations are possible. The other operation mode, instead of using the maximum latency observed, builds on the actual interference (e.g. recorded by the CCS) to decrease the quotas, entrusting the MCCU to be used as a safety measure during operation. Interrupts are only raised when interference has truly exceeded the quotas.

4. END-USER VALIDATION OF THE PLATFORM

The end-user validation of the complete platform is an important goal of the project, as it encompasses a full spectrum of hardware and software features specially tailored to help meeting the dependability requirements. To achieve this validation, a set of applicative use-cases has been defined. We present two of them hereafter.

4.1. Low-level validation with basic benchmarks

As a first validation step, and besides unit testing of the components of the platform, we ported an ensemble of

benchmark applications. The goals are multiple :

- assessment of basic core performance with standard benchmarks such as Dhrystone or CoreMark,
- test of the memory hierarchy performance with other compute kernels such as matrix multiplication or FFT with configurable data size,
- RISC-V ISA conformance with standard test suite,
- and baseline characterization of interference channels with stressing benchmarks.

The latter are small programs written in assembly, putting a configurable pressure on a shared hardware resource with repeated access in a precisely controlled fashion [19]. Those will also enable the verification of the proposed interference monitoring and mitigation mechanisms.

Additional integration tests have also been defined, in order to cover platform requirements with particular interaction between hardware and software mechanisms, such as DMA transfers or interrupts.

4.2. High-level validation with realistic space use-case

For high-level validation, a realistic space use-case will be deployed in the context of the De-RISC project. It is based on the LVCUGEN framework, and the CCSDS123 hyperspectral image compression standard, forming a full mixed-critical system.

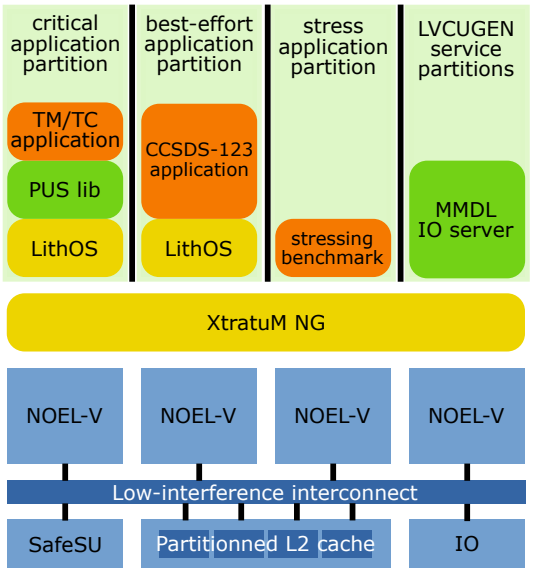

Figure 6. Example configuration of the flight software use-case

Several deployment configurations will be tested, with one example shown in figure 6. All configuration run a critical partition performing TM/TC (telemeasure and telecontrol) messaging and the LVCUGEN system partitions that manage IO and mode management. The platform also runs data-intensive partitions including an image processing partition and additional stressing benchmarks in some configurations. These put pressure on the memory hierarchy and associated shared resources. The SafeSU will allow us to monitor the impact on the critical partition execution time.

The use-case takes advantage of the LVCUGEN (Logiciel de Vol Charge Utile GENerique) framework [13], [14], the CNES generic payload software based on Time & Space Partitioning, to accelerate development and to ensure a good representativity of the use-case.

LVCUGEN is based on XtratuM and LithOS, two key components of the De-RISC platform. It provides several common services that ease the development of payload software, such as IO server, modes management and data loading, fault monitoring, telemeasure and telecontrol library, thus allowing the developer to focus on the specific features of the payload system. This state-of-the-art framework for space payload software ensures the respect of best practices for flight software and to assess the adequation with the proposed mechanisms for dependability.

As a representative data-intensive application, we chose to use the latest standard for space image compression, CCSDS-123 [15]. This algorithm performs lossless compression of hyperspectral images, i.e. with a large number of wavelength bands. The raw data can therefore be quite large and with a cube-like organization. The algorithm takes advantage of this 3D locality with a special adaptive predictor, and performs a propagation of intermediate results. The data thoughput 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.

5. CURRENT STATUS

The current progress of the project is in line with the plans.

The SoC prototype is functional and a first integration of an FPGA design with the RISC-V Hypervisor (H) extension has been delievered to project partners. The De-RISC board in cPCI Serial Space 6U form factor is currently being released for manufacturing and the SoC design will subsequently be tested also on the De-RISC board. Preparations for the radiation validation of the De-RISC SoC design are on-going.

Validation of the SafeSU is starting soon with the usage of a specific stressing $\mu kernels$ designed by BSC to stress corner cases and ensure that the SafeSU works as expected. These $\mu kernels$ take into account the specific

architecture of De-RISC SoC to, for instance, perform a sequence of multiple cache misses which later can be observed through the different SafeSU counters. Since the De-RISC platform is a multicore platform, multiple and diverse types of $\mu kernels$ can be employed simultaneously to observe how interference occurs and monitor the fairness of the bus arbiter, for example. On the final phase of the SafeSU validation, utility benchmarks will be used to validate some aspects that the $\mu kernels$ cannot trigger (e.g., counter overflowing).

Similarly, a shared contention analysis will be performed employing a subset of the $\mu kernels$ to evaluate the fairness of the current algorithm responsible of the bus arbiter. Particularly, the arbiter is expected to be fair among the different cores. Otherwise, corrective actions for the bus arbiter will be elaborated by BSC.

On the software side, XNG is running in the De-RISC platform supporting all existing cores in symmetric multiprocessing (SMP). LithOS is also running as guest OS and some performance tests have been carried out in the platform. The basic development tools have also been ported to the De-RISC platform. In particular, $xcparser$ generates binary configuration files starting with the XML configuration files provided by the user. Other development tools like xci, $xcon$ or $xtraceview$ help the user to develop a system based on XNG. More advanced supporting tools such as the schedulability analyser $Xoncrete$ or the XPM Eclipse plug-in are also being ported to support the RISC-V architecture.

The next phase will be concentrated in validating the system requirements and reducing the gap to have a space qualified version of XNG over the De-RISC platform.

The end-user validation phase has started, and most benchmarks are already ported onto the platform. The single-core performance figures at 2.82 DMIPS/MHz and 4.41 Coremark/MHz are confirmed. The porting and integration of the space use-case is ongoing, and will proceed incrementally until the end of the project.

The end goal of the project is a commercial release of the platform in Q2 2022.

ACKNOWLEDGMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 869945.

REFERENCES

[1] https://www.gaisler.com/LEON5

[2] https://riscv.org/

[3] https://www.gaisler.com/NOEL-V

[4] https://www.gaisler.com/GR740

[5] https://fentiss.com/products/hypervisor/

[6] https://fentiss.com/products/lithos/

[7] https://www.derisc-project.eu/

[8] http://www.emc2-project.eu/

[9] RTCA DO-297: Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations. Radio Technical Commission for Aeronautics, Inc. (RTCA), Washington, DC, 2005.

[10] PikeOS protecting Earth. Whitepaper, March 2021. https://www.sysgo.com/space

[11] J. Nowotsch and M. Paulitsch, "Leveraging Multi-core Computing Architectures in Avionics," 2012 Ninth European Dependable Computing Conference, 2012, pp. 132-143, doi: 10.1109/EDCC.2012.27.

[12] L. Pomante, D. Andreetti, F. Federici, V. Muttillo, D. Pascucci: Analysis and design of a Command & Data Handling platform based on the LEON4 multi-core processor and PikeOS hypervisor. DAta Systems In Aerospace (DASIA), 2017

[13] Julien Galizzi, Jean-Jacques Metge, Paul Arberet, Eric Morand, Fabien Vigeant, et al.. LVCUGEN (TSP-based solution) and first porting feedback. Embedded Real Time Software and Systems (ERTS2012), Feb 2012, Toulouse, France.

[14] Julien Galizzi, Paul Arberet, Jean-Charles Damery, Christel Guy, Alfons Crespo, Miguel Masmano, Florence Roubert: LVCUGEN - Ready for Flight?. DAta Systems in Aerospace (DASIA), May 2015, Barcelona, Spain

[15] Lucana Santos Falcon, Roberto Camarero. Introduction to CCSDS compression standards and implementations offered by ESA, European Workshop on On-Board Data Processing (OBDP2019), Feb 2019, Noordwijk, Netherlands.

[16] Jordi Cardona, Carles Hernández, Jaume Abella, Francisco Cazorla. Maximum-Contention Control Unit (MCCU): Resource Access Count and Contention Time Enforcement. Design, Automation and Test in Europe Conference (DATE2019), March 2019, Florence, Italy.

[17] Javier Jalle, Mikel Fernandez, Jaume Abella, Jan Andersson, et al. Contention-aware performance monitoring counter support for real-time MPSoCs. Symposium on Industrial Embedded Systems (SIES2016), May 2016, Krakow, Poland.

[18] Guillem Cabo, Francisco Bas, Ruben Lorenzo, David Trilla, Sergi Alcaide, et al.. SafeSU: an Extended Statistics Unit for Multicore Timing Interference. European Test Symposium 2021 (ETS2021), May 2021, Online [To appear.

[19] Petar Radojkovic, Sylvain Girbal, Arnaud Grasset, Eduardo Quiñones, Sami Yehia, Francisco Cazorla. (2012). On the Evaluation of the Impact of Shared Resources in Multithreaded COTS Processors in Time-Critical Environments. TACO. 8. 34. 10.1145/2086696.2086713.