Neutron Irradiation Testing and Analysis of a Fault-Tolerant RISC-V System-on-Chip

Douglas Almeida dos Santos, André Martins Pio de Mattos, Lucas Matana Luza, Carlo Cazzaniga, Maria Kastriotou, Douglas Rossi de Melo, Luigi Dilillo

Title: Neutron Irradiation Testing and Analysis of a Fault-Tolerant RISC-V System-on-Chip

Author(s): Douglas A. Santos, André M. P. Mattos, Lucas M. Luza, Carlo Cazzaniga, Maria Kastriotou, Douglas R. Melo, and Luigi Dilillo

Document version: Post-print version (Final draft)

Please cite the original version:
D. A. Santos et al., "Neutron Irradiation Testing and Analysis of a Fault-Tolerant RISC-V System-on-Chip," 2022 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), 2022, pp. 1-6.

Neutron Irradiation Testing and Analysis of a Fault-Tolerant RISC-V System-on-Chip

Douglas A. Santos*, André M. P. Mattos*, Lucas M. Luza*, Carlo Cazzaniga‡,
Maria Kastriotou‡, Douglas R. Melo†, and Luigi Dilillo*
*LIRMM, University of Montpellier, CNRS, Montpellier, France
†LEDS, University of Vale do Itajaí, Brazil
‡ISIS Facility, STFC, Rutherford Appleton Laboratory, United Kingdom
{douglas.almeida-dos-santos, andre.martins-pio-de-mattos, lucas.matana-luza}@lirmm.fr,
{carlo.cazzaniga, maria.kastriotou}@stfc.ac.uk, drm@univali.br, dilillo@lirmm.fr

Abstract

The radiation in harsh environments affects electronic systems, inducing permanent and temporary errors. These effects lead to unpredictable behaviors detrimental to critical applications and fail-safe systems. This work evaluates the reliability of a fault-tolerant RISC-V System-on-Chip (SoC) under atmospheric neutron irradiation in a particle accelerator. Prior work has analyzed the effectiveness of the hardening techniques of this SoC in simulation and provided a preliminary characterization in an irradiation facility. The applied hardening techniques showed a significant reliability improvement compared to the unhardened implementation of the SoC. The system executed a performance benchmark as workload, which finished correctly in most runs despite suffering from Single Event Effects (SEEs). This work presents a detailed analysis of the experimental results, reporting error rates and classification, extending the analysis given in previous works. Finally, a comprehensive discussion of implementation limitations and the proposition of further improvements are provided.

Index Terms

RISC-V, System-on-Chip, Fault Tolerance, Radiation Effects, Dependable Systems

I. Introduction

Harsh environments impose challenging design decisions for dependable systems in critical applications. For instance, in avionics, electronic systems must withstand temperature variations, mechanical stress, and ionizing radiation. Thus, if not properly handled, these conditions can lead to catastrophic failures. At avionic altitudes, the radiation-induced effects can degrade the overall reliability of the electronic systems [1].

The atmospheric radiation-induced effects on electronics mainly arise from interacting with neutrons. These particles are generated in the uppermost layers of the Earth's atmosphere as a result of the interactions between Galactic Cosmic Rays (GCRs) and Solar Energetic Particles (SEPs) with the atmosphere's molecules [2], [3]. Besides neutron generation, these interactions trigger a cascade of nuclear reactions that produce various other secondary particles. As aforementioned, neutron-induced errors are a major source of reliability degradation that presents its peak nearby avionic altitude [4]. Since these particles are uncharged, they can reach electronic components deeply inside avionic systems [5], contributing to SEEs, resulting from the interaction between energetic particles and electronics' internal structures. Generally, SEEs related to neutrons are caused by the subproducts of the neutron interaction with the materials [4]. The generated effects can be transient, intermittent, or permanent [3], [6].

Radiation hardening is a broad topic, and various approaches are employed to achieve the required dependability, ranging from radiation-hardened hardware to fault tolerance techniques in software. However, trade-offs arise from these approaches: project budget, development time, performance, and various other factors. In this context, developers must evaluate the requirements of the application and define which strategy is adequate. Since customized radiation-hardened products have a high cost, long lead times, and often limited performance, nowadays, developers tend to exploit risk acceptance by applying fault tolerance techniques with Commercial Off-The-Shelf (COTS) components [7]. Then, to achieve utmost reliability in these cases, developers employ methods that usually rely on redundancy, exploiting temporal, spatial, and informational characteristics of a system [8].

The results presented in this paper have been obtained in the framework of the EU project RADNEXT, receiving funding from the European Union's Horizon 2020 research and innovation programme, Grant Agreement no. 101008126, from the Region d'Occitanie and the École Doctorale I2S from the University of Montpellier (contract no. 20007368/ALDOCT-000932), and from the Foundation for Support of Research and Innovation, Santa Catarina (FAPESC-2021TR001907).

Avionics are highly dependent on the embedded electronics that manage all critical functions of these platforms. These electronics are usually based on processors as their core unit. Therefore, these applications must use processors that implement fault tolerance techniques to meet the required reliability criteria. An emerging processor architecture becoming an industry standard is RISC-V [9]. The RISC-V design has an optimized Instruction Set Architecture (ISA) that aims at simplifying the processor implementation. Despite that, the RISC-V standard still allows the implementation of performant computer systems. This architecture was even evaluated for use in space applications [10]. Although there are several soft-core implementations of RISC-V [11]–[14], there are few readily available fault-tolerant RISC-V processors.

Thus, in this context, in [15], we presented the design of a fault-tolerant RISC-V SoC, aiming at hardening the processor against SEEs. The pivot for the soft-core implementation was achieving utmost dependability with the lowest cost in terms of resource utilization. Also, in order to engage the academic community and expose this fault-tolerant RISC-V processor to real applications, the design sources and practical information were provided as an open-source platform [16]. The proposed SoC applies Triple Modular Redundancy (TMR) to harden the controlling logic and the Arithmetic-Logic Unit (ALU) and Hamming code for hardening all processor registers. The work presented promising results demonstrating that the circuitry added to the processor increased its reliability by reducing error propagation. In [17], the initial characterizations in a radiation environment were performed, corroborating the propositions elaborated using fault injection and presenting further challenges. In the present work, we discuss improvements made in the processor's design, present the experimental setup at the irradiation facility, and provide a more detailed analysis of errors rate and classification. Therefore, the main contributions of this work rely on the design, testing, and reliability analysis of a fault-tolerant RISC-V SoC for use in avionic applications.

The remaining of the paper is structured as follows: Section II describes key aspects of the RISC-V implementation and its fault tolerance features; Section III presents the devices used for the experiment, the test facility, and the applied test methodology; Section IV analyses and compares the results from this neutron irradiation with previous results; Section V provides insights about the observed behavior and errors; and Section VI concludes the work and prospect further improvements in the design and testing of the proposed system.

II. Fault-Tolerant RISC-V System-on-Chip

The proposed Hardened RISC-V (HARV) processor [18], and its subsequent extension to an SoC [15] (here referred to as HARV-SoC), rely on hardening strategies to meet the requirements of critical applications targeting harsh environments. At the architectural level, the most critical parts of a Central Processing Unit (CPU) are the control unit, program counter, instruction register, register file, and the Arithmetic Logic Unit (ALU), as they can cause critical failures when affected by SEEs [6]. Therefore, critical components of the processor, including control, registers, and data path, are hardened using different fault tolerance techniques. An Error-Correcting Code (ECC) is implemented in all HARV registers, capable of correcting one faulty bit and detecting up to 2-bit faults. The same Single-Event Correction and Double-Error Detection (SECDED) mechanism is used in the memory controller to harden the data located in the inferred Random-Access Memory (RAM). The critical control circuits and combinational logic are hardened using TMR, where a component is tripled, and a voter decides the correct output based on a result majority. To protect against critical failures, we used a Watchdog Timer (WDT) with a fixed deadline. We adopted these techniques due to their wide employment in related work [19]–[22]. Also, their combination results in a robust and cost-effective hardening strategy, as shown in [15], [17].

The HARV-SoC is presented in Figure 1, highlighting its internal components and fault tolerance mechanisms. The architecture comprises the HARV processor, a volatile memory used as RAM, a bus interface to connect peripherals, a non-volatile memory used for instructions (flash), and peripherals. The HARV processor is a multi-cycle implementation of the RISC-V standard, comprehending the entire integer instruction-set (RV32I), except system calls and fence instructions. Besides this, the processor implementation includes dedicated Control and Status Registers (CSRs) for the observation of fault events. These CSRs contain a summary of errors detected since the processor boot for each type of event, which are recorded: the sum of ECC events (1-bit or 2-bit faults) on the system registers and the RAM memory; TMR events on the ALU and controlling logics; and last reset cause, for reporting WDT timeouts or other

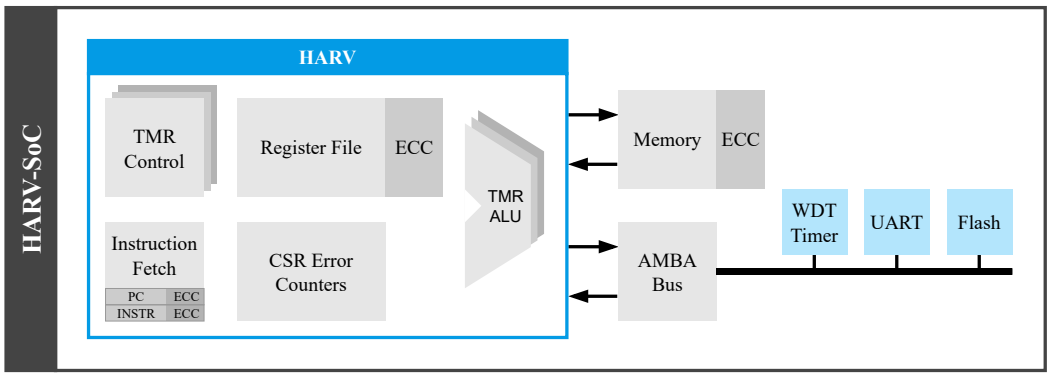

Fig. 1. Architecture diagram of the HARV-SoC design.

reset events. It is important to note that monitoring these events is always enabled, which allows reporting error rates even when the hardening features are disabled.

Since the previous work [17], HARV-SoC received an improvement in the logic implemented for the ECC operation in RAM memory transactions. Now, for each ECC event, the read data is corrected and written back in the concerned memory address. This behavior modification mitigates error accumulation in the RAM memory due to multiple bit upsets within the same word address.

III. Radiation Experiment

In order to evaluate the applied fault tolerance techniques and reliability trade-offs, several experimental campaigns with particle irradiation are planned during the development of the RISC-V SoC. In conjunction with fault injections, these experiments enable a deeper understanding of the impact of realistic radiation-induced errors on the system. Within this development methodology, the work reports the characterization of the system with neutron irradiation, which is suitable for targeting avionic environments. Besides that, these irradiation campaigns provide valuable data for identifying fault models and validating hardening techniques, which can support further studies and test campaigns targeting other radiation environments (e.g., high-energy protons and heavy ions for space applications). The following subsections provide an overview of the irradiation facility, the experimental setup, and the evaluation scenarios.

A. Irradiation Facility

The experiment related to this work was conducted in the ChipIr beamline, part of the ISIS Neutron and Muon Source, at the Rutherford Appleton Laboratory, UK. The generated neutron spectrum is similar to the encountered in the atmospheric environment with an increased intensity of several orders of magnitude depending on energy ranges and beamline configuration. ChipIr can generate a neutron flux of approximately 5×10^6 $cm^{-2}s^{-1}$ for energies with $E_n>10$ MeV. At the end of the campaign, the irradiation on the devices reached an accumulated total fluence of 8.77×10^{11} n/cm^2, representing a significant increase from the previous campaigns.

B. Experimental Setup

The experimental setup consisted of FPGA boards, auxiliary instruments, and equipment, including power supplies, FPGA programmers, and logging tools. In order to acquire a statistically significant event count, four identical boards were used to run the experiment in parallel. The experiment was conducted remotely due to restrictions on physical access to the facility. Thus, the setup included features for independent test execution: controllable USB devices for handling logging tools and programmer connections; remotely accessible power connections for resetting equipment inside the irradiation room; and a dedicated computer for the experiment execution. In order to support that, we automated the entire experiment to require minimal operator manipulation, which was present to monitor the execution and intervene when critical failures or crashes to auxiliary equipment occurred.

Figure 2 presents the board used for the experiment: Trenz's SMF2000. It has a compact and robust design, enabling convenient and effective beam area utilization. Also, the board's design is centered on a flash-based FPGA from Microchip: the SmartFusion2 M2S010 [23]. This device presents key advantages for radiation testing compared to SRAM-based FPGAs. Since the configuration memory is based on flash memory, it is more robust against single events, as shown by the manufacturer [24]. Although the enhanced reliability of the FPGA hosting the system under test, other internal components of interest are susceptible to SEEs, such as the Block RAMs (BRAMs), which are based on SRAM technology, and D-type Flip-Flops (DFFs), as shown in [25]. This setup is ideal for acquiring high-quality fault model data since only targeted structures are very sensitive to irradiation in an accelerated environment.

The RISC-V implementation takes advantage of many internal components of the device. Notably, the SoC uses internal flash memory to store the test programs, block RAMs to infer the processor's RAM memory, and the Clock Conditioning Circuitry (CCC) to synthesize the master clock frequency from an external oscillator. Also, the device provides the required internal modules for reprogramming the configuration memory, the internal user flash memory, and functional parameters. This is performed with the support of an external component located on the board to provide an interface with the experiment computer. Logs generated during the experiment are sent through a serial interface (UART), containing error counters and results from test executions. Each board requires a serial converter, externally attached to give a convenient interface for the experiment computer.

C. Evaluation Scenarios

In order to sensitize all internal components to different stimuli patterns during irradiation, the system executed an industry-standard benchmark as workload: the EEMBC's CoreMark™ [26]. It measures the processors' performance in embedded systems and is composed of four algorithms: list processing, matrix manipulation, state machines, and

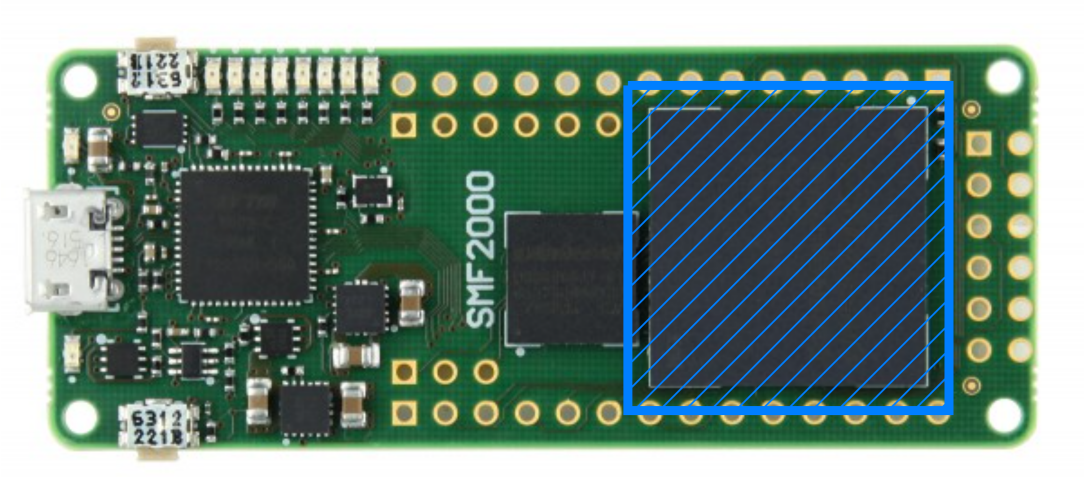

Fig. 2. Trenz's SMF2000 board. The highlighted region (in blue) encloses the M2S010 FPGA device and provides an estimate of the irradiated area.

Cyclic Redundancy Check (CRC). The latest algorithm is employed not only as a workload but also provides a self-checking mechanism for the inner steps of the execution. The benchmark performs several iterations with the option for extensive reporting of intermediate parts and a final summarizing score.

Regarding evaluation scenarios, the system executed the same workload with different hardening levels. This strategy allows the comparison and validation of the applied fault tolerance techniques. The hardening configuration is defined by software to achieve an automated and efficient execution, allowing real-time modifications without the requirement for reprogramming the FPGA's bitstream. Thus, during the experiment, the boards autonomously switch between four hardening configurations after a complete benchmark cycle: None, in which no internal component is hardened; Processor, enabling all hardening mechanisms of the CPU; Memory, when just the memory is hardened; and Memory&Processor, when all mechanisms in the CPU and the memory are enabled. As mentioned, the error counting functionality is always activated to complement the analysis provided by the benchmark.

IV. Results

This section reports the results obtained in the test campaign, providing a summary of resource utilization, an analysis of the efficacy of the hardening configurations, the impact of the errors in the benchmark execution, and a broader discussion about the observed errors.

A. Synthesis

We implemented HARV-SoC in the M2S010 device, which in turn is an SoC itself, containing two main sectors: the FPGA's Fabric and the Microcontroller Subsystem (MSS). The embedded ARM processor located in the MSS was disabled. It is important to distinguish the device's SoC architecture, inherent to the FPGA device, and the HARV-SoC, the target system under test and analysis. More specifically, the HARV-SoC uses Block RAMs, LUTs, and DFFs from the FPGA and CCC, Fabric Interface Controller (FIC), and an embedded Non-Volatile Memory (eNVM) from the MSS.

With a target operating frequency of 50 MHz for synthesis, the system has an usage of 67.4% logic elements: 64.7% LUT-4 elements and 23.6% DFF elements. As for FPGA memory elements, the design uses 92.2% of LSRAM-18K elements to infer a 16KB memory with SECDED for each 32-bit word, requiring seven additional bits in each word to store the ECC.

B. Error detection

Using the design's error-observation components, we identified the number of errors in the processor core and the data memory. In total, we could complete 996 CoreMark executions with the four hardening configurations.

Table I presents the number of detected errors during the entire test campaign and compares it to the previous work [17]. Although this work has a larger number of detected errors than the previous work (more than four times), the percentage of corrected errors remained very similar. The uncorrected errors correspond to double-bit upsets in the memory and registers. Most detected events originated in the memory, whereas less than 6% of the errors originated from the processor core. It is worth noting that the errors in the none and processor-hardened configurations accumulate and were likely to be counted repeatedly.

TABLE I
COMPARISON OF THE NUMBER OF ERRORS AND CORRECTABILITY.

Work	Hardening configuration	Errors	Corrected
[17]	**None**	165	0.00%
	Processor	147	2.04%
	Memory	106	95.28%
	Memory&Processor	105	98.10%
This work	**None**	993	0.00%
	Processor	772	2.33%
	Memory	483	95.45%
	Memory&Processor	452	98.89%

C. Classification of executions

Table II classifies the executions by comparing the UART output to a previously made golden run for each hardening configuration. The Match classification represents executions that finished without producing output errors, regardless of execution time. Note that Match executions may have detected bit upsets that were corrected or did not affect the execution. The Mismatch classification is executions that finished completely, up to the exit routine but had one or more errors in the UART output. The Timeout classification corresponds to executions that stopped executing properly and did not give any more UART output.

TABLE II
HARV-SOC CLASSIFICATION OF THE EXECUTIONS BASED ON THE UART OUTPUT.

Hardening configuration	Match	Mismatch	Timeout
None	5.39%	92.81%	1.80%
Processor	7.43%	91.89%	0.68%
Memory	42.86%	56.57%	0.57%
Memory&Processor	43.62%	55.32%	1.06%

Without hardening configuration (None), only 5.39% of the executions finished correctly (Match), while most of the other executions were classified as either Mismatch or Timeout. In comparison, when the processor hardening is enabled (Processor), the number of correct executions (Match) presents a slight increase of 2,04% and a decrease in the number of both Mismatch and Timeout executions.

The implementation of hardening in the memory reduces the number of Mismatch executions by 1.6$\times$ and increases the number of Match executions to 42.86%. When the hardening is enhanced by enabling both the processor core and memory fault tolerance, the number of Match executions increases by 1.8%. Although most errors originate from the memory, these results indicate that the processor is also a source of errors.

We noticed that most mismatches in executions were due to missing bytes in the UART received data. Hence, we analyze the executions by checking the Coremark built-in error verification, executed after all iterations are finished. This result is shown in Table III, which classifies the executions as correct benchmark result, benchmark finished with error, and benchmark execution failed to finish. This table also compares these results with our previous work [17], in which we tested a similar setup but had fewer beam-time and number of events. Therefore, the results of this work have enhanced confidence margins.

TABLE III
HARV-SOC CLASSIFICATION OF EXECUTIONS BASED ON THE COREMARK RESULT.

Work	Hardening configuration	Correct	Error	Timeout
[17]	**None**	73.08%	1.92%	25.00%
	Processor	77.08%	2.08%	20.83%
	Memory	98.21%	0.00%	1.79%
	Memory&Processor	100.00%	0.00%	0.00%
This work	**None**	75.80%	0.64%	23.57%
	Processor	73.19%	0.00%	26.81%
	Memory	97.59%	0.00%	2.41%
	Memory&Processor	97.73%	0.00%	2.27%

Generally, the results from our preliminary work and this one are consistent. The main difference is that this work relies on more meaningful statistical data (more available beam-time, more detected errors), with an increased number

of executions and events. For this reason, in the current experiment, we could also detect rare events such as execution failures in the Memory&Processor configuration leading to timeout failure. This type of event was not detected in the test campaign of the preliminary study.

In this work, we noticed a decrease in the number of correct executions of the Processor configuration compared to the None configuration. This result shows that most errors, indeed, originated in the memory. When ECC correction in the memory is enabled, the number of correct executions increases by 1.3×, resulting in 97.59% of correct executions. Further, the Memory&Processor configuration increases 0.14% the number of correct executions. Finally, eight executions failed to correctly run the benchmark using configurations with data memory error correction.

D. Analysis of failed executions

Several executions failed during the test campaign at the irradiation facility, most with the SoC configurations with error correction disabled. Few executions failed while the data memory's error correction was enabled. A total of eight executions failed with the SoC hardening configurations Memory and Memory&Processor. In order to improve the understanding of these failures, we analyzed each execution individually.

Table IV presents information regarding those executions. Each column represents one execution, referenced by labels. The UART output error is Y when there were mismatches in the execution output up to the failure moment, and N otherwise. The CoreMark runs are enumerated, showing the iteration counter of the benchmark execution when the failure occurred. Load/store access fault is Y when the processor failed due to an exception of load or store access fault, and N otherwise. The Watchdog Timer reset represents with Y the failures identified by the WDT, which performs a soft reset of the system when a failure is detected. Note that the WDT was capable of identifying all execution failures.

The Failure-inducing error detection is Y for failed executions in which the error that caused the failure was detected but not corrected, and N otherwise. Most failed executions (#M1, #M2, #M4, #MP1, #MP2, #MP3, and #MP4) failed due to Single-Event Functional Interrupt (SEFI), in which the system reached a state that requires a reset to return normal function, performed by the WDT.

The current error observation components could not detect the SEFIs because the observation may have failed, or the error did not occur inside the processor. These failures propagated in non-hardened components, such as the UART peripheral, the flash memory, or the FPGA MSS configuration. On the other hand, the #M2 execution failed due to a single-bit upset in the register file, which induced a failure because the processor core's hardening was disabled (using Memory configuration). During the #M2 execution, the register file upset caused an invalid address memory access, triggering a load/store access exception.

V. Discussion

The irradiation test campaign resulted in 43.62% executions with perfect UART output of the fully hardened implementation, compared to the 5.39% perfect executions of the non-hardened, showing that the hardened implementations increase the number of perfect executions. Despite having errors in the UART output, the CoreMark runs most of the time correctly. Results showed that 97.73% of executions finished the benchmark correctly while using the fully hardened HARV-SoC compared to 75.8% of correct executions without error correction. There were a few failing executions with memory error correction. These executions failed due to SEFIs on components external to the processor core. Finally, the hardening of the memory could correct 95.45% of detected errors, while the memory and processor hardening could correct 98.89% of detected errors. Also, the applied design change improved error counting and reduced the number of multiple-bit upsets.

The executions of the fully-hardened processor and most of the executions of the memory-hardened processor failed due to errors in the intercommunication of the SoC, which could be in the memory controllers, AMBA bus controllers,

TABLE IV
Details of failed executions with memory hardening.

Failure type	Failed executions							
	Memory hardened[1]				Memory&Processor hardened[2]			
	#M1	#M2	#M3	#M4	#MP1	#MP2	#MP3	#MP4
UART output errors	Y	N	Y	Y	N	Y	Y	N
CoreMark iteration number	107	96	15	150	N/A[3]	119	31	77
Load/store access fault	N	N	Y	N	N	N	N	N
Watchdog Timer reset	Y	Y	Y	Y	Y	Y	Y	Y
Failure-inducing error detected	N	N	Y	N	N	N	N	N

[1] #M*x* stands for executions with memory-hardened configuration.
[2] #MP*x* stands for executions with memory- and processor-hardened configuration.
[3] Execution failed before program initialization.

or the UART peripheral. These errors can potentially stop the execution of the processor for an undetermined time. Also, one execution of the memory-hardened processor failed due to an upset in a register that was being used for a memory access instruction. This upset caused the processor to throw an access fault exception that resulted in an infinite loop. Therefore, the WDT can identify these types of critical failures by detecting long idle periods, triggering a system reset to restore its normal operation.

Regarding the benchmark execution, CoreMark focuses on performance evaluation, which may present drawbacks regarding reliability evaluation due to a lack of SoC resource usage. Also, the verbose configuration creates a runtime overhead in communication instead of executing the benchmark's actual workload. Besides that, this configuration did not contribute to the result analysis since most of the identified failures shown in logs were likely to be only communication failures since they did not affect the processor execution. Therefore, exploring different workloads with lower overheads is desirable to increase confidence during the analysis of results.

The observation techniques used in the HARV processor detected several events and helped identify and characterize the failures. However, some events were still not identified by those, and enhanced observability techniques are desirable. The experimental setup had shortcomings due to radiation effects in the auxiliary equipment and tools, suffering connection losses to the experiment computer, and invalidating a few executions. Thus, additional efforts for a more stable experimental setup improve the beam time utilization.

VI. Conclusion

This work presents the effects of atmospheric neutron irradiation in a fault-tolerant RISC-V SoC, reporting the error analysis and providing insightful discussions of the applied test methodology. Moreover, it described critical architectural details of the HARV-SoC from the perspective of the hardening techniques. The employment of these techniques showed a significant improvement in reliability for neutron irradiation. The presented results show that most executions finished correctly despite suffering from SEEs.

In future work, we intend to provide more information for each detected error, such as the processor context and observed data, to enhance observability. Also, we expect to address the benchmark shortcomings by investigating other CoreMark configurations and workloads to evaluate the processor's reliability. Moreover, we plan to explore fault tolerance at the SoC level by monitoring the peripherals and controllers and even applying fault tolerance techniques in the peripherals. Finally, solutions are prospected to mitigate the problems caused by radiation-sensitive devices in the experimental setup.

References

[1] P. Cannon *et al.*, "Chapter 9 - Ionising radiation impacts on avionics and ground systems," in *Extreme space weather: impacts on engineered systems and infrastructure*. Royal Academy of Engineering, 2013.

[2] *Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices*, JEDEC Solid State Technology Association Std. JESD89B, Sep. 2021.

[3] L. Matana Luza, "Analysis of space and atmospheric radiation-induced effects on memory devices," Thesis, Université Montpellier, Dec. 2021.

[4] J. Barak and N. M. Yitzhak, "SEU rate in avionics: From sea level to high altitudes," *IEEE Transactions on Nuclear Science*, vol. 62, no. 6, pp. 3369–3380, 2015, doi: 10.1109/TNS.2015.2495324.

[5] I. Obodovskiy, "Chapter 7 - Interaction of neutrons with matter," in *Radiation - Fundamentals, Applications, Risks, and Safety*, I. Obodovskiy, Ed. Elsevier, 2019, pp. 151–160, doi: 10.1016/B978-0-444-63979-0.00007-0.

[6] M. Yang *et al.*, *Fault-tolerance techniques for spacecraft control computers*, 1st ed. Wiley Publishing, 2017.

[7] J. Budroweit and H. Patscheider, "Risk assessment for the use of COTS devices in space systems under consideration of radiation effects," *Electronics*, vol. 10, no. 9, 2021, doi: 10.3390/electronics10091008.

[8] D. J. Sorin, "Fault tolerant computer architecture," *Synthesis Lectures on Computer Architecture*, vol. 4, no. 1, pp. 1–104, 2009, doi: 10.2200/S00192ED1V01Y200904CAC005.

[9] A. Waterman *et al.*, "The RISC-V instruction set manual, volume I: Base user-level ISA," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2011-62, May 2011.

[10] S. di Mascio *et al.*, "The case for RISC-V in space," *Lecture Notes in Electrical Engineering*, vol. 550, no. 9783030119720, pp. 319–325, 2019, international Conference on Applications in Electronics Pervading Industry, Environment and Society (ApplePies), Nov. 2018. doi: 10.1007/978-3-030-11973-7_37.

[11] P. D. Schiavone *et al.*, "Slow and steady wins the race? a comparison of ultra-low-power RISC-V cores for Internet-of-Things applications," in *2017 27th International Symp. on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, Sep. 2017, pp. 1–8, doi: 10.1109/PATMOS.2017.8106976.

[12] C. Celio *et al.*, "BOOMv2: an open-source out-of-order RISC-V core," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2017-157, Sep. 2017.

[13] K. Asanović *et al.*, "The rocket chip generator," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2016-17, Apr 2016.

[14] S. Nolting *et al.*, "stnolting/neorv32: v1.7.2," Jun. 2022, doi: doi: 10.5281/zenodo.6696636.

[15] D. A. Santos *et al.*, "Reliability analysis of a fault-tolerant RISC-V system-on-chip," *Microelectronics Reliability*, vol. 125, p. 114346, 2021, doi: 10.1016/j.microrel.2021.114346.

[16] HARV, "HArdened RISC-V," 2021. [Online]. Available: http://xarc.org/harv

[17] D. A. Santos *et al.*, "Characterization of a RISC-V system-on-chip under neutron radiation," in *2021 16th International Conference on Design Technology of Integrated Systems in Nanoscale Era (DTIS)*, 2021, pp. 1–6, doi: 10.1109/DTIS53253.2021.9505054.

[18] D. A. Santos *et al.*, "A low-cost fault-tolerant RISC-V processor for space systems," in *2020 15th Design Technology of Integrated Systems in Nanoscale Era (DTIS)*, 2020, pp. 1–5, doi: 10.1109/DTIS48698.2020.9081185.

[19] A. E. Wilson *et al.*, “Neutron radiation testing of a TMR VexRiscv soft processor on SRAM-based FPGAs,” *IEEE Transactions on Nuclear Science*, pp. 1–1, 2021, doi: 10.1109/TNS.2021.3068835.
[20] A. Ramos *et al.*, “Efficient protection of the register file in soft-processors implemented on Xilinx FPGAs,” *IEEE Transactions on Computers*, vol. 67, no. 2, pp. 299–304, 2018, doi: 10.1109/TC.2017.2737996.
[21] R. C. Goerl *et al.*, “An efficient EDAC approach for handling multiple bit upsets in memory array,” *Microelectronics Reliability*, vol. 88-90, pp. 214 – 218, 2018, 29th European Symposium on Reliability of Electron Devices, Failure Physics and Analysis (ESREF 2018).
[22] A. Ramos *et al.*, “An ALU protection methodology for soft processors on SRAM-based FPGAs,” *IEEE Transactions on Computers*, vol. 68, no. 9, pp. 1404–1410, 2019, doi: doi: 10.1109/TC.2019.2907238.
[23] *IGLOO2 and SmartFusion2 Datasheet Version DS0128*, Microsemi, 2018, rev. 12.
[24] *SmartFusion2 SoC FPGA Version PB0115*, Microsemi, 2018, rev. 27.
[25] D. Dsilva *et al.*, “Neutron SEE testing of the 65nm SmartFusion2 flash-based FPGA,” in *2015 IEEE Radiation Effects Data Workshop (REDW)*, 2015, pp. 1–5, doi: 10.1109/REDW.2015.7336722.
[26] S. Gal-On and M. Levy, *Exploring CoreMark a benchmark maximizing simplicity and efficacy*, Embedded Microprocessor Benchmark Consortium, 2012.