POLITECNICO DI TORINO
Repository ISTITUZIONALE

Selective hardening of RISCV soft-processors for space applications

Original
Selective hardening of RISCV soft-processors for space applications / Cora, G., De Sio, C., Azimi, S., Sterpone, L.. - In: MICROELECTRONICS RELIABILITY. - ISSN 0026-2714. - 167:(2025). [10.1016/j.microrel.2025.115667]

Availability:
This version is available at: 11583/2998301 since: 2025-03-21T17:28:01Z

Publisher:
Elsevier

Published
DOI:10.1016/j.microrel.2025.115667

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

20 November 2026

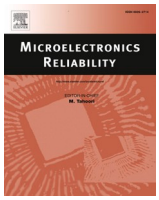
Microelectronics Reliability 167 (2025) 115667



Contents lists available at ScienceDirect

Microelectronics Reliability

journal homepage: www.elsevier.com/locate/microrel





Selective hardening of RISCV soft-processors for space applications☆

G. Cora *, C. De Sio, S. Azimi, L. Sterpone

Department of Control and Computer Engineering, Politecnico di Torino, Torino, Italy

A R T I C L E I N F O

Keywords:
RISC-V
Reliability
SEU
MBU

A B S T R A C T

RISC-V soft processors are becoming popular in various fields, including safety-critical ones, thanks to their open-source nature and flexibility. Despite the rapid progress in the reliability analysis of these devices, all the mitigation techniques are usually adopted to the whole soft-processor architecture.

In this study, we aim to identify the internal components of the RISC-V architecture that are particularly prone to errors, and accordingly investigate how the reliability of the design is affected when mitigation strategies, such as Triple Modular Redundancy (TMR), are applied selectively just to them.

The proposed approach has been applied to RISC-V architecture, NEORV32 which is implemented on Zynq 7020 SoC on a PYNQ-Z2 board. While more vulnerable modules of NEORV32 were identified through accurate reliability analysis, implementing selective TMR in these modules shows achieving satisfactory reliability levels while reducing the overall space requirements compared to a complete TMR design.

1. Introduction

In recent years, the RISC-V soft processor has attracted interest from various fields, including embedded, automotive, industrial, and aerospace ones, as per G. Furano et al. [1]. The open-source ISA led to many custom and open-source solutions available in the market. Such characteristic makes it very appealing to the space industry. Having the possibility to investigate the internal structure and compositions of the system, as well as modifying it according to the specific workload is fundamental for space systems to address the specific needs of the industry. When referring to safety-critical systems, the reliability aspect is one of the main concerns. In these fields, any failure in the device might lead to disastrous consequences. Thus, having a deeper knowledge of the internal architecture of the processor can help to exploit and improve the capabilities of the system and its dependability. Another important aspect of RISC-V-based systems is that they are often provided as HDL descriptions, meaning that they can be implemented and studied using Field Programmable Gate Arrays (FPGAs).

These devices can, in fact, achieve high performances with low costs. FPGA devices can implement both combinational and sequential logic, exploiting programmable Configurable Logic Blocks (CLBs), I/O interfaces, Memories, and Look-Up Tables (LUT) to achieve high performances and computational capabilities. Additionally, their reconfigurability ensures a high level of flexibility and adaptability both during the development phase and in the field.

Still, when considering safety-critical applications, reliability must be ensured and when FPGAs come into picture, new aspects must be considered. Given that the configuration of the implemented design is stored in the configuration memory (CRAM), a particular problem must be considered: the corruption of one or more cells may lead to modifications of the implemented design, leading to faulty behaviours of the system.

Single Event Upset (SEU) and Multiple Bit Upset (MBU), as highlighted by H. Quinn [2], are the most common sources of these problems, being caused by energy particles that can hit the device when it is deployed in a radiation environment. As a result of the interaction, they may modify the content of the configuration memory by flipping one or multiple bits.

For these reasons, SEUs and MBUs cannot be ignored, and suitable mitigation strategies have been studied to alleviate their effects. Examples are the Triple Modular Redundancy (TMR) technique or, more in general, N-modular redundancy, as showed by E. Vacca et al. [3]. These techniques require the system to be triplicated three or N number of times while letting a voter decide the correct output between all the provided ones. Even if these solutions improve overall reliability, they also lead to an increase in costs in terms of power, area, and performance. To increase yield and reduce the overhead of such approaches, a better knowledge of the system under test is fundamental.

☆ This article is part of the special issue entitled: 'ESREF 2024' published in Microelectronics Reliability.

* Corresponding author.
E-mail address: giorgio.cora@polito.it (G. Cora).

https://doi.org/10.1016/j.microrel.2025.115667
Received 21 November 2024; Accepted 26 February 2025
Available online 4 March 2025
0026-2714/© 2025 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

1.1. Main contribution

Even if the reliability analysis of RISC-V architectures has been conducted from different points of view and multiple mitigation strategies have been proposed and adopted, most of the research works applied them to the whole architecture. However, exploiting the modular property of RISC-V architectures can allow for area, power, and performances optimizations with, possibly, no critical degradation in the overall dependability.

We propose an investigation on the reliability of the internal modules of a specific implementation of RISC-V architecture and, most importantly, evaluate and discuss how to effectively apply mitigation strategies to smaller portions of the system rather than the whole design.

This work is organized as follows. Section 2 describes the state of the art. Section 3 introduces the proposed validation methodology. In Section 4, the case of the study and the result analysis are presented. Finally, the conclusion and future work are described in Section 5.

2. State of the art

In the last few years, RISC-V architectures have been undergoing several validation procedures, and their eventual adoption in the space environment has been investigated, leading to the development of various reliable architectures, such as the ones proposed by J. Andersson [4] and D. A. Santos et al. [5]. Authors S. Di Mascio et al. [6], have highlighted all the requirements, advantages, and limitations of adopting such architectures in a radiation environment. Similarly, several analyses have been conducted on the reliability and sensitivity of SRAM-based FPGAs when SEUs and MBUs affect their configuration memory, as showed by T. Li et al. [7].

Some preliminary evaluations of mitigation approaches have been conducted by Á. B. de Oliveira et al. [8], and D. Santos et al. [9], where the authors exploited the behaviour of RISC-V systems under radiation testing, targeting either FPGAs or SoC devices. Despite the effectiveness of these methodologies, they also have several disadvantages: high cost, time, and hardware requirements.

For these reasons, to simplify the verification procedure, fault emulation is often adopted in place of radiation tests. This provides a faster and simpler approach, that allows us to understand in a deeper way the reliability of the systems under test. As for radiation testing, this technique is usually applied to compare non-hardened and hardened versions of RISC-V architectures, as per Aranda, L.A et al. [10]. Additionally, suitable analysis can be conducted on the effectiveness of the adoption of mitigation techniques at software level rather than the physical hardware components, as highlighted by C. De Sio et al. [11].

However, it is hard to find considerations regarding how mitigation strategies, when applied to the internal modules of the design, can affect the reliability of the system. For example, in Santos, D.A. et al., [12], only specific modules have been hardened, but no exhaustive motivations have been given on why only some of them were selected.

In this paper, we present an analysis of which of the internal modules in a RISC-V architecture are more prone to errors. Then, suitable hardening techniques were applied only to them, and the reliability of this new architecture was evaluated with respect to two other cases: a non-hardened design and a version in which the whole processor has been triplicated. This allows us to understand how applying mitigation techniques to the internal modules of a RISC-V architecture can have a good impact on area occupation and performances, with small degradations in the overall reliability.

3. Proposed selective mitigation methodology

To efficiently improve the robustness of the system under evaluation, the major requirement was to understand which of the modules of the architecture were the less reliable ones. For this purpose, a novel design flow is proposed, involving the FPGA implementation phase and emulation of the faults in the configuration memory. Once the most critical modules are identified, mitigation techniques are applied, and, finally, the resulting system is validated by performing a reliability comparison of the non-hardened and hardened systems.

3.1. Critical modules identification

The overall procedure required to identify critical modules within the design follows the steps highlighted in Fig. 1a.

Isolation Design Flow (IDF) technique was adopted to identify which of the modules inside the soft processor are the most critical ones. This step, performed during the FPGA implementation process, gives the possibility of placing every element of the design onto specific FPGA resources and can already improve the reliability of the system, as showed by Portaluri, A. et al. [13], and E. Vacca et al. [17]. However, in this paperwork, this technique was adopted to make each of the modules of the RISC-V independent one from the other, such that every one of them could be analysed independently.

Once the modules were isolated, injections addressing specific parts of the configuration memory related to specific portions of the FPGA had to be performed. Some FPGA vendors, such as AMD, provide tools and additional information to support reliability evaluation techniques. The AMD Soft-Error Mitigation Controller, 2018, uses information such as Essential Bits Data (EBD) and Essential Bits Content (EBC) to perform injection over those that are considered essential bits of the design. Essential bits are identified as the bits in the configuration memory that, if corrupted, may modify the behaviour of the implemented system.

Even if the tools provide the possibility to inject faults at specific addresses, no information is provided about the correspondence between such addresses and associated resources or modules, forcing robustness evaluation to rely on a black-box approach. For this reason, in the last years, different tools allowing for bitstream analysis and manipulation were developed. PyXEL is a toolkit developed by C. De Sio et al. [14], whose features were used in this paperwork to analyse the bitstream and EBC files, find the correspondence between essential bits and the modules they refer to, and finally generate the addresses for the injections to be fed to the SEM-IP core [18]. Fig. 2a highlights the implemented design after the adoption of Isolation Design Flow technique, while Fig. 2b shows the plain, full and partial TMR have been implemented to validate the proposed approach (only the full TMR version is highlighted).

3.2. Fault emulation

The validation process has been conducted considering both Single Event Upset (SEUs) and Multiple Bit Upset (MBUs). A common approach consists of configuring the FPGA with a corrupted version of the configuration data, where one or multiple bits are flipped.

This methodology, however, is slow due to the time required to upload the corrupted bitstream in memory. Relying on SEM-IP allows a faster injection process for both SEU and MBU. Indeed SEM IP-Core can be instrumented, other than memory scrubbing and correction of errors, for injecting faulty values by flipping bits at specific coordinates in memory.

The entire validation process consists of different phases, as highlighted in Fig. 1b, and it is managed in an autonomous way by a Python-based handler. At the beginning of every iteration, a new injection address is fetched from the injection list, and the injection is performed through the SEM-IP. Then, a benchmark application is run, its response captured, and the fault is corrected, to prepare the system for a new injection. In case any of these steps generated an error, the FPGA was re-programmed from scratch. When MBU analyses had to be conducted, instead of a single address, a specific set of coordinates had to be fetched from the initial list.

The same procedure has been adopted to evaluate the reliability of the isolated RISC-V design, in both hardened and non-hardened

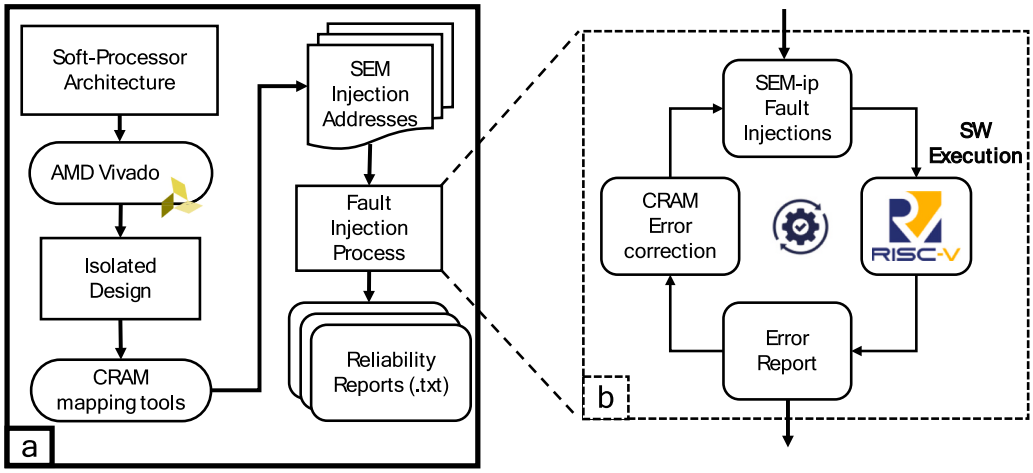


Fig. 1. (a) Critical modules Identification process and (b) The fault injection procedure.

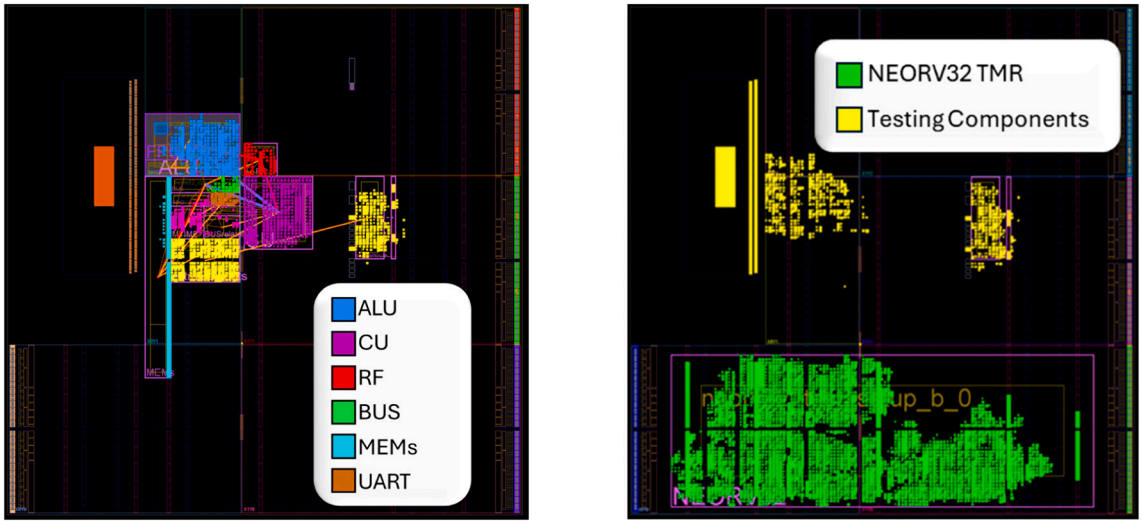


Fig. 2. (a) Neorv32 Implementation Results for Modular Fault Injection Campaign; (b) Full TMR Neorv32 implementation.

versions. The only difference is that, for the isolated design flow, an exhaustive campaign took place for every module, while, in the other cases, random injections were performed over the whole FPGA region where the processor was implemented, to resemble as much as possible space-environment conditions.

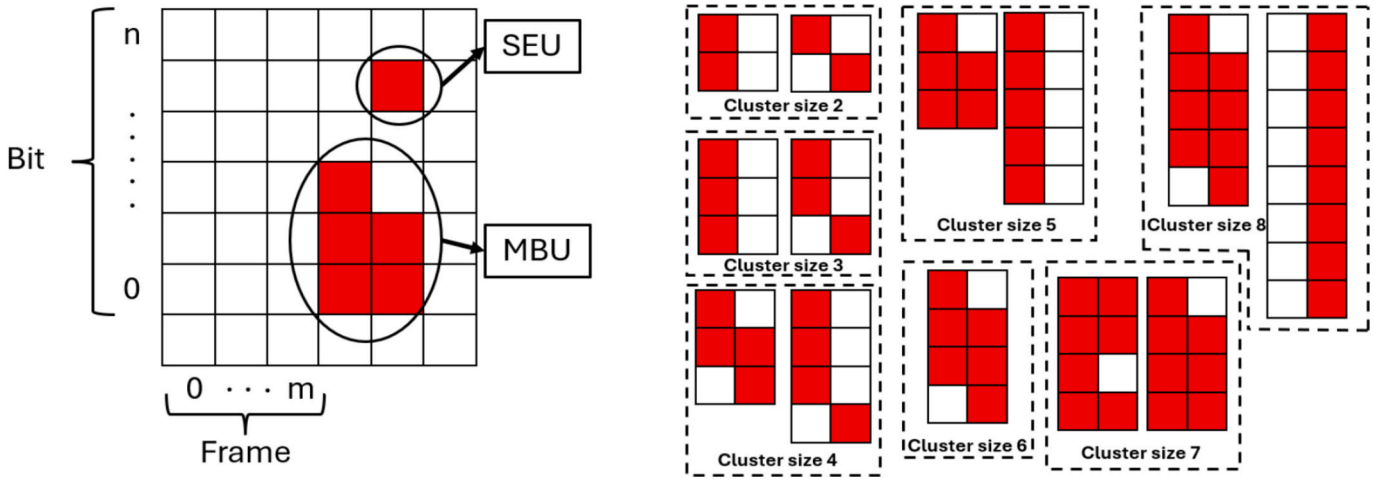


Fig. 3. (a) SEU and MBU Effects in the FPGA Configuration Memory (b) MBU Clusters Characterization.

3.3. Fault model

When addressing SEUs type of fault, whose effect on the configuration memory has been highlighted in Fig. 3a next to the MBU case, at each injection run only a single bit at time in the was altered.

For the case of Multiple Bit Upset, further analysis had to be conducted. According to the radiation test that we performed at PSI proton facility on the same technology by S. Azimi, et al. [15], when MBU affects the configuration memories of CMOS devices they tend to follow specific patterns, as represented in Fig. 3b.

For every MBU, a cluster size and pattern were identified according to their probability, a random injection address was selected and, starting from this information, the cluster to be injected was composed.

In the MBU case, two important considerations must be highlighted. First, the starting address for composing the cluster was always selected among Essential Bits, although the other bits in the cluster were not necessarily essential ones.

Second, if the starting bit was close enough to the borders of the Pblock in which the implemented design was instantiated, causing other bits in the MBU cluster to refer to resources external to the device under test, then, those bits were discarded.

4. Experimental analysis

The proposed approach has been validated by adopting Zynq 7020 SoC on a PYNQ-Z2 board. A RISC-V architecture has been implemented in the programmable logic of the device.

4.1. Case of study: NEORV32

The NEORV32, created by S. Nolting et al. [16], has been selected as a case study. It is an open-source, highly customizable soft processor that is available in both VHDL and Verilog languages, making it extremely suitable for FPGA implementations. It allows different ISA extensions to be activated. Our design implements the *C*, *M*, and *Zfinx* ISA extensions, enabling support for compressed, multiplication, division and floating-point instructions to be executed; *Zicntr and Zihpm* are enabled by default and provide support for additional control and status registers as well as performance counters. It supports both bare-metal and Real-time Operating System (RTOS) executions, but in our study the first approach has been selected. The modules we analysed in this work are the Arithmetic and Logic Unit, the BUS, the Control Unit, the Register File, the UART, and the Memories, both data and instruction ones.

4.2. Software benchmarks

Considering the space environment as the main case of study, benchmarks that require high number of mathematical operations is the usual approach to go with. Therefore, four benchmarks were selected and suitably adapted to the processor under test:

- *Whetstone:* a performance benchmark performing arithmetic, floating-point, and logic operations such as matrix convolution and angle and root computations.
- *Linpack:* this is a variation of LINPACK-100, which solves systems of linear equations with large matrices of floating-point numbers
- *Dhrystone:* a performance benchmark based on integer arithmetic, string operations, and memory accesses.
- *MatMul:* a simple benchmark performing matrix multiplication between large matrices of integers.

4.3. Experimental analysis and result discussion

During the entire procedure, more than 2 million injections in the essential bits were performed to exhaustively validate the reliability of each internal module of the RISCV against SEUs; additionally, 30,000 fault locations, selected from both essential and non-essential bits, were analysed for each of the plain, TMR and partial TMR versions of the design, to validate the effectiveness of the proposed approach. This led to a total of around 2.5 million addresses tested. For the MBU case, 30,000 clusters were tested for each of the internal modules under test, plus 20,000 for validating the different hardened and non-hardened NEORV32 designs. Overall, almost 4 million injections were performed in one week using 5 different boards.

In Table 1 the resource utilization of the 5 designs has been reported. It is evident that the adoption of a partial TMR can be beneficial from a resource point of view, with a reduction between the full and the ALU-CU TMR versions of around 3 % of the LUTs, 1 % of the FF and 1/3 of the total BRAM utilization. For the CU-BUS and BUS-RF TMR versions even better area optimization have been achieved, but the impact of this approach on the overall reliability is non-negligible, as highlighted in the following sections.

Additionally, Table 2 and Table 3 reports the number of essential bits for each of the internal modules of the RISCV, as well as for the five versions of the neorv32 design. It can be observed that, in every case, the amount of essential bits in the RISCV does not match the total amount of the essential bits of the device implemented onto the FPGA. This is because, as mentioned previously, some additional elements, such as the SEM-ip or AXI interfaces, had to be inserted to support testing functionalities, but they are unrelated to the design under test and therefore not considered.

4.3.1. Single event upsets analysis

In Fig. 4a the error rates of each internal module of the RISC-V, for the SEU case, have been reported.

Results are overall balanced among the different benchmarks, with the ALU and CU components being the least reliable ones. However, as the benchmarks shift from floating-point ones towards memory-based operations, it is clear to see that the Arithmetic and Logic unit failure rates drop below 4 % while the BUS one rises to a peak of 18 %. This has to be attributed to the fact that the floating-point components, representing the majority of the ALU size, are not stimulated when running programs like Dhrystone or Matmul, while a significant amount of data needs to be loaded to and from the memory.

Stemming from the analysis highlighted above, efficient and modular-based TMR applications can lead to an improvement in the total occupied area while reliability performance degradation is kept at a minimum.

This analysis took into consideration not only the error rates of each module but also their occupied area; this led to the decision of replicating, in one version of the design, only the ALU and the CU. Additionally, given the reliability analysis results for the memory-intensive benchmarks, we decided to try triplicating the CU and the BUS unit. Finally, the RF and the BUS were mitigated in a third version of the design.

The results coming from the fault injection campaigns conducted on the plain, full TMR and partial TMR designs have been reported in Fig. 5a. Depending on how the TMR has been implemented, a balance between reliability and resources utilization has been found.

For the ALU-CU TMR version, significative results have been achieved, especially when running the floating-point based benchmarks. Although the minimal reliability loss, the resource utilization is not

Table 1
Neorv32 resource utilization.

Neorv32 implementation	LUT [%]	FF [%]	BRAM [%]	DSP [%]
Plain	6.02	1.87	9.64	0.91
Full TMR	18.18	5.62	28.93	2.73
ALU-CU TMR	14.23	4.69	9.64	2.73
CU-BUS TMR	11.22	3.36	9.64	0.91
RF-BUS TMR	6.02	2.02	11.79	0.91

Table 2
Essential bits distribution of plain RISCV internal modules.

Module	Essential Bits	% of Total Essential Bits	% of CRAM Bits
Arithmetic Logic Unit (ALU)	183,539	20.39	0.57
Control Unit (CU)	238,327	26.48	0.74
Register File (RF)	27,021	3.00	0.08
Internal Bus (BUS)	17,730	1.97	0.05
Internal Memories (MEMs)	43,122	4.79	0.13
Serial I/O Module (UART)	13,797	1.53	0.04

Table 3
Essential bits distribution of different RISC-V design.

Neorv32 design	Essential bits	% of total essential bits	% of CRAM bits
Plain	588,463	60.98	1.82
Full TMR	1,816,228	82.83	5.62
ALU-CU TMR	1,673,022	81.63	5.18
CU-BUS TMR	1,247,267	76.81	3.86
RF-BUS TMR	705,182	65.19	2.18

particularly reduced. On the other hand, interesting consideration have been obtained when running memory-intensive benchmarks on the BUS-CU TMR version of the design. In this case, while the area occupation is significantly reduced, as highlighted in Table 1, the reliability degradation of the design is minimal, being still less than half of the plain version and experiencing a 1 % increase compared to the full TMR design. As expected, the BUS-RF TMR version does not achieve stunning result, despite the significant area reduction.

4.3.2. Multiple bit upsets analysis

Similarly to the SEU case, an additional analysis has been conducted to validate the proposed approach for Multiple Bit Upsets (MBUs). Since more configuration memory cells are affected simultaneously in the MBU case, their effect is generally more dangerous, making the design's reliability more easily compromised. Fig. 4b presents the results related to the resilience of the NEORV32 internal modules against Multiple Bit Upset. As expected, the error rates are generally higher if compared to the SEU case, with the ALU reaching a failure rate peak of 26.55 %. However, this value drops to 7.45 % when memory-intensive benchmarks are considered, for the reasons discussed in the previous section.

As for the SEU case, the TMR technique has been adopted in the same way, leading to the analysis of five different designs. Fig. 5b highlights the error rates in the different cases.

Although the ALU-CU TMR version achieves impressive results, being similar or even better than the full TMR version, significant consideration can be made about the CU-BUS TMR version of the Neorv32. In fact, a significant number of resources has been saved while maintaining the design's reliability, especially when running the Dhrystone and Matmul benchmarks. As anticipated, the triplication of the RF and BUS units alone does not provide any significant benefits, proving once more the effectiveness of the proposed approach.

5. Conclusions and future works

This work described how the division of the internal structure of RISC-V architectures can be used to obtain good reliability levels while granting better performances from an area and power point of view. Depending also on the adopted benchmarks, the hardening of specific modules might be equal to or even more efficient than hardening the whole design.

Possible future works include the adoption of other mitigation

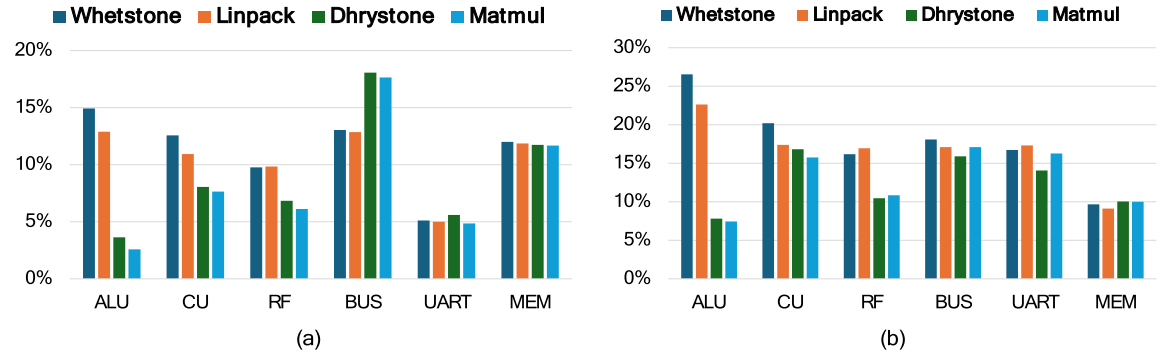


Fig. 4. (a) SEU Analysis Exhaustive Error Rates of Internal Modules on plain version (b) MBU Analysis Exhaustive Error Rates of Internal Modules.

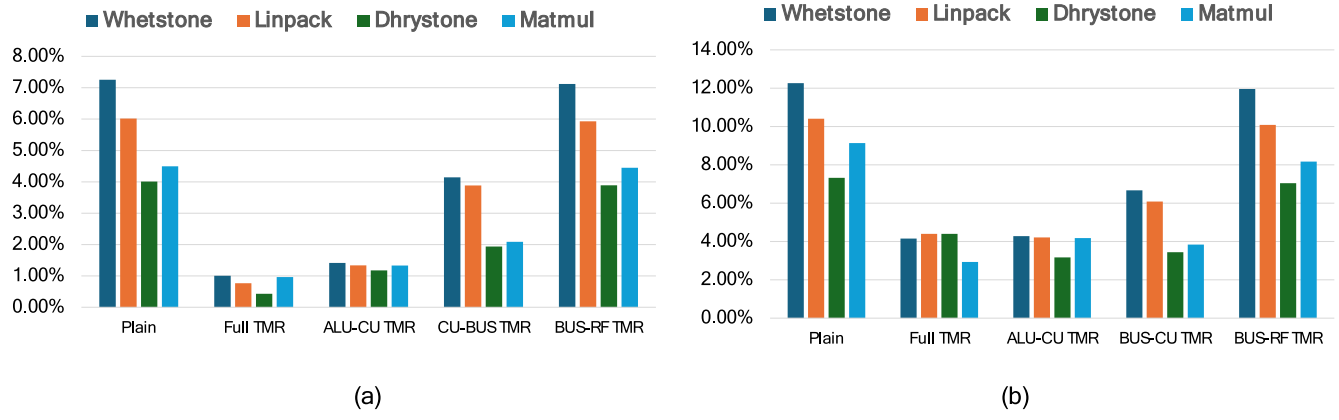


Fig. 5. (a) SEU Error Rates (b) MBU Error Rates.

techniques or implementing the Isolation Design Flow Technique next to the TMR approach to possibly obtain even better results.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] G. Furano, et al., A European Roadmap to Leverage RISC-V in Space Applications, IEEE Aerospace Conference (AERO), 2022.
[2] H. Quinn, Radiation effects in reconfigurable FPGAs, Semicond. Sci. Technol. 32 (4) (2017).
[3] E. Vacca, et al., Failure rate analysis of radiation tolerant design techniques on SRAM-based FPGA, Microelectron. Reliab. 138 (2022).
[4] J. Andersson, Development of a NOEL-V RISC-V SoC Targeting Space Applications, 2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W), Valencia, Spain, 2020, pp. 66–67.
[5] D.A. Santos, et al., A Low-Cost Fault-Tolerant RISC-V Processor for Space Systems, 2020 15th Design & Technology of Integrated Systems in Nanoscale Era (DTIS), Marrakech, Morocco, 2020, pp. 1–5.
[6] S. Di Mascio, et al., The case for RISC-V in space, in: S. Saponara, A. De Gloria (Eds.), Applications in Electronics Pervading Industry, Environment and Society, Lecture Notes in Electrical Engineering, Springer International Publishing, Cham, 2019, pp. 319–325, https://doi.org/10.1007/978-3-030-11973-7_37.
[7] T. Li, et al., Investigation into SEU effects and hardening strategies in SRAM based FPGA, in: European Conference on Radiation and Its Effects on Components and Systems (RADECS), 2017.
[8] A.B. de Oliveira, et al., Evaluating soft core RISC-V processor in SRAM-based FPGA under radiation effects, IEEE Trans. Nucl. Sci. 67 (7) (July 2020) 1503–1510.
[9] D. Santos, et al., Neutron irradiation testing and analysis of a fault-tolerant RISC-V system-on-chip, in: 2022 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), Austin, TX, USA, 2022, pp. 1–6.
[10] L.A. Aranda, et al., Analysis of the critical bits of a RISC-V processor implemented in an SRAM-based FPGA for space applications, Electronics 9 (2020) 175, https://doi.org/10.3390/electronics9010175.
[11] C. De Sio, S. Azimi, A. Portaluri, L. Sterpone, SEU evaluation of hardened-by-replication software in RISC-V soft processor, in: 2021 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), Athens, Greece, 2021, pp. 1–6.
[12] D.A. Santos, A.M.P. Mattos, D.R. Melo, L. Dilillo, Enhancing fault awareness and reliability of a fault-tolerant RISC-V system-on-chip, Electronics 12 (2023) 2557, https://doi.org/10.3390/electronics12122557.
[13] A. Portaluri, et al., New Domain Based Isolation Design Flow for Reconfigurable SoCs, IEEE International Symposium on On-Line Testing and Robust System Design (IOLTS), Torino, Italy, 2021.
[14] C. De Sio, et al., PyXEL: exploring bitstream analysis to assess and enhance the robustness of designs on FPGAs, in: International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design, 2023.
[15] S. Azimi, et al., A comparative radiation analysis of reconfigurable memory technologies: FinFET versus bulk CMOS, Microelectronics Reliability 138 (2022).
[16] S. Nolting and All the Awesome Contributors, The NEORV32 RISC-V processor, Zenodo (2024), https://doi.org/10.5281/zenodo.10851939. Mar. 22.
[17] E. Vacca et al. “Layout-oriented radiation effects mitigation in RISC-V soft processor”. In Proceedings of the 19th ACM International Conference on Computing Frontiers (CF ’22), Association for Computing Machinery, New York, NY, USA, 215–220. https://doi.org/10.1145/3528416.3530984.
[18] AMD, Soft Error Mitigation Controller v4.1,(PG036), 2018.