Exploring Fault-Tolerance in RISC-V Architectures

Mustafa Ensar Iskin
Istanbul Technical University
iskin17@itu.edu.tr

Berna Ors Yalcin
Istanbul Technical University
orssi@itu.edu.tr

Ayse Yılmazer
Istanbul Technical University
yilmazerayse@itu.edu.tr

***Abstract*— Fault tolerance is a critical aspect of modern electronic systems, particularly in environments like space, aviation, and automotive, where reliability is paramount. This paper examines the fault tolerance approaches in RISC-V architectures. Techniques such as spatial, temporal, and information redundancy are reviewed, with a focus on their integration into RISC-V-based systems. Comparative analysis of single-core and multi-core redundancy, including modular and heterogeneous approaches, demonstrates their impact on performance, energy efficiency, and cost. Testing methods ranging from simulation to real-world radiation exposure are evaluated. Finally, insights into future research opportunities are presented to guide the development of more robust and efficient fault-tolerant systems.**

Keywords—Redundancy, Fault Tolerance, SEU, MBU, ECC, TMR, DMR, ODMR

I. INTRODUCTION

The reliability of modern electronic systems, especially in critical applications, is directly related to minimizing the effects of errors. In areas such as space, aviation, and automotive, errors caused by environmental factors can threaten the stability of the system. In such applications, fault tolerance has become an indispensable requirement to ensure both system reliability and user safety.

Methods developed to provide fault tolerance are generally divided into categories such as spatial redundancy, temporal redundancy, and information redundancy. While spatial redundancy provides detection and correction of errors by physically duplicating components in the system, temporal redundancy focuses on error detection by repeating operations at different times. Information redundancy includes data protection methods such as error detection and correction codes. These techniques, used together or separately, increase the resilience of systems to faults, and must be balanced with factors such as energy efficiency, space, and cost.

The RISC-V architecture, which stands out with its open source and modular structure, provides a powerful platform for easy integration of fault tolerance mechanisms. The customizability of RISC-V allows the development of new techniques in both academic and industrial research. This flexibility allows for optimizing important metrics such as performance and energy efficiency, as well as increasing the reliability of processors.

In this study, RISC-V based fault tolerant architecture approaches will be examined in detail, and the advantages and limitations of existing methods will be evaluated. This review aims to provide a guiding framework for future designs.

Section II defines the types of faults and the basic methods used to provide fault tolerance. In the Section III, the fault tolerance approaches applied in RISC-V based architectures are examined in depth, and techniques are discussed. Section IV compares the test methods used and the results obtained from these tests. In the Section V, the analyses made in terms of performance, energy efficiency and reliability are presented; in the Section VI, the strengths and shortcomings of the reviewed studies are discussed and predictions are made about future studies. Finally, in the Section VII, the results of this study are discussed.

II. BACKGROUND

Before going into the details of the articles reviewed, it is important to explain the basic concepts related to reliability. The concepts discussed in this section are critical to understanding the rest of the study.

A. Types of Faults

Faults are generally divided into two categories: transient and permanent:

In developing fault tolerance systems, classifying the types of faults is very important. This is because different types of faults can be corrected in different ways. Transient faults are the faults that usually occur due to environmental factors and lose their effect in a short time. Data or commands can be corrupted due to single event upset or multiple bits upset [1]. Permanent faults cause the fixed logic to become stuck at 0 or 1 [2]. Permanent faults are caused by physical hardware failures and continue to affect as long as the hardware remains operational.

According to Annink et al. apart from these faults, there is also a concept called hardware trojans which can be triggered in certain situations and produce temporary errors, causing the system to function incorrectly, or they can completely disable the system. Annink and others have developed an effective method against hardware trojans using the bloom filter. [1]

B. Basic Methods in Fault Tolerance

It has been determined that certain techniques are used to detect and resolve faults in the reviewed articles. Fault detection and correction methods can be classified under the headings of spatial redundancy, temporal redundancy and information redundancy.

1) Spatial Redundancy

Spatial redundancy is the process of detecting and correcting faults by running multiple copies of a processor or system and comparing their outputs. This method encompasses approaches such as Double Modular Redundancy (DMR) and Triple Modular Redundancy (TMR). According to Dorflinger et al., aviation applications may require higher levels of spatial redundancy [3].

Spatial redundancy can be implemented at the core level or at the submodule level. Spatial redundancy is a very common method used in fault tolerance. Applications of spatial redundancy at the core level are called double core lock step (DCLS) / triple core lock step.

2) Temporal Redundancy

According to Rogenmoser et al., temporal redundancy allows fault detection by running a process multiple time [4]. This method can be particularly effective in handling

transient faults because these faults do not exhibit repetitive behavior over time.

The reviewed articles found that temporal redundancy is often used for faults that spatial redundancy detects but cannot fix. In addition, there is also a study that targets low-cost designs using only temporal redundancy. [5]

3) Information Redundancy

Information redundancy refers to the use of additional bits to provide fault tolerance. Techniques such as Error Correction Codes (ECC) are prominent applications of this method. [4] [7].

Information redundancy is generally used in structures where spatial redundancy is not possible. Thanks to information redundancy, it is possible to detect and correct errors with less area cost.

C. Advantage of RISC-V in Fault Tolerance

RISC-V offers unique advantages for both academic research and industrial applications thanks to its open source and modular structure. The transparent and flexible design of this architecture enables the development of customizable fault tolerance techniques. [6] [7]

There are many RISC-V designs that have been published as open source. The advantage of this is that a person who wants to work in the field of fault tolerance can focus on his/her field much faster by using these published cores. It was observed that open-source cores were used in most of the examined articles and the proposed solutions were integrated into these cores.

III. Fault Tolerant Architectural Approaches

The modular and extensible nature of the RISC-V architecture enables the implementation of diverse fault-tolerant design strategies. This section presents an overview of architectural approaches that employ spatial, temporal, and information redundancy at various system levels, including single-core, multi-core, and memory subsystems. Each approach is analyzed in terms of its structural characteristics, fault coverage capabilities, and applicability to different operational contexts.

A. Single Core Approaches

Architectures that address fault tolerance within a single core offer the advantage of both simplicity and energy efficiency. These approaches typically provide fault tolerance by optimizing the pipeline structure of the processor. Four prominent works are developed by Jiemin Li, Riccardo Tedeschi, Francesco Vigli and Alexander Dorflinger.

Jiemin Li and his team developed an architecture called DuckCore. DuckCore integrates fault tolerance into the pipeline structure, increasing the processor's resilience to transient errors. This architecture combines redundancy and error detection mechanisms at critical processing stages. In particular, error detection circuits have been added to pipeline stages. These circuits detect errors by verifying the data processed at each stage. Jiemin Li and his team explained, "Error detection circuits integrated at critical pipeline stages provide reliability without sacrificing performance." [7]

DuckCore includes not only error detection but also error correction mechanisms. Control circuits added to the pipeline structure use a replay strategy to fix detected errors. Li and his team emphasized the success of this approach by stating that "DuckCore's optimized error correction algorithms balance low energy consumption with reliability" [7]. In addition, lightweight hardware redundancy techniques used in the architecture ensured reliability without increasing the complexity of the processor. DuckCore architecture is shown in Fig. 1.

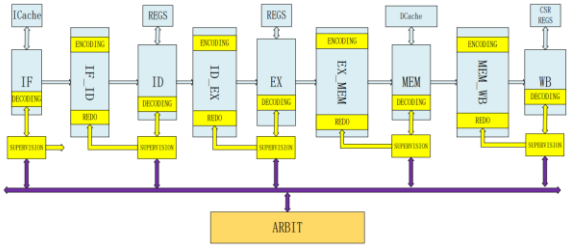

Fig. 1. Architecture of DuckCore from [7]

Riccardo Tedeschi and his team have developed an approach that increases fault tolerance by using temporal redundancy. This method, which is capable of detecting and correcting errors at critical stages in the pipeline, is described as "Temporal redundancy is an effective strategy to detect errors with repetitive operations without the need for additional hardware costs." Riccardo Tedeschi and his team also emphasized the practical benefits of the method by saying, "This approach is optimized for applications that require energy efficiency."[5]

Francesco Vigli and his team, while presenting a dual-core structure, also focused on fault tolerance mechanisms implemented within a single core. Explaining this design as "Error detection mechanisms integrated at the pipeline level can provide basic fault tolerance without loss of performance", Francesco Vigli and his team developed a simple but effective fault management system at the hardware level. The efficiency of this mechanism is demonstrated with the statements "Errors in the pipeline structure can be detected and corrected without increasing the hardware load". [2]

Deeper level detectors are distinguished by detecting errors that have not yet caused a disruption in the current state of the processor without interrupting the program flow. According to Dörflinger, "These detectors are designed to detect errors that may occur at deeper levels in the processor's control signals and data paths" [3].

These four studies present simple, cost-effective, and efficient solutions by integrating fault tolerance mechanisms into the core design. These approaches provide an important foundation for future developments and demonstrate the flexibility and customizability of the RISC-V platform.

B. Multi Core Approaches

Dual Core Lock-Step (DCLS) and Triple Core Lock-Step (TCLS) are the main implementations of multi core approach.

DCLS (Dual Core Lock-Step): It is based on the system's two cores executing the same instruction set in parallel and comparing their outputs. In this method, after errors are detected, cores are stopped or re-synchronized by the control unit. After the synchronization, faulty operation

is repeated. The main purpose of DCLS is to provide a low-cost solution that meets reliability requirements.

TCLS (Triple Core Lock-Step): The system executes the same instruction set in parallel with three cores and performs error detection based on consensus. Even if the results of two of the three cores are the same, this is considered as the correct result. In this case, the core that produced the erroneous result is identified and synchronized and included in the process. TCLS provides higher reliability but increases energy and hardware costs.

Michael Rogenmoser and his team compared TCLS and DCLS techniques in their study "Hybrid Modular Redundancy", discussing the advantages and disadvantages of these methods in space applications. According to Rogenmoser, "TCLS method provides higher fault tolerance, while significantly increasing energy and space requirements"[4]. A general structure of Multi core approaches is shown in Fig. 2. [3]

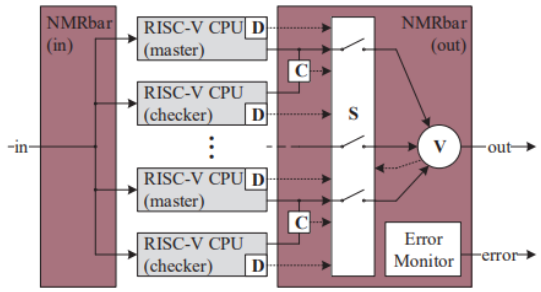

Fig. 2. General Structure of Multicore Approaches from [3]

1) Heterogeneous Multi-Core Architectures

Heterogeneous architectural redundancy aims to increase system reliability by combining different types of processor cores or specialized ones. In addition to structures such as the combination of Arm and RISC-V processors, specialized cores of the same type with different features can also be included in this architecture.

In their study Rodrigues et al. emphasized that Arm cores are optimized for high-performance tasks, while RISC-V cores are optimized for energy-efficient operations. The architecture in this study is shown in Fig. 3 [8].

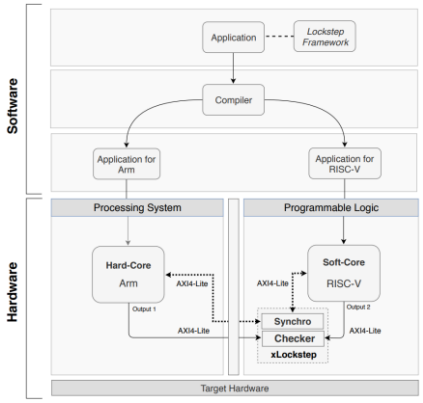

Fig. 3. Heterogeneous fault-tolerant architecture from [8]

Another study, "A RISC-V Fault-Tolerant Soft-Processor Based on Full/Partial Heterogeneous Dual-Core Protection", showed that two different RISC-V cores were used in the same system to create a heterogeneous structure. One of the cores is optimized for high-performance operations, while the other core focuses on low energy consumption. This structure balances the system's energy efficiency with performance requirements, while at the same time increasing system reliability with parallel operation and error detection mechanisms.[2]

Heterogeneous architecture provides advantages in terms of energy and area efficiency by taking advantage of the features offered by different cores. Thanks to these features, they play a critical role in applications requiring high reliability, such as space.

2) Dynamic Multi-Core Architectures

Dynamic and flexible redundancy models allow systems to adjust redundancy levels according to real-time requirements. These models are developed to optimize the balance between energy efficiency, performance, and fault tolerance. This model is mainly developed by Rogenmoser and called “On Demand Modular Redundancy”, ODMR. There are different studies about ODMR.

According to Rogenmoser et al., Hybrid Modular Redundancy saves energy by switching to triple-core mode for critical tasks while using dual-core mode for low-priority tasks [4]. These flexible transitions play a critical role in applications that require high reliability and energy savings.

Another approach which is developed by Rogenmoser et al. involves dynamically assigning processors to redundancy groups. These groups are configured based on the criticality of the task or the probability of failure and are optimized for performance or reliability. This flexibility is explained in the statement "On-Demand Redundancy Grouping allows processors to operate with customized redundancy levels for different workloads" [9].

Trikarenos study provides an example of this flexible structure. Designed with a triple modular redundancy (TMR) system and manufactured with TSMC 28nm technology, Trikarenos is described as "It switches to full redundancy mode only during critical tasks, making the most efficient use of system resources" [10]. It can shut down cores to save energy. In addition, the fact that Trikarenos will be produced and used for a cube satellite shows the tendency towards this method.

As a result, dynamic and flexible redundancy models increase the performance and reliability of RISC-V based systems thanks to their adaptable structures that respond to different application needs.

C. Memory Protection Methods

One of the most important steps in making a RISC-V core fault tolerant is to make the data and instruction memories fault tolerant. In addition, since memories occupy the largest area on a chip, they are also the sections with the highest probability of a fault being observed due to external effects. In real-world tests conducted by Santos et al. it is seen that 78% of the faults that occur in the memory, and 15% of the faults occur in the register file [6].

There are different ways to make memories fault tolerant. According to Dorflinger, although it is possible to use backups for memories, using ECC instead of full data backups significantly reduces resource usage [3]. ECC was commonly employed in the reviewed studies, and generally two different ECC structures were emphasized. While the first of these, SEC, corrects single-bit errors, SEC-DED can detect 2-bit errors in addition to correcting single-bit errors. Annink emphasized the use of ECC for protection against SEUs and bloom filters for protection against permanent

errors, and emphasized that when ECC is added to a memory, the reachable frequency decreases by 20% and the area overhead is 33.7% [1].

IV. Test Methods and Results

The methods which have been explained in the previous sections have been brought together in the articles to design fault-tolerant systems and tests have been carried out with different methods. While some of these tests are carried out in a simulation environment, some tests are real-world tests performed after the designed processor is implemented in FPGA or ASIC.

Real-world tests have been carried out with two different methods. The first method is to manipulate the data in the processor's memory or the internal structure of the processor to create errors. The second method is radiation tests. In this method, the circuit is exposed to a direct radiation flux.

A. Simulation Based Tests

The fastest and easiest way to see the behavior of the designed circuit in digital system design is to perform RTL simulations. 60% of the articles which are reviewed reported that they observed the results of RTL simulations, while 50% only described the results of RTL simulations. However, using only simulations does not provide complete information about the performance of the design. Because situations such as radiation, which are difficult to predict and model, cannot be fully created in simulation environments.

Tedeschi et al. stated that they saw a 1.82% error rate with the temporal lockstep method as a result of their simulations [5]. Secondly, Rogenmoser et al. reported that 12% of the injected faults in their simulations required fixing, and all tests completed without failure [4]. Lastly, Vigli et al. conducted different tests using different heterogeneous architectures, and the largest error rate obtained in these tests was recorded as 1.35%. [2]

B. Manually Creating a Fault on the Hardware

Four of the reviewed studies injected faults directly on real hardware. Three of these four articles implemented their designs on FPGA, while one design used TSMC 28nm technology. In all four of these studies, some memory elements or processor cores of the circuit were intervened to create errors. The primary drawback of direct hardware fault injection is the significantly lower level of control compared to simulation.

Only one of these hardware-based studies reported detailed result statistics. Dörflinger et al. observed 3.12% transient logic errors and 3.16% transient memory errors during fault injection experiments. They concluded that Deep Level Detectors (DLDs) detect the majority of logic faults, whereas ECC methods detect all observable memory faults. Additionally, DLDs significantly reduced the average detection latency of transient logic errors by up to 79% [3].

C. Radiation Tests

Radiation tests are the most expensive and difficult test methods to perform. In this test method, the designed hardware is operated in a radiation environment and the results are obtained in this environment. This method is used in only one article which are reviewed.

Santos et al. implemented the HARV-SoC architecture on an FPGA and subjected the circuit to radiation tests in two centers called ChipIr and CHARM. Different radiation fluxes were sent to the circuits in these centers. According to the information presented in the article, 55% of the errors that occurred in the tests performed in ChipIr could be corrected, 11% were detected but could not be corrected, and the remaining 33% could not be identified. In the tests performed in CHARM, 60% of the errors could be corrected, 37% were detected but could not be corrected, and the remaining 3% could not be identified [6].

D. Analysis of Results

It is seen that similar results are obtained because of errors created by manual triggering in the simulation environment and in the real world. However, the results in radiation tests are more pessimistic. The resolvable error rate in radiation tests is less than other methods. The reason for this situation may be inadequate modeling of radiation behavior. Further designs subjected to radiation testing will deepen our understanding.

Due to reasons such as the lack of common test steps and the lack of detailed results in each article, it is not possible to compare the fault tolerance capabilities of architectures.

V. Performance, Energy and Cost Evaluation

A. Performance-Cost Balance

Although the articles examined in this study aimed to improve processors in terms of fault tolerance, the performance of the developed systems and the cost of the systems are very critical. According to Rogenmoser, fault-tolerant RISC V cores are generally used in low-budget space devices and cost is important for such devices. In short, the point that is trying to be achieved is to increase the cost as little as possible and to lose as little performance as possible while making the processors fault tolerant [4].

It is seen that there are more up-to-date methods that improve the classical TCLS and DCLS methods. While some of these methods provide performance improvements, some can significantly reduce the cost.

Rogenmoser's work uses multiple processor cores to ensure fault tolerance, it can provide high performance by running these cores in parallel when appropriate. This system, which uses the flexible redundancy method, can provide a 2.96x performance increase in performance mode. At the same time, it is also more energy efficient than its competitors that offer similar performance with the developing technology. [9]

Tedeschi and his team designed the most advantageous processor in terms of area cost by using temporary redundancy. The processor calculates an instruction twice and compares the 2 results [5]. This method requires no additional hardware performance degradation is inevitable.

B. Energy Efficiency

Energy efficiency is another crucial metric. Li et al. have studied this issue and have shown that ECC can increase power consumption by 14%, but when an optimized algorithm is used, the power consumption increase is limited to 3% [7]. Rogenmoser et al. shown that energy efficiency

can be achieved by shutting down desired cores when necessary, within the scope of flexible redundancy. [10]

VI. Discussion And Future Works

A. Strengths and Weaknesses of Current Approaches

Within the scope of this study, it has been observed that there are different methods for developing fault tolerant RISCV and that new systems have been designed by combining these methods in a certain way.

Memory elements occupy the most significant area within chips. For this reason, backing up memories is much more costly and therefore redundant memory structures are not a recommended method. ECC-protected memories were frequently utilized in the reviewed articles, providing robust protection against SEUs. Although most of the studies conducted SEU-focused tests, MUE was also encountered in radiation tests.

For the fault tolerance, the first structure that comes to mind is the modular redundancy structure. It has been seen that these structures are used both at the core level and at the pipeline level. Although core level approaches are less costly, their fault detection capabilities are worse than deeper level detectors.

ODMR, a more flexible and innovative solution, aims to increase performance by using cores in parallel when desired. It has provided performance increases very close to the theoretical limit in certain applications. However, the performance increase will not be as high in applications that are not suitable for parallel computing.

B. Future Research Opportunities

The most important conclusion that can be drawn from the articles examined is that protection at the pipeline level is much more effective against faults. In addition, the ODMR method contributes more in terms of performance than any other approach. However, using the ODMR method at the core level greatly limits the type of application where performance gains can be achieved. By using a superscalar core, a TMR application at the pipeline level and an ODMR application can be combined to create a design that is both much more advanced in terms of fault correction and whose performance contribution will cover a wider range of applications.

Another part that can be improved about the articles is the test methods. Unfortunately, simulation results are too optimistic compared to radiation tests. The lack of common test steps makes it impossible to compare the fault-correcting abilities of designs. A test environment study that mimics the radiation effects in a simulation environment can be very helpful. In this way, different studies can run a common test. In this way, the results of the studies will be closer to the real world and different studies can be compared more fairly.

VII. Conclusion

In this study, different approaches in the literature for fault tolerant RISC-V design were examined. In the studies, it was seen that spatial, temporal and information redundancy types were used together. Although these approaches are used in every architecture, it was seen that each architecture is different from each other. Each architecture has its own advantages and disadvantages, and these advantages and disadvantages should be optimized according to the application.

Although the tests conducted in the studies were examined to compare the fault tolerance capabilities of the architectures, a full comparison could not be made due to the lack of a common benchmark. In addition, simulation results and radiation test results were obtained from different studies and compared. It is thought that the reason for the difference encountered in simulation and radiation tests may be the different architectures used, or it may be due to a lack of simulation.

Studies were also compared in areas such as performance, cost, energy efficiency and the advantages of innovative methods were discussed. Based on this, a prediction was made for future studies and what could be done to contribute to this field was discussed.

References

[1] E. B. Annink *et al*., "Preventing Soft Errors and Hardware Trojans in RISC-V Cores," *2022 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, Austin, TX, USA, 2022, pp. 1-6, doi: 10.1109/DFT56152.2022.9962340

[2] F. Vigli, M. Barbirotta, A. Cheikh, F. Menichelli, A. Mastrandrea and M. Olivieri, "A RISC-V Fault-Tolerant Soft-Processor Based on Full/Partial Heterogeneous Dual-Core Protection," in *IEEE Access*, vol. 12, pp. 30495-30506, 2024, doi: 10.1109/ACCESS.2024.3366806.

[3] A. Dörflinger, B. Kleinbeck, M. Albers, H. Michalik and M. Moya, "A Framework for Fault Tolerance in RISC-V," *2022 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, 2022, pp. 1-8, doi: 10.1109/DASC/PiCom/CBDCom/Cy55231.2022.9927800.

[4] M. Rogenmoser, Y. Tortorella, D. Rossi, F. Conti, and L. Benini, "Hybrid Modular Redundancy: Exploring modular redundancy approaches in RISC-V Multi-Core computing clusters for reliable processing in space," *ACM Transactions on Cyber-Physical Systems*, Nov. 2023, doi: 10.1145/3635161.

[5] R. Tedeschi et al., "A low-cost fault tolerance technique for microcontroller-class RISC-V processors," in Lecture Notes in Electrical Engineering, vol. 1263, M. Valle et al., Eds. Cham, Switzerland: Springer, 2025, pp. 38–45, doi:10.1007/978-3-031-71518-1_3.

[6] D. A. Santos, A. M. Mattos, D. R. Melo, and L. Dilillo, "Enhancing fault awareness and reliability of a fault-tolerant RISC-V system-on-chip," Electronics, vol. 12, no. 12, p. 2557, Jun. 2023. doi:10.3390/electronics12122557.

[7] J. Li, S. Zhang, and C. Bao, "DuckCore: A fault-tolerant processor core architecture based on the RISC-V ISA," *Electronics*, vol. 11, no. 1, p. 122, Dec. 2021. doi:10.3390/electronics11010122

[8] C. Rodrigues, I. Marques, S. Pinto, T. Gomes and A. Tavares, "Towards a Heterogeneous Fault-Tolerance Architecture based on Arm and RISC-V Processors," *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, Lisbon, Portugal, 2019, pp. 3112-3117, doi: 10.1109/IECON.2019.8926844.

[9] M. Rogenmoser, N. Wistoff, P. Vogel, F. Gürkaynak and L. Benini, "On-Demand Redundancy Grouping: Selectable Soft-Error Tolerance for a Multicore Cluster," 2022 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Nicosia, Cyprus, 2022, pp. 398-401, doi: 10.1109/ISVLSI54635.2022.00089.

[10] M. Rogenmoser and L. Benini, "Trikarenos: A Fault-Tolerant RISC-V-based Microcontroller for CubeSats in 28nm," *2023 30th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, Istanbul, Turkiye, 2023, pp. 1-4, doi: 10.1109/ICECS58634.2023.10382727.