POLITECNICO DI TORINO
Repository ISTITUZIONALE

In-Hardware Fault-Tolerance Controller for Multi-FPGA Clustered Architectures

Original
In-Hardware Fault-Tolerance Controller for Multi-FPGA Clustered Architectures / Cora, G., Rizzieri, D., De Sio, C., Azimi, S., Sterpone, L.. - In: IEEE TRANSACTIONS ON COMPUTERS. - ISSN 1557-9956. - 75:9(2026), pp. 3312-3325. [10.1109/TC.2026.3709831]

Availability:
This version is available at: 11583/3013173 since: 2026-07-16T09:44:48Z

Publisher:
IEEE

Published
DOI:10.1109/TC.2026.3709831

Terms of use:

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

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2026 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

20 November 2026

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

In-Hardware Fault-Tolerance Controller for Multi-FPGA Clustered Architectures

Giorgio Cora[1], *Student Member, IEEE*, Daniele Rizzieri[1], Corrado De Sio[1], *Member, IEEE,* Sarah Azimi[1], *Member, IEEE,* Luca Sterpone[1], *Senior Member, IEEE,*
[1]Politecnico di Torino, Torino, Italy

***Abstract*— The growing adoption of Commercial-Off-the-Shelf (COTS) components in space applications necessitates addressing performance and reliability challenges in radiation-intensive environments. Field-Programmable Gate Arrays (FPGAs), widely employed in these systems, offer exceptional computational capabilities but are vulnerable to radiation-induced faults, such as Single Event Upsets (SEUs), which can cause unpredictable behavior or even system failure. This paper presents a solution for enhancing the reliability of FPGA cluster-based systems through two custom-designed hardware solutions, the beacon controller and the robust bus, as well as error correction and workload redirection mechanisms. They have been integrated into an FPGA computing cluster for space telecommunication applications and tested against radiation-induced effects by fault injection techniques and radiation test campaigns. Experimental results demonstrated that the proposed techniques significantly reduce the system downtime, achieving up to 99% system availability, and reduce the radiation-induced failure rate effects below 4%. These results validate the proposed mitigation strategies as a robust and reliable approach for ensuring system dependability in harsh environments, demonstrating excellent performance through a fully FPGA-based design that eliminates the need for external controllers, unlike most state-of-the-art solutions.**

***Index Terms*— FPGA cluster, fault-tolerance, reliability, radiation hardening, FPGA-FPGA communication**

I. INTRODUCTION

In recent years, Field Programmable Gate Arrays (FPGAs) have attracted a growing interest from many application fields. Space exploration has adopted FPGA systems, including Commercial-Off-the-Shelf (COTS) solutions, in countless applications taking advantage of the low costs, fast time-to-market, flexibility, and high performance [1]. Using such devices in space missions and projects led to many technological advancements thanks to the involvement of stakeholders, researchers, and device vendors. Due to the demanding safety-critical and mission-critical requirements for space and avionic applications, reliability and fault-tolerance have been one of the main topics where to continue thriving innovative and better solutions. Since electronic devices operating in radiation environments like outer space are subject to disturbance induced by highly energetic particle interactions, adopting mechanisms to assure reliability and fault tolerance is paramount. The occurring faults depend on many factors such as the type of device, system, and environment. Such faults can lead to different misbehaviors, possibly resulting in critical errors that could compromise the lifespan of the entire mission [2]. When considering reconfigurable systems like SRAM-based FPGAs, the most susceptible part of such systems is the Configuration Random Access Memory (CRAM), which contains configuration data (CDATA) that defines the function implemented by the programmable fabric. A high-energy particle striking the device can cause charge deposition in one of the memory elements, leading to a Single Event Upset (SEU). A SEU is the corruption of the value of a memory cell. When such an event happens in the CRAM of a reconfigurable device, the functionality of the device may be corrupted, leading to wrong computations, Byzantine behavior, or system crashes. Critical errors could propagate to the user application layer, possibly compromising the whole system mission [3].

In the complex development flow ranging from silicon to the user application, many options are available to enhance the dependability of an FPGA system, and the research interest is ever-growing. Many different approaches have already been studied and characterized: several techniques can be applied at different abstraction levels, starting from the design and manufacturing of the device itself - this is the case of Radiation Hardened By-Design (RHBD) devices - going through all the FPGA application development up to the mitigation solutions applied to the software layer, like Algorithm-Based Fault Tolerance (ABFT) [7].

Along with improving reliability and dependability, performance is another essential aspect of mission-critical applications. A tradeoff is usually necessary between accuracy, robustness, resource usage, and budget availability to achieve the desired performance. In this context, one advantageous option is represented by multi-FPGA clustered systems. In such architectures, several programmable devices are connected to cooperate, communicate, and achieve enhanced operational throughput. In this way, the FPGAs' intrinsic parallelization and acceleration capabilities can be combined with High-Performance Computing (HPC) state-of-the-art techniques. As a fact, many active projects are investigating and developing ExaScale supercomputers starting from the FPGAs' computational capabilities: interesting examples are the EU-H2020 EuroEXA Project [8] and the PC2 Noctua2 supercomputer at Paderborn University [9].

A clustered FPGA system might initially seem to introduce greater complexity, with potential performance benefits overshadowed by increased reliability challenges due to the multiplication of the possible fault locations. However, by effectively leveraging the cluster structure, it becomes possible to enhance the system's robustness and mitigate these reliability concerns.

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

A. The main contribution

In this paper, we propose a novel approach to enhance the fault tolerance of multi-FPGA systems, which revolves around three main contributions.

The first contribution is the design of a TMR Beacon Controller (TMR-BC), a reliability-oriented in-hardware monitoring module inherently robust against radiation-induced faults, which monitors local computational resources and maintains cluster-level synchronization by exchanging node status information across nodes via a robust-by-design hardware bus, the Robust Bus, specifically designed to prevent unwanted reconfigurations caused by radiation-induced bit flips on inter-node signals.

The second contribution lies in the error correction strategy: at the module level, Dynamic Partial Reconfiguration (DPR) selectively restores failing resources without interrupting the rest of the system; at the node level, a custom Second Stage Bootloader (SSBL) ensures full-node recovery by re-initializing the DDR memory subsystem, relocating partial bitstreams to the correct memory regions, and restoring the application execution in the on-board processors.

The third contribution is a workload remapping mechanism operating at both node and cluster levels: within a node, tasks from failing computational units are redistributed among the remaining functional ones; at the cluster level, the workload of a failing node is reassigned to the healthy nodes during reconfiguration through high-speed communication link based on Aurora protocol, ensuring continuous operation.

The proposed solutions have been integrated into the European FPGA Radiation-hardened Architecture for Telecommunications (EuFRATE), a FPGA cluster architecture developed in collaboration with the European Space Agency, the Argotec Company, and the Technical University of Dresden (TUD). A radiation test campaign has been performed on the developed architecture at the HollandPTC facility in Delft to verify the effectiveness of the proposed solutions. Finally, although the proposed solutions have been validated within the EuFRATE platform, they are designed to be device-agnostic: the TMR-BC is entirely VHDL-based and can be integrated into any FPGA design with minimal effort, while the SSBL requires only minor address-level adaptations to target different processing architectures and memory configurations.

The paper is organized as follows. Section II discusses the state-of-the-art about FPGA-cluster applications and the reliability studies about such systems, with particular regard to research works similar to the one presented in this paper. The main contribution of the work is described in Section III. Section IV is devoted to the experimental analysis, while Section V investigates and discusses it. Finally, Section VI presents conclusive considerations about the study and related future work and developments.

II. State of the Art

FPGAs excel in parallelizing algorithms and boosting computational throughput in mission-critical scenarios. However, each computational block is vulnerable to configuration memory corruption, often due to particle interactions. To ensure nominal function, rigorous verification is essential. In [10], a low-overhead method enhances FPGA reliability by using a hardwired processing unit to validate each block's operations via checksum logic, with BRAM storing validation results. A similar approach is exposed in [11], exploiting a soft-core processor.

A common approach to enhancing fault tolerance in programmable systems is to develop dedicated hardware modules for the target FPGA. These modules often adopt existing fault detection and correction features like configuration memory readback, scrubbing, and partial reconfiguration. For instance, [12] introduces a custom dynamic partial reconfiguration controller, whose reliability is evaluated in [13], which uses vendor-specific on-board components, such as the ICAP controller, to enable partial reconfiguration of the monitored modules present on the FPGA. The main limitation of this architecture is its inability to provide detailed information about the current health and status of the monitored modules, while just being able to notify the occurrence of a hard error after a partial reconfiguration is performed. This limitation can significantly reduce the benefits of partial reconfiguration, as it becomes impossible to determine precisely when a specific module has failed or has been reconfigured. An improved controller with Coarse-Grained Triple Modular Redundancy (CGTMR) is proposed in [14], but the same limitation previously mentioned is still present.

An alternative approach, as in [15], uses an external rad-hard FPGA to improve system lifetime when the main application is implemented on a non-rad-hard COTS FPGA. While effective, this setup incurs high costs and data transfer bottlenecks, which can be critical, as confirmed in studies like [16] and [1]. These show that radiation-induced Single Event Upsets (SBUs) and Multi-Bit Upsets (MBUs) can significantly affect data-transfer modules, potentially causing system failures.

Several notable works have contributed to reliability and cluster-aware fault tolerance in FPGA systems. In [18], [19] and [20], the authors presented an initial approach to a fault-tolerant controller that incorporates FPGA protection techniques such as configuration memory scrubbing, partial reconfiguration, and functionality relocation in response to permanent faults. While effective as a reliability-oriented controller, this solution was implemented on legacy Xilinx Virtex-IV FPGAs, which differ significantly from modern devices and lack a full reliability assessment, making it difficult to evaluate its robustness for current applications.

Radiation testing experiments investigated the dependability of FPGA clustering in high-radiation environments, where devices must reliably monitor and instrument complex events and systems. To ensure continued functionality, [21] describes a custom configuration memory scrubber for data acquisition systems, leveraging majority voting across a heterogeneous multi-FPGA cluster. In [22], a high-performance Internal Configuration Access Port (ICAP) controller is introduced, designed to withstand radiation-induced SEUs through a heterogeneous TMR structure. However, error detection for the

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

target modules requires a separate external function, adding complexity to the fault-tolerance approach.

Although with some similarities, our approach differs from [15], relying on a central controller tailored to the specific role of each peripheral system. Moreover, in contrast to [1], our solution is based on a homogeneous multi-FPGA cluster, where each node features identical hardware and FPGA design. The closest approach to the solution proposed in this paper is found in [23], where the authors describe a system for space exploration missions, comprising a heterogeneous cluster with multiple hardwired processors and FPGA devices relying entirely on COTS components. The FPGAs manage computational units, with a VHDL-based controller to monitor node health, reboot on failure, and include/exclude units based on their status. Unlike this architecture, which uses external processors that may lack radiation resilience and add complexity to centralized monitoring, our approach employs a fully FPGA-based solution without external controllers or computational units. This ensures comprehensive monitoring and fault tolerance within the FPGA system itself, with built-in protections like self-checking and redundancy to mitigate SEUs in critical functions.

The proposed work introduces a versatile and intrinsically reliable TMR Beacon Controller designed for seamless integration into complex multi-FPGA systems. The architecture combines fine-grained in-hardware monitoring with a Robust Bus, ensuring fault-tolerant exchange of status and health information across the cluster. By integrating this monitoring layer with a multi-tier recovery strategy, leveraging Dynamic Partial Reconfiguration (DPR) for localized failures and a custom Second Stage Bootloader (SSBL) for full-node restoration, the system can execute autonomous error correction. Furthermore, a dedicated workload remapping mechanism ensures that system throughput is maintained across the entire cluster, effectively eliminating downtime and minimizing performance degradation even under high radiation-induced fault rates. Applying the proposed solution to the architecture of an FPGA-cluster system, carried on by ESA, and performing a radiation proton test at the HollandPTC facility that verifies the effectiveness of the proposed reliability approaches. In the next sections, these contributions are analyzed and discussed in depth, with a comprehensive reliability analysis integrating fault-injection results with interesting data retrieved from the radiation test.

III. The Proposed Reliability-aware Architecture

Operating in harsh environments exposes FPGA-based systems to extreme conditions that can lead to critical faults. Among these, radiation-induced malfunctions are particularly common in space missions, arising from the interaction between particle radiation and the programmable memory devices used in FPGAs. These memories store the data programming the circuit necessary for proper operation, making them inherently vulnerable to errors caused by radiation events. To address this challenge, both design-agnostic and design-tailored fault mitigation techniques are typically employed. This section details the proposed fault-tolerance architecture, focusing on the TMR Beacon Controller, workload redirection capabilities, and recovery mechanisms to restore correct operation after fault detection.

A. The TMR Beacon Controller

The proposed mitigation solution involves the design of a hardware monitoring module, the TMR Beacon Controller (TMR-BC), which is an intrinsically mitigated module tailored for FPGA implementation. The TMR-BC integrates seamlessly with the target architecture and operates non-intrusively to continuously monitor and control designated modules. In the event of an error, the module generates precise error signals, identifying the affected functions or modules and enabling the exclusion of the faulty component from the system's operation.

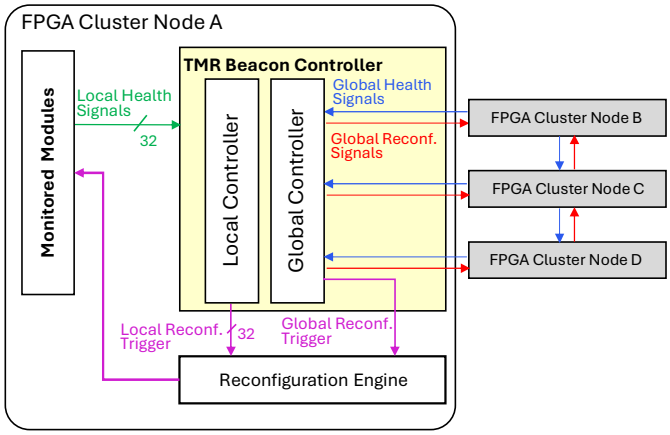


Fig. 1 The developed Beacon Controller architecture.

Since the hardware monitoring module is implemented within the FPGA fabric alongside the rest of the design, the Beacon Controller is also susceptible to radiation-induced faults that could compromise its functionality. To ensure maximum reliability, the proposed architecture incorporates fine-grained Triple Modular Redundancy (TMR) within the Beacon Controller's internal structure.

This approach enhances its robustness against radiation-induced bit flips, preserving its correct operation even under adverse conditions.

The internal structure of the TMR-BC embeds a set of Control and Status registers, which are accessed via software to initialize the device and collect information about the health of monitored modules. To avoid unwanted access to such registers, especially when performing write operations, they have been protected through a pattern lock-unlock procedure.

It is important to highlight that the TMR Beacon Controller can monitor an FPGA cluster at two different levels: at the node level, it can check the status and health of the internal resources within the node; at the cluster level, it can validate the status of the other nodes in the cluster, also called remote nodes. To perform such operations, the Beacon Controller embeds two sub-modules: the "Local Controller" and the "Global Controller". Fig. 1 reports the general scheme of the TMR-BC design, highlighting interfaces and status signals within a node and towards the external devices.

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

TMR-BC Structure: the Local Controller

The primary task of the Local Controller (LC) is to monitor the internal computational resources within a node, ensuring the integrity and functionality of the local system.

The detection of erroneous behaviors is achieved through the implementation of a watchdog mechanism. Each monitored resource is assigned a dedicated timer, and each device must trigger a signal before its timer expires. If a resource fails to reset the timer before it expires, a functional interruption is triggered, and the module is marked as faulty.

Each resource resets its timer and communicates its health status by toggling a dedicated health signal directly connected to the TMR-BC.

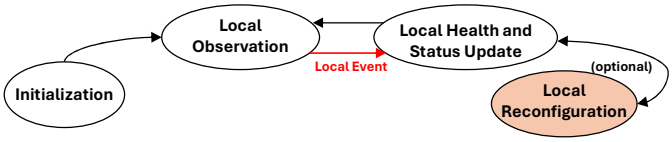


Fig. 2 The Local Controller FSM scheme.

Fig. 2 details the Finite State Machine (FSM) on which the Local Controller works. It implements four phases:

- *Initialization*: During this phase, all status and control registers in the system are initialized, and the timers for each of the monitored units are set to the desired value. This phase occurs either at the startup or after a system reset.
- *Observation*: After initialization, the FSM transitions to the Observation phase, during which the Local Controller actively monitors the resources, awaiting any interrupt triggers.
- *Health and Status Update*: When an interrupt occurs, the Local Controller transitions to the Health and Status Update phase, during which it updates the internal status registers and notifies the Global Controller about the status of the local node.
- *Reconfiguration*: When a resource fails, the TMR Beacon Controller enters the Reconfiguration Phase and sets the corresponding failure signal. This signal must be collected by other modules, which will handle the specific recovery action based on the system design. A possible solution, and the one adopted in our case, is to trigger the Partial Reconfiguration of the failing unit through the Dynamic Function Exchange (DFX) controller. Following the recovery process, the system returns to the Health and Status Update phase.

The Local Controller's behavior can be modified and adapted as needed through its control and status registers, which are configurable via specific software instructions. This device was developed to monitor up to 32 internal local resources simultaneously, each with its dedicated timer and health signal. Limiting the number of monitorable resources to 32 represents a good balance between device flexibility and hardware constraints, as each reconfigurable partition must be assigned to a dedicated FPGA region to allow partial reconfiguration. As the number and the size of partially reconfigurable modules increase, the more constrained the Place And Route (PAR) procedure becomes, possibly leading to implementation failures. However, if partial reconfiguration is not feasible or alternative recovery strategies are adopted, few simple hardware modifications are required to increase the number of monitoring channels.

As detailed earlier, each monitored module must periodically toggle its corresponding signal to the Beacon Controller. The way of achieving this depends on the use case and may require hardware or software modifications in the monitored module, such as using additional GPIOs or other dedicated signals. This mechanism can also be effectively applied to black-box IPs or encrypted netlists, provided that their output signals can be routed to the BC to serve as an indirect health indicator. When an IP produces no observable outputs or does not expose any signals that reflect its operational state, the TMR-BC cannot directly monitor it. Additionally, it is possible to exclude one or multiple resources from being monitored. This can be useful in various situations, for example, when such resources are still being initialized.

Finally, it should be noted that a false-positive reconfiguration request may trigger an unnecessary partial reconfiguration of the corresponding module. While this might represent a minor availability overhead, its impact is mitigated by the workload remapping mechanism, which ensures the system continues to operate even during the reconfiguration window. Furthermore, the fine-grained TMR structure of the TMR-BC itself significantly reduces the likelihood of such spurious triggers, as demonstrated in the experimental results analysis.

TMR-BC Structure: the Global Controller

The second sub-system that is implemented in the TMR Beacon Controller is the Global Controller (GC), whose general FSM is reported in Fig. 3. The main goal is to monitor the health of the cluster by cooperating with the Global Controllers deployed on the other nodes of the cluster, thereby enhancing the reliability of the system, also at the cluster level, rather than only at the node level. Specifically, the GC checks the health and status signals of other nodes in the cluster and ensures that other nodes are aware of the local node's health status. Additionally, the GC maintains constant communication with the internal Local Controller.

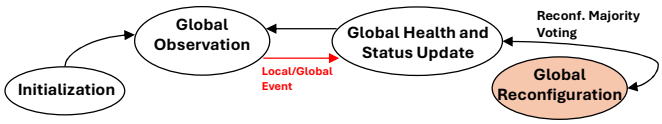


Fig. 3 Global Controller FSM Scheme

These operations become critical when a node in the cluster fails. In such cases, all Global Controllers within the cluster are notified of the failure through the health status exchange mechanism. The central node, which is responsible for nominal workload distribution, receives this information and proceeds to reassign the tasks previously handled by the failing node to the remaining healthy ones. It is important to clarify that the workload redistribution procedure is not directly managed by the Global Controller itself, as it is strictly dependent on the topology of the system, such as the type of interconnection within the cluster and the way data is managed by the specific

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

application. Instead, it is carried out by the other nodes in the cluster, as detailed in the following section. After the failing node completes reconfiguration and re-establishes communication with the other nodes in the cluster, nominal operation is resumed. Another essential function of the Global Controller is the management of global reconfiguration triggers. Each Global Controller has access to a signal for each of the other nodes composing the cluster.

These triggers enable the reconfiguration of a failing node through a voting mechanism. Specifically, the Global Controller of each node can vote to reconfigure a remote node if the health and status signals are not received correctly by setting the corresponding reconfiguration signal. A full-node reconfiguration is initiated only if the majority of the functioning nodes in the cluster agree on the failure. For instance, in a four-node cluster, at least two out of three nodes must indicate the fourth as faulty for reconfiguration to proceed.

To further reduce the risk of unwanted reconfigurations caused by radiation-induced corruption of the inter-node signals themselves, a dedicated hardware bus, the Robust Bus, has been designed to handle all health and reconfiguration signal exchanges between Global Controllers. Rather than relying on raw binary signals, which are inherently vulnerable to stuck-at faults induced by radiation on FPGA I/O pins, the Robust Bus encodes each signal using a one-hot scheme. This design choice maximizes the reliability of inter-node communication for node reconfiguration control signals in a radiation-prone environment. Three valid states are defined for the health signal (b001 nominal, b010 error detected, b100 reconfiguring), and two for the reconfiguration trigger (b01 nominal, b10 reconfiguration requested). Any other combination is treated as invalid, and the originating node is closely monitored: if the anomaly is not resolved autonomously within a specified time interval, a full-node reconfiguration is triggered. The communication mechanism is entirely asynchronous, allowing signals to be sampled at any time and simplifying synchronization requirements across the cluster. However, while the increased signal redundancy enhances robustness against SEEs, the number of signals can escalate rapidly with the number of nodes in the cluster, potentially complicating the Robust Bus implementation. In that case, a custom Printed Circuit Board (PCB) must be designed and manufactured to simplify the connections and ensure easy integration of the Robust Bus into the system. For a limited number of nodes, such as in our case of study, the pin-out usually offered by COTS boards and FPGAs is sufficient.

By combining the majority-voting mechanism of the Global Controller with the fault-tolerant signal encoding of the Robust Bus, the probability of single points of failure causing unwanted node reconfiguration is essentially nullified. Additionally, if a single point of failure within the local controller triggers consecutive, unnecessary reconfiguration requests for an internal module, the global controller would log the event in its internal registers and on the global status signals, eventually triggering a full-node reconfiguration request from the neighboring nodes via the Robust Bus. Although such a scenario results in a temporary loss of node-level functionality, it represents a fundamental failure within the BC structure that requires a full-node reset to ensure complete restoration of the monitoring integrity. Nevertheless, the workload remapping strategies detailed in the following section ensure that the overall cluster remains operational and maintains service continuity even during these exceptional recovery events.

B. Error correction mechanisms

To ensure high performance and operational reliability in complex FPGA clusters, recovery mechanisms must scale with the severity of detected errors. Instead of relying solely on time-consuming full-node reboots, modern systems leverage Dynamic Function Exchange (DFX). This technology enables the definition of Reconfigurable Partitions (RPs), allowing specific hardware modules to be independently reconfigured via Partial Bitstreams without interrupting the rest of the system. In AMD-Xilinx architectures, this process is managed by a DFX Controller IP, which fetches partial configuration data from accessible memory, like DDR in our case, and uploads it to the configuration memory via dedicated interfaces, such as the Internal Configuration Access Port (ICAP).

The TMR Beacon Controller (TMR-BC) integrates directly with this DFX infrastructure to minimize performance degradation. At the node level, the Local Controller can autonomously trigger partial reconfiguration for up to 32 monitored modules. By targeting only the specific module where an error is detected, the system achieves a fast, low-overhead repair that restores functionality in milliseconds while the node remains active. However, when local module-level repairs are insufficient, or the module does not recover properly, a full node reconfiguration is performed, managed at the cluster level by the Global Controllers through majority voting via the Robust Bus. Performing a full node reboot is particularly critical, not only because the system must be reset from scratch, but also because it is necessary to ensure that DDR memory is properly reinitialized so that partial bitstream locations remain at fixed, known positions for the DFX controller. To manage this, a dedicated Second Stage Bootloader (SSBL) has been developed, with the full node image stored within the node's Flash memory. This image embeds the main bitstream, with the main processor embedding the SSBL executable. Upon a full reset, the built-in First Stage Bootloader (FSBL) loads the base bitstream, which immediately triggers the SSBL on the target processor to fetch the full application from Flash and extract the embedded bitstream headers into a dedicated shared region of the DDR. By relocating these bitstreams to predefined addresses, the SSBL ensures that the DFX Controller is properly re-armed and ready to resume monitoring and repair duties immediately after the reboot, effectively restoring the node's self-healing capabilities. Once the SSBL finishes execution, the processor begins execution of its main task.

C. Workload remapping strategy

The workload remapping strategy is designed to operate seamlessly at both the node and cluster levels, ensuring that computational performance is never compromised by recovery events. At the node level, when the Local Controller detects a failure in one of the internal processing units, it immediately notifies the onboard processor. While the faulty module is partially reconfigured, the processor autonomously redistributes its pending tasks among the remaining functional

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

units. This remapping occurs in parallel with the execution of the main application, effectively absorbing the loss of one unit within the node's internal resources.

If a critical failure affects the entire node, the recovery strategy is scaled to the cluster level. The mesh-based topology provides high-bandwidth, parallel inter-node connectivity through a communication subsystem based on the Aurora protocol. In this architecture, the communication logic is tightly integrated with the system memory via Direct Memory Access (DMA) engines, allowing for high-speed data transfers without constant processor intervention. To prevent the central management node from becoming a single point of failure, a redundant master configuration is implemented. In this setup, two nodes are designated with leadership capabilities: a primary Leader and a secondary Follower.

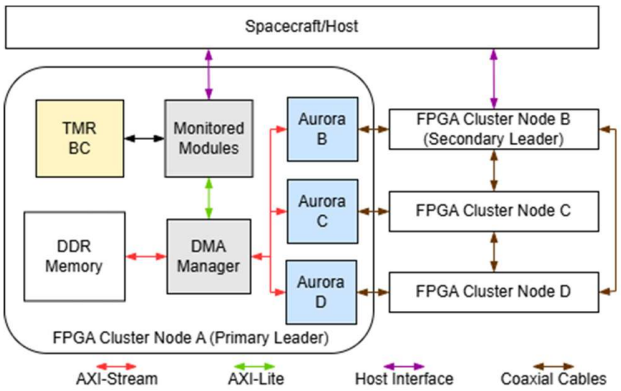


Fig. 4 The implemented Aurora-based Mesh topology.

Upon detecting a node failure, the active leader node manages the workload redirection through the communication subsystem and DMA. If the primary Leader node fails, the secondary master immediately assumes leadership, ensuring the cluster-wide remapping logic remains operational. By leveraging dual-master redundancy and the parallel nature of the mesh topology, the system can rapidly relocate the necessary computational state, maintaining high operational throughput. This mechanism effectively masks the 16-second full-node reboot (SSBL) from the global system, ensuring the recovery process remains transparent and results in zero observable downtime at the cluster level. In Fig. 4, the mesh architecture employed to properly manage this workload redirection strategy at cluster level has been reported.

IV. The Experimental Analysis

This section presents an in-depth analysis of the experimental validation conducted to assess the proposed fault-tolerant architecture's effectiveness in mitigating radiation-induced faults within multi-FPGA clustered systems. The proposed solutions were implemented and tested within the EuFRATE project, which serves as a practical platform for validating these mitigation techniques, targeting radiation-hardened designs for satellite telecommunication payloads in geostationary orbit using Commercial-Off-the-Shelf FPGA devices. The experiments include comprehensive fault injection campaigns and radiation testing targeting node-level and cluster-level reliability. The key experimental results demonstrate the robustness of the proposed mitigation strategies in ensuring system reliability and operational availability.

A. The EuFRATE Design

All the proposed mechanisms have been integrated in the EuFRATE research project, which is the result of the collaboration between Politecnico di Torino, the European Space Agency (ESA), the aerospace company Argotec (Turin, Italy), and the Technical University of Dresden (TUD). The goal of the project is to develop and test a radiation-hardened design for a satellite telecommunication payload, to be deployed in the Geostationary-Earth Orbit (GEO) and using Commercial-Off-The-Shelf FPGA Devices. The overall computational system is structured as a distributed system made of four tiles, each tile consisting of four boards, each one embedding one FPGA device, called a node. Both between tiles and nodes, a full-mesh communication-based protocol has been adopted.

The COTS device chosen for the implementation of the EuFRATE cluster is the AMD-Xilinx XCKU040 FPGA. This device represented the most balanced alternative between FPGA size, cost, available IPs, and external interfaces. Smaller devices would have limited the number of instantiated components and made partial reconfiguration more challenging due to the constraints it imposes. Additionally, the large number of the board's external interfaces enabled the implementation of a large cluster with fast and efficient communication channels between nodes.

The project allowed validating the effectiveness of the proposed mitigation techniques and solutions in addressing radiation-induced faults and errors, targeting both the *node* and *tile* levels.

In Fig. 5, the actual mounted system is illustrated, showing a tile of the cluster composed of four nodes interconnected via several links, including the data and control channel and the reliability-oriented channel, the Robust Bus.

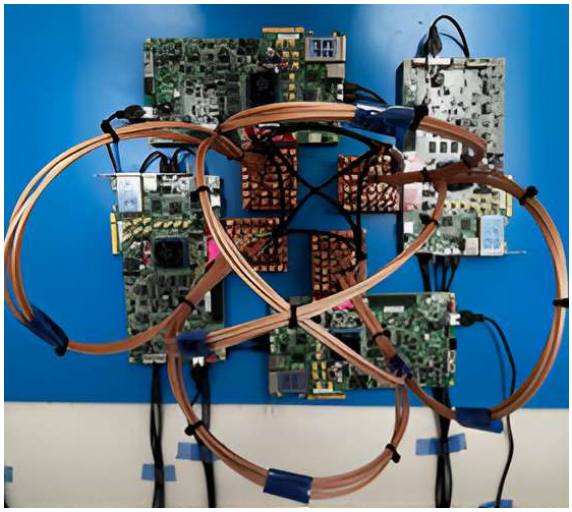


Fig. 5 The mounted EUFRATE System: a tile including four nodes.

At the node level, each FPGA embeds three main computational cores:

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

1. A TMR Microblaze architecture, called *TOWER*, performs cluster management and workload distribution tasks.
2. A non-hardened Microblaze soft processor running Free-RTOS, called *GaRDeN*, performs several data-related tasks and manages the flow of data inside the node between the computational modules.
3. The third element consists of a set of accelerators to perform the majority of heavy telecom coding/decoding tasks, named the Programmable Functional Unit (PFU) cluster.

To enhance the robustness of the design and ensure the continuous operability of the system, several mitigation techniques were adopted, starting from the ones detailed in the previous section. Each node embeds a TMR Beacon Controller and a DFX core to perform the Partial Reconfiguration of the monitored modules. Additionally, these solutions were paired with the AMD-Xilinx SEM-IP Core [24], which constantly performs configuration memory scrubbing and error correction.

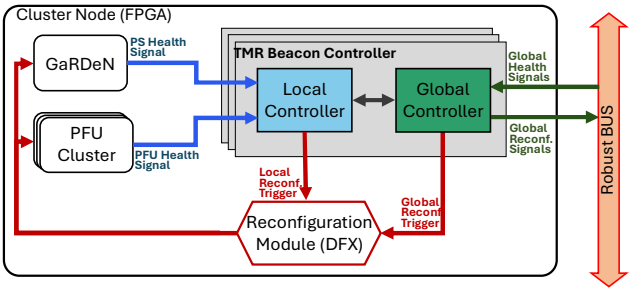


Fig. 6 EuFRATE Node Architecture

In Fig. 6, a conceptual scheme is reported to describe the internal structure of each node FPGA design, highlighting the use and integration of the TMR Beacon Controller and the Robust Bus. As can be seen in the figure, the two main computational modules have been selected for being monitored by the Controller, being the GaRDeN soft-processor and the cluster of PFUs. Firstly, both these modules have been slightly adapted to provide the needed health toggling signal towards the TMR Beacon Controller. Secondly, further minor actions took place, allowing the reconfiguration of these modules and therefore exploiting the integration between the proposed error detection system and the correction approach involving the Dynamic Partial Reconfiguration.

The described configuration, characterized by several mitigation layers, allowed the design of a clear, efficient, and modular recovery policy.

The overall strategy consists of a tradeoff between time and computation overhead. When one of the monitored modules fails repeatedly over time, the software application can decide whether to reconfigure the module again or ignore its failure state, possibly excluding the unit from the computational process. This scenario can become effective in case of a hard error in some of the FPGA resources used for a certain monitored module: in such case, instead of reconfiguring the whole node, inevitably interrupting the mission tasks, through the TMR Beacon Controller the system is capable of comprehending if DPR is resolving the failure and, possibly, try to apply some different recovery measures. Most importantly, during these recovery operations, the rest of the cluster will always be aware of the state of the node, thanks to the Global Controller and the Robust Bus. In fact, through the GC, the current state of the node will be constantly updated based on the actual node situation – whether it is "OK", "NOK" or "REC" (reconfiguring) – and this information will be shared with the other cluster nodes quickly and reliably through the Robust Bus feature.

Thanks to the proposed solution, the shared cluster health state awareness could also be exploited at the application layer to manage the workload and the tasks to be carried out efficiently and safely. For instance, considering that one out of four nodes of the cluster is currently trying to resolve a failure of one of its PFUs, the rest of the nodes will be aware of this system status, and they could perform additional checks on the data produced by the node. Furthermore, at the same time, they could check the elapsed time since the last "OK" state, voting for the target node reconfiguration if that period exceeds a certain threshold.

B. Fault Injection Campaigns

To validate the effectiveness and robustness of the proposed solution, multiple separate fault injection (FI) campaigns have been conducted on an actual hardware system implementing the EuFRATE architecture. Before describing the results obtained from these analyses, it is worth describing the considered fault model and the differences between the fault injection methods exploited during the process.

Fault model

Operating computational systems in harsh radiation environments, such as space, exposes FPGA-based devices to different physical phenomena. While factors like temperature excursions, voltage instability, and aging are typically mitigated through mission-specific hardware engineering, such as thermal management and robust power-supply design, radiation-induced effects present a more persistent challenge to digital logic.

In SRAM-based FPGAs, the primary reliability concern is the corruption of the Configuration Memory. Regardless of the source type, being radiation particles the most common one, these events manifest as bit-flips, commonly known as Single Event Upsets (SEUs) [25]. Since this memory defines both the functional logic and the routing resources of the device, even a single SEU can alter the implemented circuit's behavior [26]. Consequently, corruption of these memory elements can lead to a wide range of malfunctions, from localized data errors within specific flip-flops that may propagate and cause system-wide failures, to the shutdown of the entire device.

To understand if an FPGA-based system is sufficiently robust against SEUs, we have considered two main hardware fault injection methodologies:

1. The first method aims at the alteration of the configuration memory content after its initialization, so during the nominal operation of the implemented system.
2. The second one consists of programming the device with a manipulated *bitstream file* to initialize the content of the configuration memory.

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

Fault injection experimental results

To validate the reliability and fault-detection capabilities of the TMR Beacon Controller, two fault-injection campaigns were carried out.

The first campaign assessed the reliability of the controller itself, since integrating an in-hardware mitigation module must not introduce an additional weak point into the system. To evaluate the controller under worst-case conditions, the campaign targeted its essential configuration bits, which are identified by the vendor as the most critical bits of the FPGA configuration memory, whose corruption is more likely to affect the implemented design.

The relevant essential bits were identified using the PyXEL[27] framework, together with the Essential Bits Data (EBD) and Essential Bits Content (EBC) files provided by AMD. PyXEL enables the mapping of configuration bits to specific design resources, allowing selective fault injection into the bits associated with the TMR Beacon Controller.

Since all essential bits belonging to the controller were exhaustively tested, the large number of fault locations required a run-time injection approach. Therefore, faults were injected after configuration memory initialization, through the ICAP interface.

TABLE I
FAULT INJECTION ANALYSIS TARGETING TMR BEACON CONTROLLER

Total injected faults		**821,695**	
Success (fault masked)		788,814	96%
SDE masked by TMR		13,215	1.61%
Failure	Controller Halt	10,638	1.29%
	Silent Data Error	9,028	1.10%

The obtained data are reported in Table I. In these numbers, there are two main key points to be noted. First, the controller operates nominally in 96% of the injected cases, confirming its robustness against radiation-induced SEU effects. Second, the internal TMR structure effectively masks several erroneous internal states. In particular, some faults affecting the controller registers are prevented from propagating to the output and, therefore, do not become effective system-level errors.

A second fault-injection campaign was conducted on a monitored module to evaluate the TMR Beacon Controller's detection capabilities. In this case, faults were injected by programming the device with corrupted bitstream files. The GaRDeN MicroBlaze processor was selected as the representative monitored resource, and random faults were injected into the configuration bits associated with its implementation, targeting both essential and non-essential ones. To ensure that only the bits belonging to the selected module were targeted, the PyXEL framework was again used.

TABLE II
FAULT INJECTION ANALYSIS TARGETING GARDEN SOFT-CORE

Total injected SEU in CRAM		13,847	
Masked (No Error to output)		10,960	79.15%
Failure (Detected)	FreeRTOS Exception	273	1.97%
	Detected Core Halt	2280	16.47%
Failure (Undetected)	Undetected Core Halt	82	0.59%
Other Errors	Silent Data Error	252	1.82%

Table II reports the results of the fault injection experimental analysis targeting the GaRDeN soft-core processor. The number of faults injected into the device was determined using the formula (1) from [28], ensuring a 99% confidence level with a maximum error of 1%. Given a total of N=1,172,185 possible bits associated with the module under test, of which 522,734 were categorized as essential ones, an error margin of e=0.01, an expected error rate of p=0.5, and a confidence level of t=0.99, the minimum required number of fault injections to be performed was approximately 2,445. However, to enhance our analysis, we extended our investigation by injecting faults into at least 1% of the total bits associated to the monitored module, granting a good balance between the number of injected faults and the time required to obtain the results. The faulty outcomes for both injections campaigns have been categorized into four groups:

1. The *Halt* condition in which the core stops any operation without the possibility of recovery and therefore blocks the overall system activities.
2. The *FreeRTOS Exception* refers to the condition in which the FreeRTOS operating system running on the Microblaze processor stops operating correctly, causing a software halt.
3. The *Silent Data Error* condition occurs when the core executes without hanging but produces incorrect outputs.
4. In the *Masked* case, despite having injected a fault, the system behaves normally.

It is important to note that of the 2,635 faults that led to hardware or software halts in the MicroBlaze processor, 2,553 (96.9%) were successfully detected by the TMR Beacon Controller, demonstrating its high efficiency in halt detection.

Whereas the TMR-BC effectively detects halt conditions, Silent Data Errors are not. In fact, these events do not necessarily cause a module halt: the affected resource may continue operating and updating its watchdog signal while producing incorrect data. Detecting SDEs, therefore, requires application-specific mechanisms, such as duplication with comparison, TMR, or dedicated consistency checks. Although SDEs remain a significant reliability metric, their impact can be effectively mitigated by integrating dedicated protection mechanisms within the target module. For this reason, halt-related failures represent the primary fault class addressed by the proposed TMR-BC monitoring approach, but SDE faults have also been reported for completeness.

Finally, before conducting the radiation test, a fault-injection campaign was performed to evaluate the system's reliability and provide an initial estimate of its expected availability. In both the irradiated design and the fault-injection setup, only the PFU

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

cluster and the GaRDeN processor were monitored by the TMR-BC. This choice was motivated by the placement constraints required for partial reconfiguration: each monitored resource must be implemented as a dedicated reconfigurable partition, limiting the overall routing flexibility of the design. Therefore, a system-level fault-injection campaign was carried out to evaluate the reliability of all main modules, while the TMR-BC was configured to monitor only the PFU cluster and the GaRDeN processor. The error rates obtained for each module are reported in Table III.

As shown in the table, the triplication of the Tower processor ensures very good reliability results, with an overall error rate below 0.2%. For the PFU cluster and the GaRDeN processor, the results confirm the effectiveness of the TMR-BC in detecting halt-related failures and enabling the corresponding recovery actions. For the other, non-monitored modules, the aforementioned results indicate the potential benefit that TMR-BC monitoring could deliver if these resources were also implemented as reconfigurable partitions and connected to the controller. In that case, most halt-related failures could be detected and properly managed, significantly reducing their impact on system availability.

Regarding SDEs, as discussed above, their mitigation requires dedicated module-level protection strategies. For example, the PFU architecture's internal redundancy can be configured to implement Duplication with Comparison (DWC) or TMR, allowing data-level errors to be detected or masked before propagating to the system output.

TABLE III
EUFRATE PLATFORM FAULT INJECTION RESULTS

Failure Mode	Error Rate [%]	BC Detection Rate [%]
Tower SDE	0.04	-
Tower Hang	0.15	-
Garden Hang	3.33	96.2
Garden SDE	2.57	-
MailBox Hang	0.04	-
MailBox SDE	0.35	-
PFUs Hang	0.01	95.3
PFUs SDE	23.36	-
DDR Controller Hang	5.38	-
DDR Controller SDE	3.79	-

Alongside the fault injection campaign, an environmental analysis was conducted to evaluate the expected SEU and failure rates during the whole mission duration, especially in the solar minimum, solar maximum, and worst day conditions. By combining these data, obtained using the CRÈME96 tool, with the data from the fault injection campaigns, it was possible to estimate the availability of the service and observability subblocks, as highlighted in Table IV.

TABLE IV
EUFRATE AVAILABILITY ESTIMATION

		Worst Day [%]	Solar Minimum [%]	Solar Maximum [%]
Observability	Availability	99.95	≈100	≈100
	Unavailability	0.05	8.97E-12	5.5E-12
Service	Availability	96.68	≈100	≈100
	Unavailability	3.32	8.30E-09	5.1E-09

The experimental results show three different analyses related to Solar minimum and maximum conditions and the Worst day scenario. As expected, the worst possible performances are registered in the latter case, with an availability that drops to 96%. In Solar minimum and maximum conditions, the availability remained higher than 99%, corresponding to the expected rates for which the system was designed. It is important to highlight that, during this analysis, no reconfiguration, either DPR or full-node reboots, were considered.

C. Radiation Test Validation

In addition to the fault injection campaigns, a radiation test was conducted to validate the proposed methodology in a real-world scenario. Two different configurations were tested: the first focused on a single-node version of the design, while the second involved the full-tile system, comprising all four nodes and the spacecraft emulator. This dual approach enabled the validation of the implemented mitigation techniques and demonstrated the system's capabilities in the event of node failures.

Radiation Test Setup

When electronic components have to be tested through radiation tests, preliminary decisions have to be made depending on the target application. When the requirement is to validate the system reliability against Single Event Upsets, or Single Event Effects, and in general, one of the most suitable sources of effects, and the one selected in our case, is protons. Since the EuFRATE project targets a 15-year mission in GEO orbit, additional analyses were conducted to be able to perform an accelerated radiation test capable of simulating the estimated effects of the entire mission duration within just a few hours. For this reason, we used a single energy level of 120 MeV, which allowed us to achieve a rate of ≈1 SEU per second throughout the entire test duration. It is important to highlight that the radiation test for device characterization involves characterizing the device across multiple energy levels. In this case, our objective was not to assess the device's response to different radiation conditions but rather to validate the application's behavior against the specific effects expected in GEO over a 15-year mission. Given this goal, conducting the test at a fixed energy level of 120 MeV was a reasonable choice, allowing us to simulate the impact of radiation-induced events

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

in an accelerated timeframe. The actual test flow has been reported in Fig. 7.

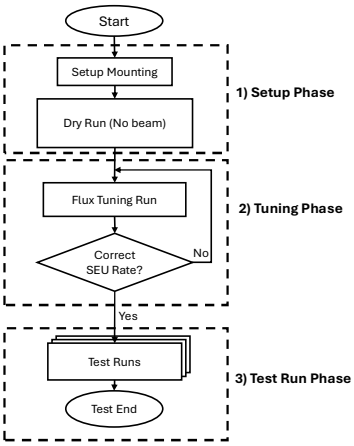


Fig. 7 Radiation Test Flow

Three different phases can be identified:

- *Setup Phase*: During this phase, the system was mounted in the radiation room, and suitable connections were performed with the computers in the control room to be able to monitor the entire system during the tuning and test phases. A schematic describing these connections is highlighted in Fig. 7.
- *Tuning Phase*: The expected SEU rate for conducting the experiment was ≈1 SEU/s; thus, this phase was required to tune the flux to achieve such a value.
- *Test Runs*: At the end of the tuning phase, performed only once for the whole test campaign, the actual test runs were executed. The system was irradiated while performing its nominal operations, and all the test data, in the form of log files, were collected and stored for further analysis.

In Fig. 8, we highlighted the actual setup that was mounted in the radiation and control rooms during the test. For the full-tile case, only one of the four boards was exposed to the radiation beam, while the others were positioned as far as possible from the beam zone to avoid secondary particles hitting them, thus compromising the collected data.

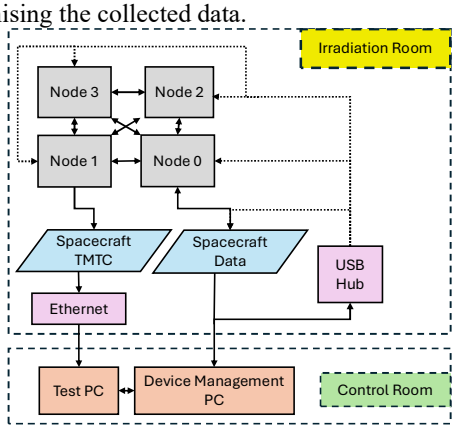


Fig. 8 Radiation Test System Setup

For the single-node case, instead, only one of the four boards was left connected, together with the spacecraft and the one managing telemetry, while the others were turned off. Table V reports the characteristics of the irradiation beam used during all the runs.

TABLE V
RADIATION BEAM PARAMETERS

Beam Size (cm)	Energy (MeV)	Flux	Fluence	Irradiation Time (s)
4x4	120	3E7	2.16E10	720

Radiation Test Result Analysis

Post-radiation test data analysis provided critical validation of system reliability, particularly when comparing the full-tile configuration to the single-node setup. The analysis focused on key metrics such as device cross-section, availability time, fault detection and correction performance, and error types. To facilitate this, data was collected from five different key sources during the test runs, including:

- *Garden Logs:* It captures the status updates and operational details of the Garden module, providing a record of system activities and events.
- *Tower Logs:* It records the data transmission to telemetry systems and identifies nodes that experienced failures.
- *Spacecraft Logs:* It tracks the number of processed codewords and the communication status between the spacecraft and the leader node.
- *SEM-IP Logs:* It records the fault detection and correction actions, offering insights into system recovery processes.
- *Telemetry Data:* It stores the availability status of each node in the tile, categorizing them as "OK" (operational), "NOK" (non-operational), or "REC" (undergoing reconfiguration).

The analysis of spacecraft and telemetry logs enabled the estimation of the overall system's availability. System's availability is defined as the period during the test when the leader node, the one under irradiation, maintained successful communication with the spacecraft and rest of the cluster, exchanging data and status updates with the tile. The availability time was calculated by subtracting the recorded unavailability periods from the total test duration. These findings are detailed in the subsequent sections.

TABLE VI
FULL TILE AVAILABILITY ANALYSIS

Fault Type	Node Failures	Detected Garden Failures (Undetected)	PFUs Failures (Undetected)
Occurrences [#]	4	33 (2)	3 (0)
Detection and Recovery Time [s]	16	2.06	0.023
Unavailability [s]	64	67.98	0.069
Total Unavailability [s]	131.98		
Expected Operational Time [s]	157,723,200 (5 years)		
Node Availability [%]	99.99992		

It is important to clarify that, although the TMR-BC is capable of monitoring a vast number of resources, this analysis specifically targeted the GaRDeN processor and the PFU

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

cluster for partial reconfiguration, due to physical FPGA constraints; instantiating additional reconfigurable partitions would have introduced excessive placement and routing constraints, compromising timing closure.

The experimental results for the Full-Tile configuration, reported in Table VI, detail the occurrences of node-level, GaRDeN, and PFU halts. The analysis demonstrates the high fault-coverage of the Beacon Controller: all PFU failures were correctly detected, while only 2 GaRDeN halts remained undetected. Such events, however, were treated as node-level faults and directly included in that category; since the GaRDeN processor handles local task coordination, its silent failure results in a loss of synchronization, necessitating a full-node reconfiguration to restore operations. Despite these individual node recovery periods, the cluster architecture ensures that the overall mission throughput is preserved through workload redirection to the other functional nodes.

Additional information can be gathered through the analysis of the SEM-IP log files. It is possible to classify all the detected faults into two main categories: *Corrected* and *Uncorrected ones*. The latter group includes, for example, ROM, ECC, and CRC errors that the SEM-IP was not able to correct. Correctable faults, on the other hand, not only could be corrected, but the SEM-IP also reports additional information, including its position in the configuration memory. This information allows for a further distinction of the corrected faults between essential and non-essential bits. The experimental results are reported in Fig. 9.

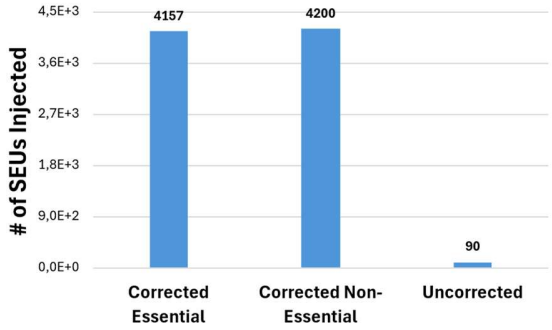


Fig. 9 The Full-Tile SEM-IP experimental statistics.

TABLE VII
SINGLE NODE AVAILABILITY ANALYSIS

Fault Type	Node Failures	Detected Garden Failures (Undetected)	Detected PFUs Failures (Undetected)
Occurrences [#]	17	15 (1)	2(0)
Detection and Recovery Time [s]	16	2.06	0.023
Unavailability [s]	272	30.9	0.046
Total Unavailability [s]	302.95		
Expected Operational Time [s]	78,883,200 (2.5 years)		
Node Availability [%]	99.99962		

About 99% of the faults were detected and corrected, while only 1.1% were classified as uncorrectable. Additionally, 49.2% of the corrected upsets were affecting essential bits, proving once again the resilience of the entire system against SEUs, even when they affect the most sensitive portions of the design. Combining the data regarding the number of upsets detected every run with the flux information coming from the facility logs, it was also possible to extract information about the cross-section of the device. This value has been computed concerning both the beam area and the bits in the configuration memory.

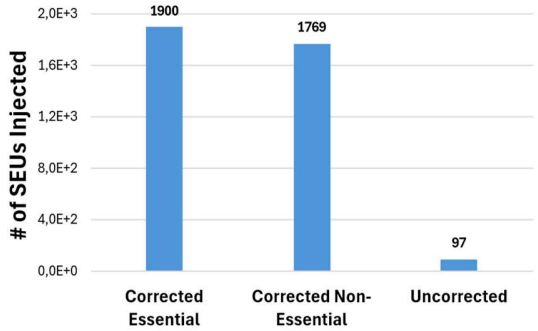


Fig. 10 The Single-Node SEM-IP experimental statistics.

As for the full-tile case, the single-node setup has been tested in the same way. In this configuration, only one of the four available boards was used while the others were disconnected from the tile. The single node would operate both as the leader node, managing data exchange with the spacecraft, as well as the main elaborating unit, managing the whole workload that was previously dispatched through four different nodes. While this setup naturally leads to increased elaboration time, the experimental results demonstrate that performance degradation is not the only disadvantage of a non-clustered architecture.

The post-radiation analysis results, reported in Table VII and Fig. 10, provide a direct comparison with the full-tile system, validating both the efficiency of the implemented mitigation techniques and the resilience of the clustered architecture. In a standalone configuration, any failure in the node results in a system halt. In contrast, within the full-tile environment, such events are managed as temporary node failures; the workload redirection protocol allows the data elaboration to proceed through the remaining functional nodes while the affected unit undergoes a partial reconfiguration or full reboot via the SSBL.

Regarding detection efficiency, the TMR-BC maintained consistent performance, with only one undetected GaRDeN halt and zero undetected PFU failures. As previously noted, the undetected GaRDeN failure led to a loss of local coordination and synchronization, necessitating a full-node reset to restore the node's operational integrity. Finally, the SEM-IP statistics for the single-node case align with the full-tile results, confirming that the ratio of corrected to uncorrected upsets and the distribution of essential vs. non-essential bit flips remained consistent across both test scenarios.

V. DISCUSSION ON EXPERIMENTAL RESULTS

The EuFRATE design and its proposed fault mitigation techniques have been successfully implemented on the AMD/Xilinx KCU105 development, which mounts the

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

XCKU040 FPGA device as detailed previously. The resource utilization for the entire design is presented in Table VIII.

TABLE VIII
EUFRATE SINGLE NODE RESOURCE USAGE

Design Module	LUT (%)	Registers (%)	BRAMs (%)	DSPs (%)
Tower	10,236 (4.2)	7,377 (1.5)	48 (8.0)	9 (0.5)
Garden	2,779 (1.2)	2,898 (0.6)	14 (2.3)	3 (0.2)
PFU Clusters	46,148 (19.0)	41,988 (8.7)	32 (5.3)	32 (1.7)
Aurora	14,964 (6.2)	26,706 (5.5)	91.5 (15.3)	0 (0)
TMR-BC	3,183 (1.3)	3,282 (0.7)	0 (0)	0 (0)
DFX	2,907 (1.2)	4,196 (0.9)	0 (0)	0 (0)
SEM-IP	344 (0.1)	458 (0.1)	4 (0.7)	1 (0.1)
Additional Resources	30,347 (12.5)	41,388 (8.5)	45.5 (7.6)	0 (0)
Total	110,908 (45.8)	128,243 (26.5)	235 (39.2)	45 (2.3)

It is evident that the proposed TMR BC occupies a negligible portion of the design area, requiring less than 2% of the available resources. This minimal footprint ensures high portability of the controller across different devices and clustered architectures, allowing it to be integrated without introducing strict constraints or limiting the implementation of the remaining logic.

A similar consideration applies to the communication infrastructure. Despite each node implementing three independent Aurora IP instances to manage the cluster's full-mesh connectivity, the impact on total resources remains extremely limited when compared to the overall system complexity. The majority of the logic is allocated to the computational components, specifically the PFU clusters, which constitute the accelerator's operational core.

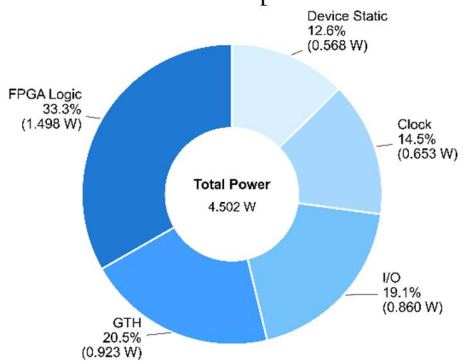


Fig. 11 The EuFRATE System Power experimental data.

Finally, although the total utilization remains below 50% of the FPGA's capabilities, the implementation is significantly influenced by the floorplanning required for the reconfigurable partitions. The placement constraints associated with the GaRDeN processor and the PFU clusters, which together occupy most of the design's logic, pose strict timing requirements that limit the overall routing capabilities.

Fig. 11 illustrates the estimated power consumption of nodes in the EuFRATE Cluster. It is evident that the FPGA Logics, which include resources such as internal signals, BRAMs, and DSPs, consume the most power. Due to the high number of external interfaces required for data exchange among the different nodes, components such as Transceivers and I/Os also draw a considerable amount of power. However, despite the design's complexity, we managed to maintain a relatively low power requirement to supply an entire node. The estimated power consumption for the whole cluster is around 20W.

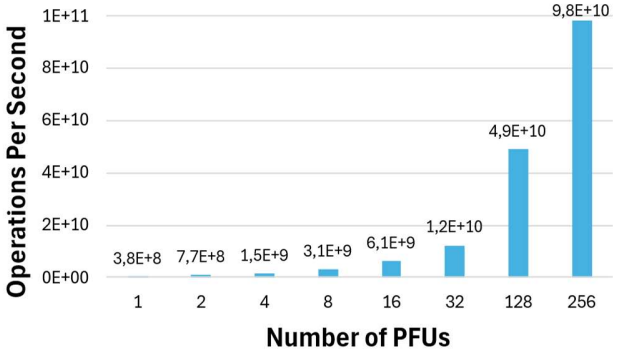


Fig. 12 The PFUs experimental performances.

Additionally, an in-depth analysis was conducted to evaluate the computational performance of the proposed design by combining the operational-time data obtained from the radiation test campaign with the processing capability of the PFUs. This allowed the number of operations executable by the cluster to be estimated under both nominal and faulty conditions.

Since the EuFRATE design operates at 100 MHz, each cluster of 8 PFUs achieves a throughput of approximately 3 GOPS, as shown in Fig. 12. With four clusters operating in parallel in each node, the overall node throughput reaches about 12 GOPS. Considering the LDPC-extended telemetry codeword encoding algorithm, the system processes approximately 1,600 codewords every 2.7 s under fault-free conditions, with each node processing 400 codewords and each cluster handling 100 codewords.

When faults affect one or more clusters, the effective throughput can be reduced to $1{,}600 - n \cdot 100$ codewords per processing batch, where n is the number of faulty clusters. However, the implemented mitigation techniques significantly limit this degradation. Since cluster reconfiguration takes approximately 10 ms and the worst-case error-detection latency is 20 ms, the maximum degradation interval is limited to about 30 ms. As each cluster processes around 37 codewords per second, at most two codewords may be incorrectly processed before recovery is completed, and this number can be lower if only a subset of the PFUs is affected. Failures affecting the GaRDeN processor, which manages data exchanges between the PFUs and the rest of the EuFRATE cluster, mainly introduce a delivery delay rather than data corruption. In this case, the maximum downtime is approximately 110 ms, while the clusters in the node continue operating correctly.

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

TABLE IX
COMPARISON WITH SIMILAR WORKS

	This Work	Preliminary Work (DATE2023) [29]	Sterpone et al. [20]	Bolchini et al. [18]	Giordano et al. [1]	McLoughlin et al. [23]	Panek et al. [13]
Target Device	AMD KCU105	AMD KCU105	AMD Virtex4	AMD Virtex4	Spartan6	N/A	Virtex-5
Cluster Type	Mesh Topology	Mesh topology	Single Node	Mesh topology	Point-to-Point	Dual Node	Single Node
Monitoring Logic	Full-TMR BC + Robust Bus	Preliminary BC unit	Reconfiguration Controller	Reconfiguration Controller	External scrubber	Node health monitoring	GPDRC + detection
Node Recovery	DPR+WL Redistribution	DPR	DPR+bitstream relocation	DPR + Logical relocation	Memory scrubbing	Node Reboot	Node-level DPR
Cluster recovery	Node Recovery + WL Redistribution	Not supported	Not supported	Inter-FPGA logic Redistribution	Inter-node majority voting	SW-based exclusion	Not supported
Resource Overhead	3282FF 3183 LUT	N/A	~4'000 Slices	~1'153 Slices	Extra device	N/A	~324 Slices (no TMR)
Controller Reliability	2.39%	N/A	N/A	N/A	N/A	N/A	3.97%
Detection rate	96.9%	~94%	N/A	N/A	N/A	N/A	N/A
Overall downtime	~10-30ms (DPR), ~16s (Full node)	~30-50ms (Theoretical)	~10-100ms (Relocation)	N/A	N/A	N/A	N/A

Finally, an additional analysis has been performed to compare the proposed approach with state-of-the-art architectures and with the preliminary results obtained in the earlier study, as reported in Table IX. Compared with existing solutions, the main advancement of the finalized EuFRATE architecture lies in the integration of reliable monitoring, recovery, and workload continuity within a unified cluster-aware framework. In particular, the proposed TMR Beacon Controller is intrinsically protected against radiation-induced faults and, according to the experimental campaign, reduces the controller error rate from 3.97% reported in [13] to 2.39%, while preserving high fault-detection effectiveness with limited hardware overhead. Beyond the controller, the architecture combines fault detection with workload remapping at both node and cluster levels. Unlike previous solutions, which typically restore functionality only after recovery is completed, the proposed system can reassign the workload of a module or node under reconfiguration to the remaining healthy resources. Therefore, reconfiguration latency is not directly translated into full system downtime, but is partially absorbed by the distributed architecture, improving operational continuity at the cluster level.

VI. CONCLUSION

This research introduces a novel mitigation approach that embeds state-of-the-art devices, such as DFX and SEM-IP, and combines them with custom architectures, the TMR Beacon Controller, the Robust Bus, and a dedicated workload redirection mechanism to enhance the reliability of a cluster-based reconfigurable FPGA system. The proposed architectures were tested through fault injection campaigns, proving extremely high reliability levels, and then inserted into the EuFRATE clustered architecture. The whole system performances were evaluated through proton-based radiation tests, during which the cluster granted more than 99% system uptime, and more than 98% of the induced faults were detected and corrected.

ACKNOWLEDGMENT

This study was partially carried out within the EuFRATE project, which received funds by the European Space Agency (ESA). Special thanks go to the Argotec group, Torino, Italy, through which we were able to implement our developed architecture in a real case application and test its efficiency and functionalities under real case considerations.

REFERENCES

[1] Patrick Michel et al 2022 Planet. Sci. J. 3 160DOI 10.3847/PSJ/ac6f52
[2] S. Azimi, C. De Sio, D. Rizzieri, and L. Sterpone, "Analysis of Single Event Effects on Embedded Processor," Electronics, vol. 10, no. 24, Art. no. 24, Jan. 2021, doi: 10.3390/electronics10243160.
[3] E. Vacca, S. Azimi and L. Sterpone, "Analyzing the SEU-induced Error Propagation in Systolic Array on SRAM-based FPGA," 2023 23rd European Conference on Radiation and Its Effects on Components and Systems (RADECS), Toulouse, France, 2023, pp. 1-4, doi: 10.1109/RADECS59069.2023.10767017
[4] P. Maillard et al., "Single-Event Evaluation of Xilinx 16nm UltraScale+™ Single Event Mitigation IP," in 2018 IEEE Radiation Effects Data Workshop (REDW), Jul. 2018, pp. 1–5. doi: 10.1109/NSREC.2018.8584298.
[5] Microsemi Handbook: CoreEDAC v2.11, Microsemi Corporation, Jul. 2020
[6] Á. B. de Oliveira et al., "Analyzing the Influence of using Reconfiguration Memory Scrubber and Hardware Redundancy in a Radiation Hardened FPGA under Heavy Ions," in 2018 18th European Conference on Radiation and Its Effects on Components and Systems (RADECS), Sep. 2018, pp. 1–4. doi: 10.1109/RADECS45761.2018.9328683.
[7] Kuang-Hua Huang and J. A. Abraham, "Algorithm-Based Fault Tolerance for Matrix Operations," in *IEEE Transactions on Computers*, vol. C-33, no. 6, pp. 518-528, June 1984, doi: 10.1109/TC.1984.1676475
[8] J. Lant, J. Navaridas, M. Luján, and J. Goodacre, "Toward FPGA-Based HPC: Advancing Interconnect Technologies," IEEE Micro, vol. 40, no. 1, pp. 25–34, Jan. 2020, doi: 10.1109/MM.2019.2950655.
[9] M. Meyer, T. Kenter, and C. Plessl, "Multi-FPGA Designs and Scaling of HPC Challenge Benchmarks via MPI and Circuit-switched Inter-FPGA Networks," ACM Trans. Reconfigurable Technol. Syst., vol. 16, no. 2, p. 24:1-24:27, Mar. 2023, doi: 10.1145/3576200.
[10] J. J. Davis and P. Y. K. Cheung, "Achieving low-overhead fault tolerance for parallel accelerators with dynamic partial reconfiguration," in 2014 24th International Conference on Field Programmable Logic and Applications (FPL), Sep. 2014, pp. 1–6. doi: 10.1109/FPL.2014.6927447.
[11] S. Azimi et al., "Exploring the Impact of Soft Errors on the Reliability of Real-Time Embedded Operating Systems," Electronics, vol. 12, no. 1, Art. no. 1, Jan. 2023, doi: 10.3390/electronics12010169.

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2026.3709831

[12] M. Straka, J. Kastil, and Z. Kotasek, "Generic partial dynamic reconfiguration controller for fault tolerant designs based on FPGA," in NORCHIP 2010, Nov. 2010, pp. 1–4. doi: 10.1109/NORCHIP.2010.5669477.
[13] R. Panek, J. Lojda, J. Podivinsky, and Z. Kotasek, "Reliability Analysis of Reconfiguration Controller for FPGA–Based Fault Tolerant Systems: Case Study," in 2020 International Symposium on VLSI Design, Automation and Test (VLSI-DAT), Aug. 2020, pp. 1–4. doi: 10.1109/VLSI-DAT49148.2020.9196269.
[14] R. Pánek and J. Lojda, "The Fault-tolerant Single-FPGA Systems with a Self-repair Reconfiguration Controller," in 2023 IEEE 14th Latin America Symposium on Circuits and Systems (LASCAS), Feb. 2023, pp. 1–4. doi: 10.1109/LASCAS56464.2023.10108372.
[15] C. Bolchini, A. Miele, and C. Sandionigi, "Increasing autonomous fault-tolerant FPGA-based systems' lifetime," in 2012 17th IEEE European Test Symposium (ETS), May 2012, pp. 1–6. doi: 10.1109/ETS.2012.6233006.
[16] A. Portaluri, S. Azimi, C. De Sio, L. Sterpone, and D. M. Codinachs, "Radiation-induced Effects on DMA Data Transfer in Reconfigurable Devices," in 2022 IEEE 28th International Symposium on On-Line Testing and Robust System Design (IOLTS), Sep. 2022, pp. 1–7. doi: 10.1109/IOLTS56730.2022.9897262.
[17] C. De Sio, S. Azimi, and L. Sterpone, "On the Evaluation of SEU Effects on AXI Interconnect Within AP-SoCs," in Architecture of Computing Systems – ARCS 2020, A. Brinkmann, W. Karl, S. Lankes, S. Tomforde, T. Pionteck, and C. Trinitis, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 215–227. doi: 10.1007/978-3-030-52794-5_16.
[18] C. Bolchini, L. Fossati, D. M. Codinachs, A. Miele, and C. Sandionigi, "A Reliable Reconfiguration Controller for Fault-Tolerant Embedded Systems on Multi-FPGA Platforms," in 2010 IEEE 25th International Symposium on Defect and Fault Tolerance in VLSI Systems, Oct. 2010, pp. 191–199. doi: 10.1109/DFT.2010.30.
[19] C. Bolchini and C. Sandionigi, "Design of Hardened Embedded Systems on Multi-FPGA Platforms," ACM Trans. Des. Autom. Electron. Syst., vol. 20, no. 1, p. 16:1-16:26, Nov. 2014, doi: 10.1145/2676551.
[20] L. Sterpone, M. Porrmann and J. Hagemeyer, "A Novel Fault Tolerant and Runtime Reconfigurable Platform for Satellite Payload Processing," in IEEE Transactions on Computers, vol. 62, no. 8, pp. 1508-1525, Aug. 2013, doi: 10.1109/TC.2013.80
[21] R. Giordano et al., "Frame-Level Intermodular Configuration Scrubbing of On-Detector FPGAs for the ARICH at Belle II," IEEE Transactions on Nuclear Science, vol. 68, no. 12, pp. 2810–2817, Dec. 2021, doi: 10.1109/TNS.2021.3127446.
[22] A. Ebrahim, K. Benkrid, X. Iturbe, and C. Hong, "A novel high-performance fault-tolerant ICAP controller," in 2012 NASA/ESA Conference on Adaptive Hardware and Systems (AHS), Jun. 2012, pp. 259–263. doi: 10.1109/AHS.2012.6268660.
[23] I. V. McLoughlin and T. Bretschneider, "Achieving low-cost high-reliability computation through redundant parallel processing," in 2006 International Conference on Computing & Informatics, Jun. 2006, pp. 1–6. doi: 10.1109/ICOCI.2006.5276450.
[24] AMD, Soft Error Mitigation Controller v4.1, (PG036), 2018
[25] Sophie Duzellier, Radiation effects on electronic devices in space, Aerospace Science and Technology, Volume 9, Issue 1, 2005, Pages 93-99, https://doi.org/10.1016/j.ast.2004.08.006.
[26] T. Li, H. Yang, H. Zhao, N. Wang, Y. Wei and Y. Jia, "Investigation into SEU Effects and Hardening Strategies in SRAM Based FPGA," 2017 17th European Conference on Radiation and Its Effects on Components and Systems (RADECS), Geneva, Switzerland, 2017, pp. 1-5, doi: 10.1109/RADECS.2017.8696177
[27] C. De Sio, S. Azimi, L. Sterpone, D. M. Codinachs and F. Decuzzi, "PyXEL: Exploring Bitstream Analysis to Assess and Enhance the Robustness of Designs on FPGAs," 2023 19th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), Funchal, Portugal, 2023, pp. 1-4, doi: 10.1109/SMACD58065.2023.10192116
[28] R. Leveugle, A. Calvez, P. Maistri and P. Vanhauwaert, "Statistical fault injection: Quantified error and confidence," 2009 Design, Automation & Test in Europe Conference & Exhibition, Nice, France, 2009, pp. 502-506, doi: 10.1109/DATE.2009.5090716
[29] L. Bozzoli et al., "EuFRATE: European FPGA Radiation-hardened Architecture for Telecommunications," 2023 Design, Automation & Test in Europe Conference & Exhibition (DATE), Antwerp, Belgium, 2023, pp. 1-6, doi: 10.23919/DATE56975.2023.10137035.



Giorgio Cora, (Student Member, IEEE) received his B.S. and M.S. degrees in Electronic Engineering from Politecnico di Torino, Turin, Italy, in 2021 and 2023 respectively. He is currently a Ph.D. Student in the CAD and Reliability Group, Departement of Computer and Control Engineering at Politecnico di Torino. His research focuses on the reliability analysis of electronic components, reliable RISC-V based systems and reconfigurable designs.



Daniele Rizzieri received the B.S. degree in Computer Engineering and the M.S. degree in Mechatronic Engineering from Politecnico di Torino, Turin, Italy, in 2019 and 2021 respectively. He's been a Research Assistant at Politecnico di Torino in the Department of Computer and Control Engineering (DAUIN) from 2021 to 2023 and is currently working in Argotec as Lead of Avionics System Engineering. His research activities included radiation tests, FPGA design, and mitigation approaches on reconfigurable devices in the context of several space-related projects such as HERA, carried on by European Space Agency



Corrado De Sio, (Member, IEEE) earned M.S. degrees in Computer Engineering from the University of Pisa. He received his Ph.D. in Computer and Control Engineering from Politecnico di Torino in 2023. In 2022, he served as a visiting scientist at ESTEC/ESA, where he focused on the reliability of soft error estimation in reprogrammable devices. Currently, he is an Assistant Professor at Politecnico di Torino and a member of the CAD and Reliability Group. His research interests include the reliability and fault tolerance of electronic systems, reconfigurable devices, and radiation effects.



Sarah Azimi, (Member, IEEE) received the Ph.D. degree from Politecnico di Torino, Turin, Italy, in 2019. She is an Assistant Professor with the CAD and Reliability Group, Department of Computer and Control Engineering, Politecnico di Torino. Her research interests include fault-tolerant electronic design, intelligent informative systems, physical models, and validation platforms.



Luca Sterpone (Senior Member, IEEE) received the M.S. and Ph.D. degrees in computer engineering from the Politecnico di Torino, Turin, Italy, in 2003 and 2007, respectively, He is currently a Full Professor with the Department of Computer and Control Engineering, Politecnico di Torino. He has authored more than 220 papers. His current research interests include re-configurable computing, computer-aided design algorithms, fault tolerance architectures, and radiation effects on components and systems. Prof. Sterpone received several awards for his research activities.

© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.