Resilience Analysis of a Fault-Tolerant MPSoC Interconnection Architecture under SEU Fault Injection

Thiago H Rausch, Wesley Grignani, Luigi Dilillo, Douglas R Melo

Title: Resilience Analysis of a Fault-Tolerant MPSoC Interconnection Architecture under SEU Fault Injection

Author(s): Thiago H. Rausch, Wesley Grignani, Luigi Dilillo, and Douglas R. Melo.

Document version: Pre-print version (Final draft)

Please cite the original version:
Thiago H. Rausch, Wesley Grignani, Luigi Dilillo, and Douglas R. Melo, ”Resilience Analysis of a Fault-Tolerant MPSoC Interconnection Architecture under SEU Fault Injection”, in 27th IEEE Latin American Test Symposium - LATS, 2026.

Resilience Analysis of a Fault-Tolerant MPSoC Interconnection Architecture under SEU Fault Injection

Thiago H. Rausch*, Wesley Grignani†, Luigi Dilillo†, and Douglas R. Melo*
*LEDS, University of Vale do Itajaí, Brazil
†IES, University of Montpellier, CNRS, Montpellier, France
thiagorausch@edu.univali.br, {wesley.grignani, luigi.dilillo}@umontpellier.fr, drm@univali.br

Abstract

The reliability of on-chip communication is a key enabler for dependable Multiprocessor System-on-Chip (MPSoC) platforms operating in radiation-harsh environments. This work investigates the fault-tolerance of the eXtensible Interconnect Network Architecture (XINA), a configurable Network-on-Chip (NoC) integrated with an AMBA AXI-compatible Network Interface (NI), both of which employ Triple Modular Redundancy (TMR) in the control logic and Hamming Error-Correcting Codes (ECC) in the data buffers. The evaluation focuses on the interconnection fabric of a 10×10 MPSoC prototype composed of 100 routers and 100 AXI-connected cores, where manager and subordinate IPs act as traffic generators and monitors data integrity under full load. A large-scale fault-injection campaign comprising 1,000 simulations with Single-Event Upset (SEU) injections was conducted to assess the impact of fault-tolerant mechanisms across four hardening configurations. The results demonstrate improved reliability, with the fully protected architecture achieving an 87.2% reduction in observed errors relative to the baseline while maintaining stable throughput and data integrity across all tests. These findings confirm the effectiveness of combining NoC and NI hardening for resilient interconnect design, reinforcing XINA as a dependable and scalable solution for high-reliability MPSoCs in mission-critical applications.

Index Terms

Multiprocessor System-on-Chip, Network-on-Chip, AMBA AXI, Reliability.

I. Introduction

System-on-Chip (SoC) platforms have evolved from single-core devices to Multiprocessor Systems-on-Chip (MPSoCs), which integrate multiple heterogeneous processing elements, memory subsystems, and I/O peripherals on a single die [1]. This integration enables parallel execution and functional specialization under power and area constraints for embedded electronics ranging from mobile to industrial control [2]. As integration scales, the on-chip interconnection fabric becomes a central determinant of system performance, predictability, and energy efficiency.

Conventional shared buses face structural limitations in highly parallel MPSoCs. Bandwidth contention, increased arbitration latency, and timing closure challenges under heavy load restrict scalability beyond modest core counts. Network-on-Chip (NoC) interconnects address these issues by replacing global buses with packet-switched router networks, providing modular composition, path diversity, and scalable bandwidth for many-core integration [3]. In contemporary design flows, NoCs must also interoperate with widely used bus protocols, notably AMBA AXI [4], via Network Interfaces (NIs) that translate transactions into packets and back while preserving ordering and flow-control semantics [2].

Reliability is a primary requirement for MPSoCs used in safety-critical applications, such as autonomous vehicles and aerospace systems. Radiation effects, such as Single-Event Upsets (SEUs) and permanent faults, can perturb router state, buffers, and control paths, disrupting end-to-end communication. Fault tolerance techniques commonly applied to the interconnect include Error-Correcting Codes (ECC) for data integrity, modular redundancy for control logic, and routing strategies that limit or avoid faulty resources [5]. Evaluating these mechanisms at the system level is necessary because local protections interact with global traffic patterns, endpoint behavior, and timing.

This work evaluates the eXtensible Interconnect Network Architecture (XINA), a configurable NoC designed for fault-prone environments [6], together with an AXI-compatible NI [7]. Both the NoC routers and the NI incorporate optional hardening mechanisms, including Triple Modular Redundancy (TMR) in the control logic and Hamming ECC in the input buffers. The assessment targets a 10×10 MPSoC instance that integrates routers and NIs with AXI traffic generators and monitors. A simulation-based fault-injection methodology emulates SEUs as single-bit upsets in memory elements during timed simulation, enabling quantitative analysis of error manifestation and its dependence on protection scope.

This work was supported in part by the Foundation for Support of Research and Innovation, Santa Catarina – FAPESC, Call 51/2024 (Grants 2023TR000880, 2024TR001897 and 2025TR001565), the Brazilian National Coordination of Superior Level Staff Improvement (CAPES/PROSUC), the Brazilian National Council for Scientific and Technological Development – CNPq (Processes 408641/2023-1 and 350794/2023-5), the EU Horizon Europe Twinning project TWIN-RELECT (Grant 101160314), and the École Doctorale I2S from the University of Montpellier.

The remainder of this paper is organized as follows. Section II presents the XINA NoC, the AMBA AXI-compatible NI, and the fault injection methodology. Section III summarizes the state of the art on fault-tolerant MPSoCs and NoCs. Section IV details the experimental setup, scope of fault-tolerance, MPSoC architecture, testbench, and injection workflow. Section V reports synthesis and performance metrics, simulation runtime, and the SEU fault-injection campaign. Finally, Section VI discusses the main findings and outlines future work.

II. Background

A. XINA NoC

The eXtensible Interconnect Network Architecture (XINA) [6] is a configurable Network-on-Chip (NoC) designed for fault-prone environments such as space applications, where radiation-induced faults are frequent. It employs a five-port router configured in a 2D mesh topology, featuring one local port and four directional ports interconnected by a central crossbar. The baseline configuration uses handshake-based flow control, deterministic XY routing, round-robin arbitration, input FIFO buffering, and wormhole switching, ensuring predictable communication and moderate latency even under high load.

XINA's main distinction lies in its configurability for fault-tolerance mechanisms. Routing, flow control, and arbitration controllers can be synthesized as Moore or Mealy state machines, allowing trade-offs among area, speed, and timing predictability. Buffers support both shift-register and ring-buffer implementations, while critical control modules may include TMR and input buffers that employ Hamming ECC. Prior studies [6], [8], [9] have demonstrated the effectiveness of these design choices, confirming XINA as a robust and flexible interconnect architecture.

The XINA Network Interface (NI) [10] bridges AMBA AXI-5 IP cores and the XINA NoC by translating memory-mapped AXI transactions into flits for packet-switched communication. Its architecture includes a configurable front-end for AXI protocol adaptation and a parameterizable back-end equipped with packetization and depacketization units, Hamming encoding and decoding, FIFO buffers, and flow and integrity control modules. Critical back-end controllers are protected by TMR, while data buffers use Hamming ECC to ensure fault tolerance and data integrity. An address translation module maps core identifiers to mesh coordinates, enabling seamless integration of standard AXI IPs into the network. Both Manager and Subordinate NI variants have been validated through simulation-based fault-injection experiments [7], demonstrating their effectiveness in maintaining reliable communication in fault-tolerant MPSoC interconnects.

B. Fault injection

Fault injection is a systematic methodology for evaluating the dependability by intentionally introducing faults and observing their effects. In the context of MPSoCs, it enables the controlled emulation of radiation-induced disturbances, such as SEUs, to analyze how faults propagate through the architecture or are masked. Existing approaches include hardware, software, simulation, emulation, and hybrid-based methods, each offering a distinct balance of controllability, observability, and execution time. Simulation-based fault injection (SFI) offers temporal control and detailed observability at the RTL level, enabling the extraction of dependability metrics, such as error probability, fault coverage, and recovery latency. Although physical methods replicate real fault conditions with limited control, simulation-based techniques offer greater precision and repeatability when supported by a fault model that specifies the type, location, and duration of injected faults [11], [12].

III. Related Work

Research on fault-tolerant MPSoC and NoC architectures spans multiple domains, addressing reliability at different abstraction levels and targeting both transient and permanent faults. Fuchs et al. [13] presented a CubeSat-oriented MPSoC using hybrid redundancy and memory scrubbing to ensure dependable operation under radiation. Rashid et al. [14] proposed a router-centric fault tolerance scheme with redundant pipeline stages and bypass mechanisms to improve resilience against permanent faults. Dang et al. [15] developed the 3D-FETO architecture, which combines ECC-based soft-error protection and adaptive routing to tolerate diverse fault types in 3D NoCs. NASA's HPSC [16] introduced a radiation-hardened, cache-coherent MPSoC employing a packet-switched NoC with multi-layer protection and lockstep cores for space-grade computing. Derin et al. [17] proposed the MADNESS framework, which enhances adaptability in MPSoCs through task migration and dynamic remapping to recover from permanent faults.

Other studies have focused on validation methodologies and lightweight protection schemes. Yaghini et al. [18] investigated synchronous and asynchronous NoC routers under SEU and SET fault injection and found reduced fault propagation in handshake-based designs. Coutinho and Berejuck [19] evaluated Hamming and TMR protection in nanosatellite-oriented NoCs, demonstrating that ECC provides fault coverage at a lower hardware cost. El Salloum et al. [20] introduced the ACROSS MPSoC, which integrates a Time-Triggered NoC for determinism and strong fault containment in mixed-criticality systems. Wächter et al. [21] proposed a hierarchical MPSoC with router- and processing-element-level fault detection and recovery via a dedicated fault-notification network, enabling sustained operation under multiple concurrent faults.

Table I summarizes these studies, highlighting their respective targets, platforms, and focus. The comparison shows that most works address fault-tolerance at isolated levels, typically within routers or validation frameworks, rather than evaluating the interconnect as an integrated subsystem. In contrast, this work jointly analyzes the NoC and its AMBA AXI-compatible network interfaces within a unified MPSoC interconnection, emphasizing communication reliability through a consistent simulation-based fault-injection methodology.

TABLE I
Summary of related work on FT NoC and MPSoC designs

Work	Scope	Platform	Focus
[13]	MPSoC	FPGA	Fault-tolerant MPSoC
[14]	NoC router	ASIC	Fault-tolerant NoC router
[15]	NoC router	ASIC	Fault-tolerant NoC router
[16]	MPSoC	ASIC	Space-grade Fault-tolerant MPSoC
[17]	MPSoC	FPGA	Fault-tolerant MPSoC framework
[18]	NoC router	—	SEU fault injection on FT router
[19]	SoC	FPGA	SEU fault injection on FT SoC
[20]	MPSoC	FPGA	Fault injection on FT MPSoC
[21]	MPSoC	FPGA	Hierarchical FT MPSoC framework
This work	MPSoC	FPGA	SEU fault injection on FT MPSoC

IV. Implementation

This section describes the scope and configuration for evaluating fault-tolerance in an MPSoC. We detail the hardware and software setup, explain the simulation-based fault-injection method, and describe how performance is measured.

A. Materials and Methods

Synthesis and implementation were performed with Vivado 2024.2 using VHDL 2008 sources. Functional fault-injection simulations were executed with ModelSim-Intel FPGA Starter Edition 20.1, driven by TCL scripts for batch compilation, execution, and log collection. All experiments were conducted on Ubuntu 24.04 LTS using an Intel Core i7-13700 CPU and 16 GB of RAM, targeting the AMD Zynq ZCU104 UltraScale+ (XCZU7EV-2FFVC1156) FPGA. Post-processing was carried out using Python and Bash scripts to parse ModelSim transcripts and trace logs, align faulty runs with the golden reference, classify divergences, and generate the summarized datasets presented in the results tables.

B. Fault Tolerance Scope

The evaluated fault tolerance mechanisms are confined to the NoC routers and the AMBA AXI-compatible network interfaces. SEUs are injected exclusively into sequential elements within these components, including control-state registers, finite-state machines, pointers, and FIFO storage elements. Traffic generators (AXI managers) and traffic monitors (AXI subordinates) connected to the interfaces remain unprotected and are excluded from the fault-injection scope, serving solely as controllable stimulus and observation sources and sinks. As a result, the reliability analysis isolates the contributions of NoC and NI hardening to the interconnection fabric, and the reported improvements reflect resilience gains that are independent of endpoint behavior.

Four NoC/NI protection configurations are examined to quantify the effectiveness of optional hardening features: the baseline (STD/STD), NoC-only hardened (FT/STD), NI-only hardened (STD/FT), and fully hardened (FT/FT). For each configuration, the analysis includes reliability under SEU injection, resource utilization, maximum operating frequency, power consumption, and end-to-end throughput and energy efficiency under identical traffic conditions. This organization enables correlating the hardening scope with implementation cost and observed fault resilience, while also localizing residual vulnerabilities across datapath buffers, pointer registers, and control paths within the interconnect.

C. MPSoC Architecture

Fig. 1 illustrates the 10×10 two-dimensional mesh used in our fault-tolerant NoC-based MPSoC, which corresponds to the largest MPSoC configuration that could be fully implemented on the target FPGA device while maintaining feasible synthesis and simulation runtimes. Each grid node integrates a router (in blue) and a Network Interface (NI) connected to either a Manager or a Subordinate IP block. Manager NIs are highlighted in red, while Subordinate NIs appear in green.

Each NI handles separate AXI read and write channels, mapping red tagged manager transactions and green tagged subordinate transactions into blue NoC flits. Packets are then routed through bidirectional flow-control links to north, south, east, and west neighbors.

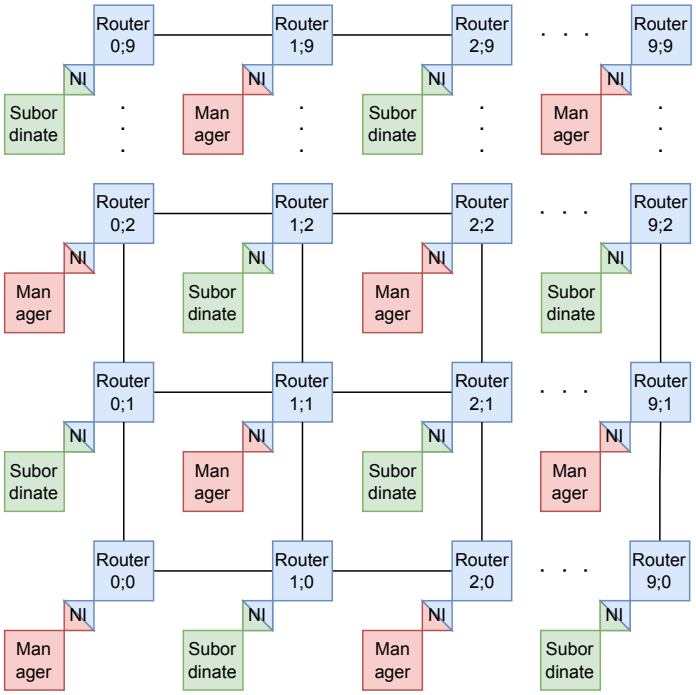

Fig. 1. Complete 10×10 mesh topology for the fault-tolerant NoC-MPSoC. Routers (blue) form a regular grid; Manager NIs (red) and Subordinate NIs (green) attach at each node.

D. Testbench Setup

Fig. 2 exemplifies the dedicated MPSoC testbench that instantiates the complete 10×10 mesh NoC, integrating 50 manager traffic generators, 50 subordinate traffic monitors, their associated network interfaces, and all routers. Each mesh node contains a router and one network interface, connected either to a manager or to a subordinate endpoint, ensuring uniform hop distances and balanced communication patterns.

Managers inject traffic into the network, while subordinates act as sinks, recording the transactions they receive. Endpoints are placed in a checkerboard pattern, where $(x + y) \bmod 2 = 1$ denotes managers and $(x + y) \bmod 2 = 0$ denotes subordinates. Each manager communicates with a unique subordinate given by $(x', y') = \big((x+5) \bmod 10, (y+4) \bmod 10\big)$, ensuring one-to-one communication evenly distributed across the mesh.

Each manager NI translates AXI-5 transactions into 32-bit flits, while subordinate NIs perform depacketization and store transactions in local logs. Routers and NIs use four-word FIFOs for buffering, and all control logic is implemented as Moore FSMs to simplify timing analysis and hardening. A global clock and reset synchronize all modules. During the baseline (fault-free) run, every AXI request/response is logged in per-node traces, creating a golden reference for subsequent comparison in fault-injection campaigns.

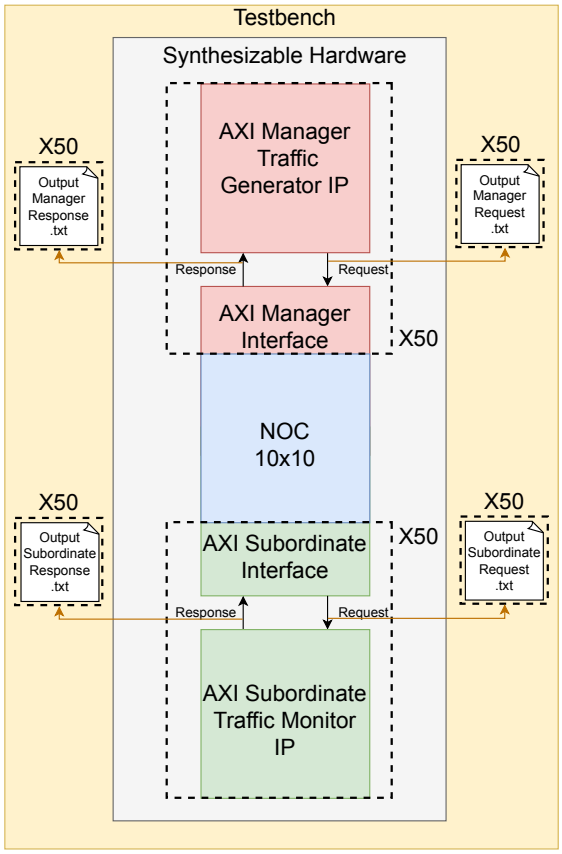

Fig. 2. Testbench setup: 50 manager IPs (red) and 50 subordinate IPs (green) connected via dedicated network interfaces to the 10×10 NoC (blue). All transactions are recorded for golden reference and fault-injection analysis.

E. Fault Injection Setup

In this study, the adopted fault model is an SEU obtained by flipping individual bits in sequential elements during simulation. This simulation-based methodology, introduced in [6] and used in related work [7], [22], enables a systematic and low-cost pre-irradiation evaluation of circuit resilience. A fault-free *golden run* is first executed to generate reference logs. Then, `totalRuns` (e.g., 1000) fault-injection runs are performed, each injecting a single-bit upset at a random simulation timestamp into a uniformly random eligible register within the evaluated scope (NoC routers and/or NIs, depending on the configuration). This normalizes the injection distribution to each configuration's eligible register set, enabling fair cross-configuration comparison. The simulation then resumes to completion, and traces are stored for later comparison.

Automation is managed via TCL scripting within the ModelSim shell, which handles compilation, execution, and batch iteration for all runs. Post-processing is performed using Python and Bash scripts to align faulty traces against the golden reference, detect deviations such as altered data, missing or extra transactions, and timing mismatches, and classify errors by type and location. This workflow ensures reproducibility and quantitative assessment of the NoC and NI resilience under SEU injection, as summarized in Fig. 3.

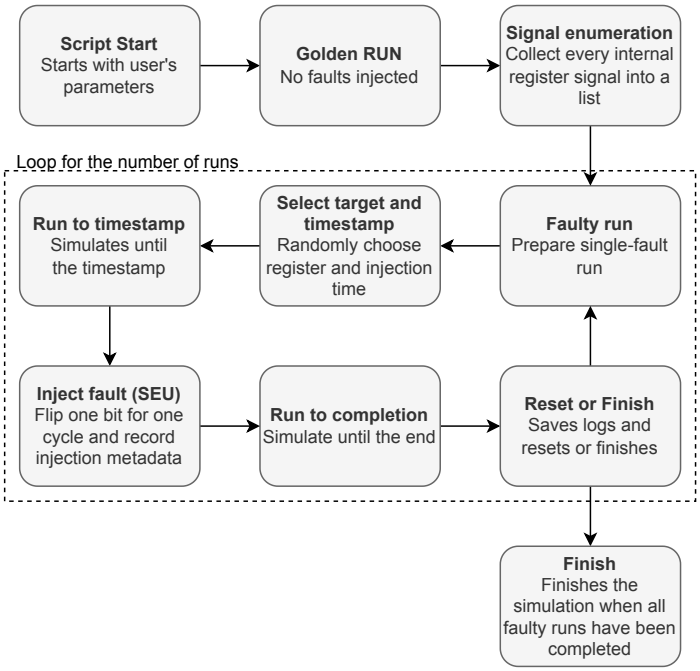

Fig. 3. Fault injection methodology, including the golden reference run, randomized fault injection, and log comparison against the baseline.

V. Results

A. Synthesis and Performance Results

Tables II and III present the synthesis and performance results for the four NoC/NI protection configurations: standard (STD/STD), NoC-only (FT/STD), NI-only (STD/FT), and fully protected (FT/FT). The baseline achieves the highest operating frequency and throughput while consuming the least power and occupying the smallest area. Introducing protection mechanisms increases resource utilization and reduces frequency and throughput, reflecting the cost of redundancy. The NoC-only configuration exhibits the greatest frequency degradation, whereas the NI-only variant incurs moderate overhead with intermediate performance. Complete protection yields the highest resource and power demands, along with the lowest throughput.

TABLE II
Performance results for different NoC/NI configurations

NoC	NI	Fmax (MHz)	Power (mW)	Throughput (Gbps)[1]	Energy (mJ)[2]
STD	STD	239.75	903	2.119	1.488
FT	STD	158.98	1,031	1.405	2.563
STD	FT	194.02	973	1.714	1.981
FT	FT	167.36	1,115	1.479	2.633

[1] Throughput of 109,132 transmitted flits of 32 bits each. [2] Estimated energy consumption to process the communication load over 395,085 cycles.

B. Simulation Runtime

The simulation campaign incurs a nontrivial computational cost that varies with the scope of protection. Table IV reports the wall-clock runtime per NoC/NI configuration under the same workload, simulator settings, and host machine described in the implementation section. Each entry corresponds to one whole campaign, comprising the fault-free golden run and 1,000 runs with SEU injection.

TABLE IV
Simulation runtime for each NoC/NI configuration

Configuration	Runtime
STD/STD	6 days, 4h, 40 min
STD/FT	9 days, 23h, 56 min
FT/STD	19 days, 9h, 20 min
FT/FT	25 days, 14h, 43 min

TABLE III
Resource utilization across NoC/NI protection configurations

STD/STD				FT/STD			
Block	**LUTs**	**LUTRAMs**	**FFs**	**Block**	**LUTs**	**LUTRAMs**	**FFs**
NIs	25,744 (11.2%)	3,400 (3.3%)	22,344 (4.8%)	NIs	29,273 (12.7%)	0 (0.0%)	18,350 (3.9%)
NoC	67,323 (29.2%)	0 (0.0%)	66,919 (14.3%)	NoC	106,374 (46.2%)	0 (0.0%)	80,946 (17.3%)
Endpoints	7,150 (3.1%)	800 (0.8%)	8,100 (1.7%)	Endpoints	7,150 (3.1%)	800 (0.8%)	8,100 (1.7%)
Total	100,344 (43.6%)	4,200 (4.1%)	96,369 (20.6%)	**Total**	143,940 (62.5%)	800 (0.8%)	107,396 (22.9%)
STD/FT				**FT/FT**			
Block	**LUTs**	**LUTRAMs**	**FFs**	**Block**	**LUTs**	**LUTRAMs**	**FFs**
NIs	45,773 (19.9%)	4,600 (4.5%)	32,912 (7.0%)	NIs	45,784 (19.9%)	4,600 (4.5%)	20,408 (4.4%)
NoC	67,160 (29.1%)	0 (0.0%)	69,919 (14.9%)	NoC	103,992 (45.1%)	0 (0.0%)	80,948 (17.3%)
Endpoints	7,150 (3.1%)	800 (0.8%)	8,100 (1.7%)	Endpoints	7,150 (3.1%)	800 (0.8%)	8,100 (1.7%)
Total	120,203 (52.2%)	5,400 (5.3%)	98,419 (21.0%)	**Total**	157,014 (68.1%)	5,400 (5.3%)	109,448 (23.4%)

Percentages indicate relative utilization of FPGA resources with respect to the target device (AMD Xilinx Zynq UltraScale+ ZCU104).

C. SEU Injection Campaign

Table V presents the results of the fault-injection campaign performed on the MPSoC interconnect. The injected faults were classified according to the affected structure: *FIFO datapath* (buffers and checksum-protected storage), *Pointer registers* (indexing logic within FIFOs), and *Control logic* (FSMs, counters, and status registers). This categorization enables a detailed assessment of fault manifestation across protection configurations and provides a consistent basis for evaluating the contribution of each hardening mechanism.

The baseline (STD/STD) configuration, without protection in either the NoC or the NIs, resulted in 109 observable errors across all structural categories. Applying protection only to the NoC (FT/STD) reduced the error count to 29, removing faults in control logic and confining most residual errors to pointer registers. In contrast, protection limited to the NIs (STD/FT) reduced errors to 75, effectively mitigating failures in interface datapaths but leaving the unprotected NoC as the primary source of system-level faults. The fully hardened configuration (FT/FT) achieved the fewest observable errors, with only 14 remaining cases, predominantly associated with pointer registers and minor control elements, whereas datapath-related faults were entirely masked by the introduced hardening features.

Overall, the results confirm that NoC-level hardening has the greatest impact on system resilience, whereas NI-level hardening provides intermediate benefits at a lower implementation cost. When combined, the two mechanisms complement each other, substantially reducing fault propagation and confining residual vulnerability to compact control structures that can be hardened in future design iterations. It is worth noting that, as hardening increases the amount of sequential logic (e.g., due to redundancy), the absolute SEU-sensitive state may also grow; thus, while we apply the same one-fault-per-run injection policy to all configurations, absolute in-field error rates would additionally depend on the architecture's effective sensitive area and the radiation environment. This evaluation further indicates a shift

TABLE V
Signals, faults, and errors across configurations

STD/STD				FT/STD			
Category	**Signals**	**Faults**	**Errors**	**Category**	**Signals**	**Faults**	**Errors**
FIFO datapath	87,120 (85.5%)	845 (84.5%)	40 (36.7%)	*FIFO datapath*	98,160 (79.8%)	786 (78.6%)	7 (24.1%)
Pointer registers	3,220 (3.2%)	37 (3.7%)	37 (33.9%)	*Pointer registers*	3,220 (2.6%)	20 (2.0%)	19 (65.5%)
Control logic	11,610 (11.4%)	118 (11.8%)	32 (29.4%)	*Control logic*	21,730 (17.7%)	194 (19.4%)	3 (10.4%)
Total	101,950 (100%)	1,000 (100%)	109 (100%)	**Total**	123,110 (100%)	1,000 (100%)	29 (100%)
STD/FT				**FT/FT**			
Category	**Signals**	**Faults**	**Errors**	**Category**	**Signals**	**Faults**	**Errors**
FIFO datapath	91,920 (71.4%)	716 (71.6%)	25 (33.3%)	*FIFO datapath*	102,960 (68.6%)	487 (81.0%)	0 (0.0%)
Pointer registers	3,220 (2.5%)	29 (2.9%)	26 (34.7%)	*Pointer registers*	3,220 (2.1%)	13 (2.2%)	11 (78.6%)
Control logic	33,710 (26.2%)	255 (25.5%)	24 (32.0%)	*Control logic*	43,830 (29.2%)	101 (16.8%)	2 (14.3%)
Total	128,850 (100%)	1,000 (100%)	75 (100%)	**Total**	150,010 (100%)	1,000 (100%)	14 (100%)

in the dominant error sources as the protection scope changes. While NI hardening reduces error manifestations at the interface datapath, NoC hardening is necessary to avoid network-wide fault propagation. Consequently, NI-only hardening can be a cost-effective first step when endpoint transaction integrity is the primary concern, but it is insufficient for interconnect-wide resilience if routers remain unprotected.

VI. Conclusion

The evaluation of the proposed fault-tolerant MPSoC interconnection reveals a trade-off among reliability, resource utilization, and performance. The adoption of TMR and ECC increases resilience but also raises implementation costs, particularly in the fully protected configuration (FT/FT), where LUT usage increases by approximately 56% and flip-flop usage grows by about 14% relative to the baseline. These overheads reduce the maximum operating frequency and increase power demand, which must be balanced against resilience requirements, particularly in FPGA-based systems with constrained resources.

Across configurations, the impact of protection mechanisms varies according to their scope and application. The baseline (STD/STD) configuration achieved the highest performance but also the largest number of observable errors (109). Introducing fault tolerance exclusively in the NoC (FT/STD) reduced the error count to 29 (73.4% reduction) by eliminating control-related faults, leaving pointer registers as the primary residual source. NI-only protection (STD/FT) achieved partial mitigation, reducing the error count to 75 (31.2% reduction) and eliminating datapath-related faults in the interfaces. The fully hardened configuration (FT/FT) achieved the highest reliability, with only 14 uncorrected errors (87.2% error reduction), leaving the remaining vulnerabilities confined to pointer registers and small control elements. These results confirm that NoC-level hardening provides the greatest improvement in fault tolerance, whereas NI-level protection yields intermediate gains with reduced overhead.

Future work will extend this analysis by conducting radiation testing of the complete MPSoC, including routers, network interfaces, and endpoint IPs. These experiments will allow correlation between simulation-based fault injection and measured error rates, providing experimental confirmation of the modeled resilience. In parallel, an AXI-Stream variant of the XINA network interface will be developed and evaluated using the same methodology, enabling a direct comparison of memory-mapped and streaming communication modes under fault-prone conditions.

References

[1] W. Wolf and A. Jerraya, *Multiprocessor Systems-on-Chips*. Morgan Kaufmann, 2005.

[2] D. Greaves, *Modern System-on-Chip Design on Arm*. ARM Education Media, 2021. [Online]. Available: https://www.arm.com/resources/ebook/modern-soc

[3] S. Kundu and S. Chattopadhyay, *Network-on-Chip: The Next Generation of System-on-Chip Integration*. CRC Press, Taylor & Francis Group, 2014.

[4] ARM. (2025) AMBA AXI Protocol Specification. Available at: https://developer.arm.com/documentation/ihi0022/latest.

[5] F. L. Kastensmidt and P. Rech, *FPGAs and Parallel Architectures for Aerospace Applications: Soft Errors and Fault-Tolerant Design*. Springer, 2016.

[6] D. R. Melo, C. A. Zeferino, L. DiLillo, and E. Bezerra, “Maximizing the inner resilience of a network-on-chip through router controllers design,” *Sensors*, 2019.

[7] T. H. Rausch, W. Grignani, G. H. S. Müller, D. A. Santos, L. Dilillo, and D. R. Melo, “Hardening an AMBA-AXI network interface for a reliable Network-on-Chip,” in *2025 IEEE 16th Latin America Symposium on Circuits and Systems (LASCAS)*, vol. 1, 2025, pp. 1–5.

[8] D. R. Melo, C. A. Zeferino, E. A. Bezerra, and L. Dilillo, “Design and evaluation of implementation impact on a fault-tolerant network-on-chip router,” in *2021 16th International Conference on Design & Technology of Integrated Systems in Nanoscale Era (DTIS)*. IEEE, 2021, pp. 1–6.

[9] G. S. Mafra, T. H. Rausch, D. A. Santos, L. Dilillo, E. A. Bezerra, and D. R. Melo, “Assessing the reliability of a network-on-chip through physical validation,” in *LASSS/LACW 2022-Joint 3rd IAA Latin American Symposium on Small Satellites and 5th IAA Latin American CubeSat Workshop*, 2022.

[10] G. H. S. Muller, T. H. Rausch, and D. R. Melo, “Interface de rede amba-axi para uma rede-em-chip confiável,” *Anais do Computer on the Beach*, vol. 15, pp. 172–178, 2024.

[11] H. Ziade, R. Ayoubi, and R. Velazco, “A survey on fault injection techniques,” *The International Arab Journal of Information Technology*, vol. 1, no. 2, pp. 171–186, 2004.

[12] I. Koren and C. M. Krishna, *Fault-Tolerant Systems*. San Francisco, CA, USA: Morgan Kaufmann, 2007.

[13] C. M. Fuchs, P. Chou, X. Wen, N. M. Murillo, G. Furano, S. Holst, A. Tavoularis, S.-K. Lu, A. Plaat, and K. Marinis, “A fault-tolerant MPSoC for CubeSats,” in *2019 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, 2019, pp. 1–6.

[14] M. Rashid, N. K. Baloch, M. A. Shafique, F. Hussain, S. Saleem, Y. B. Zikria, and H. Yu, “Fault□tolerant network□on□chip router architecture design for heterogeneous computing systems in the context of internet of things,” *Sensors*, vol. 20, no. 18, p. 5355, 2020.

[15] K. N. Dang, Y. Okuyama, and A. B. Abdallah, “Soft-error and hard-fault tolerant architecture and routing algorithm for reliable 3D-NoC systems,” *arXiv preprint arXiv:2003.09616*, 2020. [Online]. Available: https://arxiv.org/abs/2003.09616

[16] National Aeronautics and Space Administration, “High performance spaceflight computer,” NASA Science & Technology Mission Directorate, White Paper TMG-23Jul2024-1, 2024. [Online]. Available: https://www.nasa.gov/game-changing-development-projects/high-performance-spaceflight-computing-hpsc/

[17] O. Derin, E. Cannella, G. Tuveri, P. Meloni, T. Stefanov, L. Fiorin, L. Raffo, and M. Sami, “A system-level approach to adaptivity and fault-tolerance in noc-based mpsocs: The madness project,” *Microprocessors and Microsystems*, vol. 37, no. 6, pp. 515–529, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0141933113000975

[18] P. M. Yaghini, A. Eghbal, H. Pedram, and H. R. Zarandi, “Investigation of transient fault effects in synchronous and asynchronous Network on Chip router,” *Journal of Systems Architecture*, vol. 57, no. 1, pp. 61–68, Jan. 2011. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1383762110001293

[19] L. C. M. Coutinho and M. D. Berejuck, “Evaluation of a network-on-chip designed to deal with multiple processors in a nanosatellite,” *Revista Brasileira de Computação Aplicada*, vol. 12, no. 2, pp. 93–102, 2020. [Online]. Available: https://seer.upf.br/index.php/rbca/article/view/10120

[20] C. El Salloum, M. Elshuber, O. Höftberger, H. Isakovic, and A. Wasicek, “The ACROSS MPSoC – A new generation of multi-core processors designed for safety–critical embedded systems,” *Microprocessors and Microsystems*, vol. 37, no. 8, pp. 1020–1032, Nov. 2013. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0141933113001002

[21] E. W. WÄchter, V. Fochi, F. Barreto, A. M. Amory, and F. G. Moraes, “A hierarchical and distributed fault tolerant proposal for noc-based mpsocs,” *IEEE Transactions on Emerging Topics in Computing*, vol. 6, no. 4, pp. 524–537, 2018.

[22] W. Grignani, D. A. Santos, M. Kastriotou, C. Cazzaniga, L. Dilillo, and D. R. Melo, “Implementation and characterization of a fault-tolerant CCSDS 123 hardware accelerator under neutron radiation,” *Microprocessors and Microsystems*, p. 105184, 2025.