Design and Evaluation of Implementation Impact on a Fault-Tolerant Network-on-Chip Router

Douglas Rossi de Melo, Cesar Albenes Zeferino, Eduardo Augusto Bezerra, Luigi Dilillo

Title: Design and Evaluation of Implementation Impact on a Fault-Tolerant Network-on-Chip Router

Author(s): D. R. Melo, C. A. Zeferino, E. A. Bezerra and L. Dilillo

DOI: 10.1109/DTIS53253.2021.9505053

Published: 09 August 2021

Document version: Post-print version (Final draft)

Please cite the original version:
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,” 2021 16th International Conference on Design & Technology of Integrated Systems in Nanoscale Era (DTIS), 2021, pp. 1-6, doi: 10.1109/DTIS53253.2021.9505053.

Design and Evaluation of Implementation Impact on a Fault-Tolerant Network-on-Chip Router

Douglas Rossi Melo[*][†][‡], Cesar Albenes Zeferino[*], Eduardo Augusto Bezerra[†][‡], and Luigi Dilillo[‡][§]
[*]Laboratory of Embedded and Distributed Systems (LEDS), University of Vale do Itajaí, Brazil
[†]Space Systems Research Laboratory (SpaceLab), Federal University of Santa Catarina, Brazil
[‡]Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM), France
[§]Centre National de la Recherche Scientifique (CNRS), France
drm@univali.br, zeferino@univali.br, eduardo.bezerra@ufsc.br, dilillo@lirmm.fr

Abstract

This work investigates synthesis alternatives to minimize error propagation in the controllers responsible for flow regulation, packet routing, and resource arbitration in a Network-on-Chip router. The controllers are based on Finite-State Machines to provide flexibility and favor low resource usage in programmable logic devices. The proposed router embeds hardening techniques by using triple modular redundancy on controllers and the Hamming code on buffers. Experimental results show that the packet routing controller has the most impact on the metrics evaluated and that the migration from a Moore to a Mealy controller implementation reduces the error propagation and offers a higher throughput than hardening the controllers. The main contribution of this work consists of assessing the impact of different implementations of a router in terms of error propagation.

Index Terms

SoC, NoC, Fault Tolerance

I. Introduction

Reducing the component size and increasing the operating frequency of integrated circuits makes the Systems-on-Chip (SoCs) more susceptible to faults. Thus, depending on its operational environment, an SoC requires fault-tolerant components to minimize error propagation [1], including the communication infrastructure. As current multi- and many-core systems use Networks-on-Chip (NoCs) as interconnection architecture, the NoC must be able to detect a fault and prevent the resulting error from causing an application failure. However, providing reliability in an NoC affects silicon costs, communication performance, and power consumption, as it is usually done through redundancy.

Studies about fault tolerance in NoCs mainly address both transient and permanent faults. For instance, [2]–[7] examine Single Event Upset (SEU) in NoC components. The studies [2]–[4] address transient faults that are due to *crosstalk*. The authors of [8]–[10] investigate the problem of short and open circuit faults in the links of a Network-on-Chip. In works [11]–[13], the authors discuss the yield of vertical links in 3-D NoCs. The work [14] proposes a fault-tolerant buffer design. The studies [8], [15], and [16] examine intermittent faults and treat them as permanent faults. All of these works address the use of techniques to protect a given component and do not explore the hardening of different components using different techniques.

In this context, this work aims at evaluating the performance and resilience of an NoC router using combinations of flow regulation (or flow control), routing, and arbitration controllers, presenting the trade-off between the use of hardware resources and the susceptibility to error propagation. The proposed router implements hardening techniques through the application of the Triple Modular Redundancy (TMR) on controllers and the Hamming code on buffers. The first versions of router architecture were presented in [17] and [18], where we explored the combination of different router controllers.

In this work, we protect the router's buffers and combine it with the controllers protection for evaluation in terms of silicon cost, communication performance, and reliability. The main contribution of this work is the assessment of the impact of different implementations on the inner resilience of the router itself in terms of error propagation.

The remainder of this paper is structured in the following sessions. Session II describes the architecture of the proposed fault-tolerant router, while Session III presents the methods to evaluate the router. Following, Session IV presents and discusses the experimental results, and Session V gives the final remarks.

II. Router Architecture

A. The Baseline Router

This work presents a router architecture for 2-D mesh networks. The router has five ports named *Local*, *North*, *East*, *South*, and *West*. The *Local* port is the terminal at which a processing core is attached, and the other ports connect the router to its neighbors. Internally, it uses a distributed architecture with input and output channels interconnected by a crossbar (Fig. 1). The input channels comprise controllers responsible for input flow regulation and packet routing, while the output channels include controllers, which perform the channel arbitration and output flow regulation. Each input channel also integrates a parameterizable input buffer.

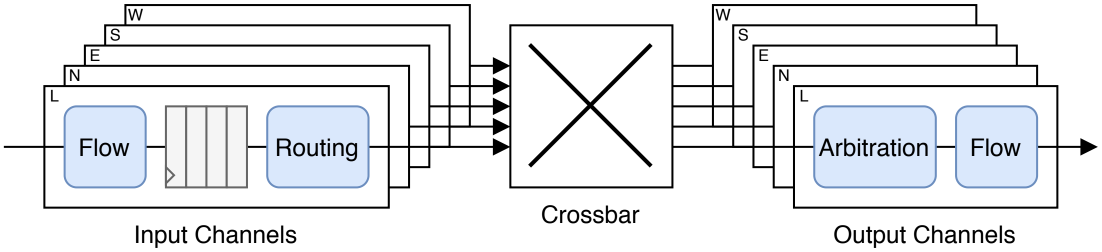


Fig. 1. Proposed router architecture.

The flow regulation controller implements a 4-stage handshake protocol for receiving and sending packet flits (a flit is the smallest piece of data over which is performed the flow regulation). The routing controller runs the XY algorithm to request an output channel to forward an incoming packet.
The arbitration controller consists of a Round-Robin arbiter that schedules the use of the output channel by the packets in the input channels. All these controllers are composed of Finite-State Machines (FSMs) and the router was designed to enable each controller to use a different type of FSM (Moore or Mealy). This approach enables us to better investigate architectural trade-off.

B. The Hardened Router

The proposed router architecture implements fault tolerance techniques. Since the memorization elements are the most susceptible to SEU faults, the internal controllers and the input buffers of the router were hardened.

The Triple Modular Redundancy (TMR) was the technique chosen to protect the FSM of each controller as it can mask an error transparently [1]. It consists of replicating the component in three units operating over the same input signals and using a major voter to compare each controller's output and elect the most common output value, as illustrated in Figure 2. This technique implies a high resource overhead if applied to complex structures. However, in the case of components with few interface signals and memory elements, as the controllers used in the proposed router, the TMR technique can provide a good cost/benefit.

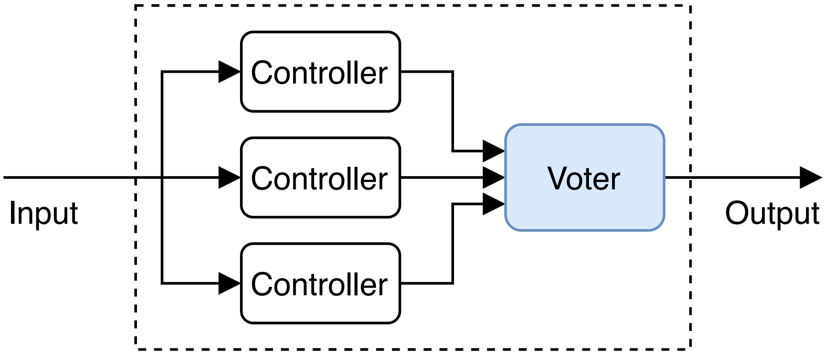


Fig. 2. Triple Modular Redundancy on controllers.

Hamming code [19] was used to protect the input buffers of the router. This technique relies on interleaving parity bits to detect up to two errors and correct a single error in a data word. Its implementation consists of an encoder, which generates the parity bits that compose the Error-Correcting Code (ECC) field, and a decoder that performs the error correction. We use Hamming solely to correct a single error. However, it is possible to forward a double error detection to an external structure, such as a network interface, for retransmission requests.

The encoder used for the Hamming code technique comprises a set of XOR gates, while the decoder checks parities through a Look-Up Table. In case of a mismatching output, the decoder fixes the wrong bit by inverting its value. As the ECC information must be attached to each packet flit, the number of memory elements of the input buffers increases. Figure 3 shows the application of the Hamming code.

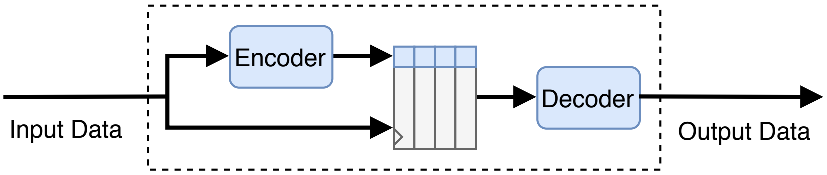


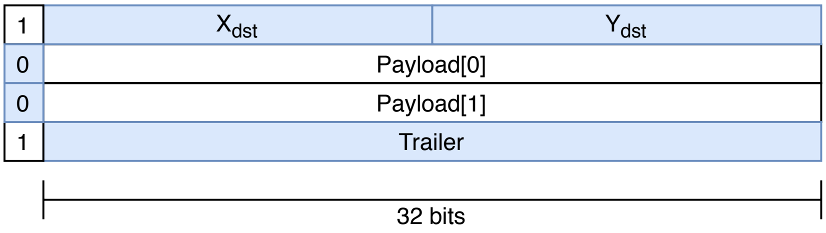
Fig. 3. Hamming code on buffers.

III. Materials and Methods

We described the baseline and the hardened routers using VHDL. Afterward, we synthesized them to FPGA using the Intel Quartus Prime design suite for costs and performance assessment. We selected the Intel Cyclone V 5CGTFD9E5F35C7 as the target device and disabled all the synthesis tool's optimizations options. Costs express the usage of resources—i.e., Look-Up Tables (LUTs) and Flip-Flops (FFs)— reported by the synthesis tool. Performance is given by the maximum operating frequency and the maximum throughput. The former is obtained by using TimeQuest Analyzer, and the latter is computed by dividing the number of bits transferred through all the communication channels by the simulated time.

Figure 4 shows the packet format used for router verification. It consists of a single bit to perform frame control, a single flit as the header, two payload Flow Control Units (flits), and a trailer. The header flit solely addresses the coordinates of the destination router. Both the header and the last payload flit (trailer) use '1' as the frame bit, while the regular payload flits use '0'.

For resilience assessment, we simulated a fault injection campaign on a single router using Mentor Graphics ModelSim simulator. For assessment, we defined a worst-case scenario with a workload that continuously injects 4-flit packets (i.e., injection rate equals 100%) to a fixed set of non-concurrent routes inside the router, thus enabling an assessment to be done at maximum loading, as shown in Fig. 5. Fault coverage is obtained by the number of propagated errors normalized in space, considering the amount of FFs, and along the time, adjusted by the throughput.



1	X_{dst}	Y_{dst}
0	Payload[0]	
0	Payload[1]	
1	Trailer	

Fig. 4. Packet format for verification.

For fault injection, we adopted the strategy proposed by the authors of [20], which injected SEU faults into the registers of a processor using built-in commands of the ModelSim simulator, causing a bit-flip. Each iteration of the fault injection strategy includes the following steps:

1) Simulation without fault injection to obtain a golden run.
2) Listing of all registers in the circuit and picking of a random one to inject a fault.
3) Randomly determination of the time occurrence of the fault within the simulation time window,
4) Simulation pre-fault injection.
5) Fault injection: bit flip forced into the selected register.
6) Simulation for a predefined time interval.
7) Comparison between the outputs outcoming the fault injection and the golden runs.

In the experiments, we evaluated the baseline and the hardened router in configurations without any fault-tolerant mechanism (which we refer to as STD – standard) or applying TMR to the controllers or the Hamming code to the buffers (which we refer to as HAM). We also varied the types of FSM used to implement the flow regulation (input

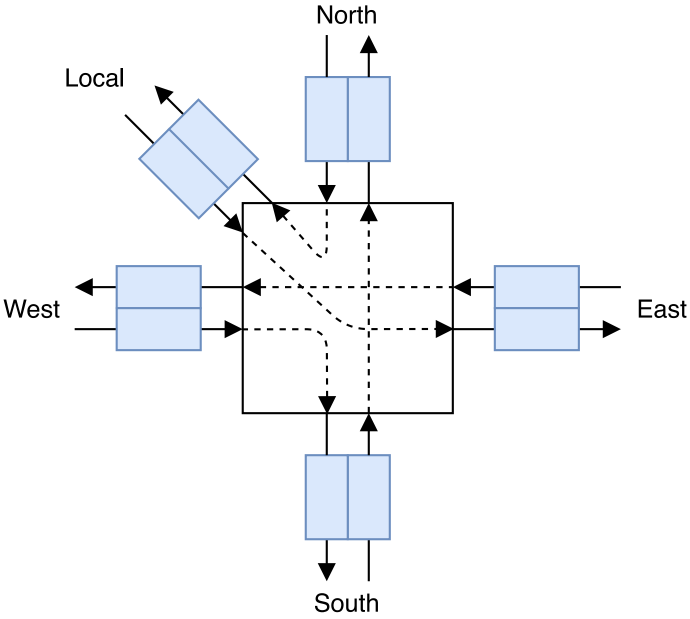


Fig. 5. Traffic scenario.

and output), routing, and arbitration controller, resulting in eight different architectural configurations for each router. The configurations used 32-bit wide flits and buffers capable of storing up to 4 flits. Each configuration performed 1,000 simulations running for 100 μs (i.e., the simulated time). This approach was applied to obtain a more accurate measurement of the error propagation rate among all scenarios.

IV. Results

A. Baseline Router

Table I presents the synthesis results for the baseline router. Note that all the controllers and buffers use the so-called standard (STD) implementation. As expected, we see that most of the Mealy-based controllers require more LUTs, due to the additional decoding of the output signals in states, and require less FFs because the FSMs encode few states. Moreover, increasing combinational logic implies a longer critical path and a lower operating frequency. Thus, the maximum operating frequency of the fully Moore approach is 70% higher than that of the configuration using only Mealy controllers. We highlight that the five buffers require the use of 895 LUTs, on average, and 695 FFs, being responsible for at least 50% of the combinational area and 90% of the sequential area, depending on the configuration used. This aspect has impact on the effectiveness of each fault-tolerant technique, as we show below.

TABLE I
Results for standard controllers and standard buffers

Controller			Buffer	LUTs	FFs	F_{max} (MHz)
Flow	**Routing**	**Arbitration**				
Moore STD	Moore STD	Moore STD	STD	1 367	750	225.33
Moore STD	Moore STD	Mealy STD	STD	1 353	744	200.64
Moore STD	Mealy STD	Moore STD	STD	1 412	745	150.53
Moore STD	Mealy STD	Mealy STD	STD	1 370	739	133.53
Mealy STD	Moore STD	Moore STD	STD	1 364	740	223.71
Mealy STD	Moore STD	Mealy STD	STD	1 356	734	195.50
Mealy STD	Mealy STD	Moore STD	STD	1 420	735	153.99
Mealy STD	Mealy STD	Mealy STD	STD	1 374	729	131.77

B. Hardened Controllers

Table II presents the synthesis results for the hardened controllers using TMR (the buffers are not protected). The technique increased the number of logical elements needed, which reflected in a longer critical path and

TABLE II
RESULTS FOR HARDENED CONTROLLERS AND STANDARD BUFFERS

Controller			Buffer	LUTs	FFs	F_{max} (MHz)
Flow	Routing	Arbitration				
Moore TMR	Moore TMR	Moore TMR	STD	1 621	860	184.88
Moore TMR	Moore TMR	Mealy TMR	STD	1 544	845	155.55
Moore TMR	Mealy TMR	Moore TMR	STD	1 681	845	146.84
Moore TMR	Mealy TMR	Mealy TMR	STD	1 587	830	115.38
Mealy TMR	Moore TMR	Moore TMR	STD	1 662	830	188.15
Mealy TMR	Moore TMR	Mealy TMR	STD	1 533	815	158.65
Mealy TMR	Mealy TMR	Moore TMR	STD	1 665	815	138.72
Mealy TMR	Mealy TMR	Mealy TMR	STD	1 586	800	115.69

consequent performance degradation. In comparison with the baseline router, we see an average increase of 16.9% in combinational logic and 12.2% in sequential logic. The maximum operating frequency decreases 14.2%, on average.

Fig. 6 presents the maximum throughput for the different configurations of the FSM type used in the baseline (STD) and hardened (TMR) implementations. Throughput is computed considering each configuration running for 10 Kcycles at its maximum operating frequency. As we can see, the higher the operating frequency, the highest the throughput. Moreover, we see that the use of TMR degrades performance in 13.7%, on average, and the highest levels of throughput are obtained by using the Mealy machine in the flow regulation controller and the Moore FSM in the arbitration controller.

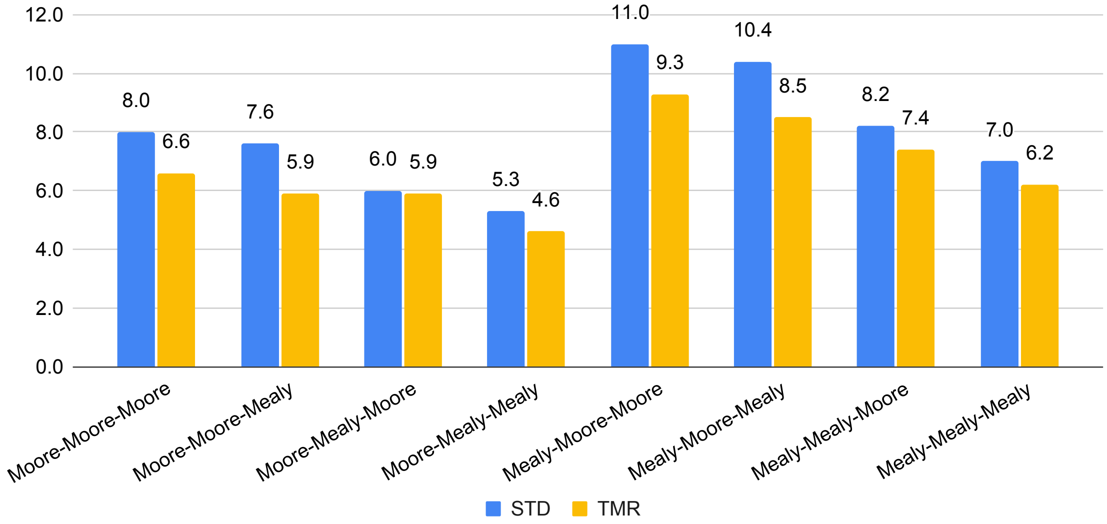


Fig. 6. Throughput of standard and hardened controllers (Gbit/s).

Fig. 7 presents the error propagation rate for the different configurations. We see that the use of TMR to protect the controllers has low effectiveness in reducing error propagation. Moreover, in configurations with an error propagation rate higher than 15%, the use of TMR propagates even more errors than the unprotected version. This result is because the faults affect the router randomly, and most of the faults are injected into the buffers' registers, which are not protected by the TMR technique. After, we show how reliability can be improved by hardening these buffers.

C. Hardened Buffers

Table III presents the synthesis results for a router configuration in which only the buffers are hardened (i.e., the controllers are not protected). Buffer protection resulted in a similar overhead in combinational (average of 533 LUTs) and sequential (120 FFs) logic resources among all combinations. The maximum operating frequency decreases when the Mealy machine is used for routing.

Figure 8 presents the throughput of the standard and hardened buffers configurations. We can see that the use of Hamming code on input buffers imposed a performance degradation of approximately 40% in comparison with the baseline router configuration. This high reduction observed in the maximum operating frequency is due to the increase in the critical path, given by the Hamming decoder at the buffers' output and the combinational logic of the Mealy-based routing controllers.

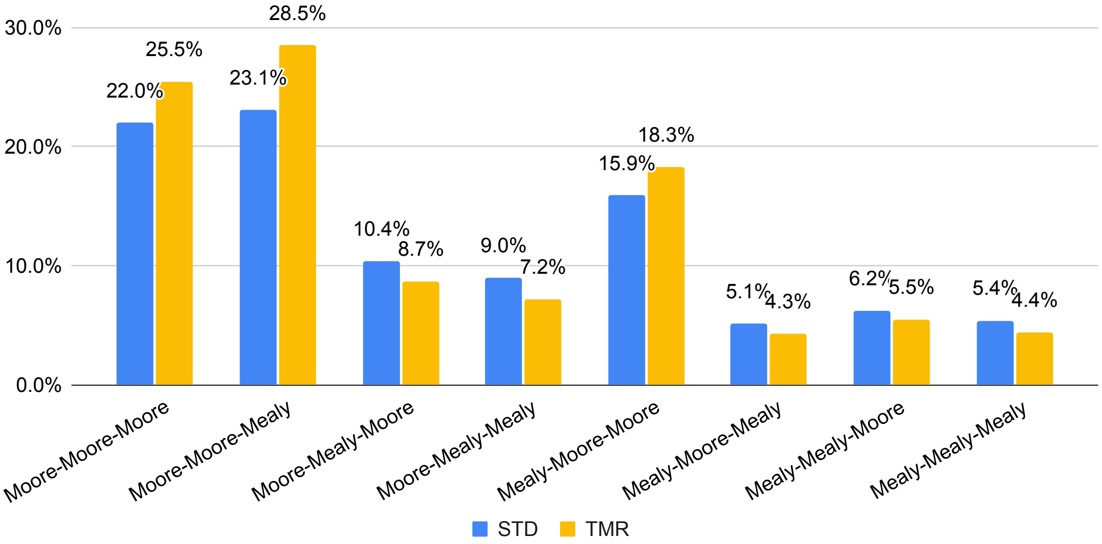


Fig. 7. Error rate of standard and hardened controllers.

TABLE III
RESULTS FOR STANDARD CONTROLLERS AND HARDENED BUFFERS

	Controller					
Flow	**Routing**	**Arbitration**	**Buffer**	**LUTs**	**FFs**	**F_{max} (MHz)**
Moore STD	Moore STD	Moore STD	HAM	1 900	870	123.44
Moore STD	Moore STD	Mealy STD	HAM	1 893	864	121.58
Moore STD	Mealy STD	Moore STD	HAM	1 942	865	100.19
Moore STD	Mealy STD	Mealy STD	HAM	1 905	859	87.40
Mealy STD	Moore STD	Moore STD	HAM	1 913	860	121.77
Mealy STD	Moore STD	Mealy STD	HAM	1 908	854	124.32
Mealy STD	Mealy STD	Moore STD	HAM	1 953	855	96.42
Mealy STD	Mealy STD	Mealy STD	HAM	1 904	849	87.40

Regarding the error rate, presented in Fig. 9, we see that the use of the Hamming code technique effectively reduces error propagation. As discussed earlier, these results confirm that the buffers are the components most affected by the injected faults.

D. Hardened Controllers and Buffers

As a last experiment, we analyzed the indicators for the fault-tolerant router implementing the TMR technique on the controllers and Hamming code on the buffers. Table IV presents the costs in terms of the resources occupied in all the configurations. As expected, these implementations have the highest occupancy of logic resources. As the router area increases, the longer wires increase the critical path and degrades performance. The maximum operating frequency and the throughput (Fig. 10) of this router are even lower than those of the configurations that implement only one fault-tolerant technique.

TABLE IV
RESULTS FOR HARDENED CONTROLLERS AND HARDENED BUFFERS

	Controller					
Flow	**Routing**	**Arbitration**	**Buffer**	**LUTs**	**FFs**	**F_{max} (MHz)**
Moore TMR	Moore TMR	Moore TMR	HAM	2 190	980	115.55
Moore TMR	Moore TMR	Mealy TMR	HAM	2 063	965	119.42
Moore TMR	Mealy TMR	Moore TMR	HAM	2 220	965	93.23
Moore TMR	Mealy TMR	Mealy TMR	HAM	2 156	950	76.19
Mealy TMR	Moore TMR	Moore TMR	HAM	2 189	950	117.08
Mealy TMR	Moore TMR	Mealy TMR	HAM	2 147	935	110.50
Mealy TMR	Mealy TMR	Moore TMR	HAM	2 223	935	92.65
Mealy TMR	Mealy TMR	Mealy TMR	HAM	2 144	920	81.67

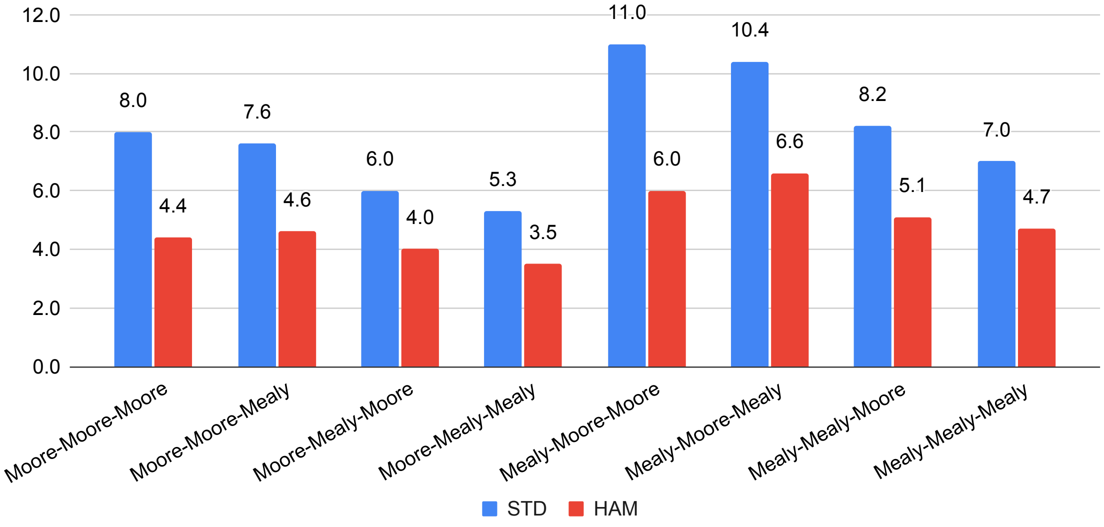


Fig. 8. Throughput of standard and hardened buffers (Gbit/s).

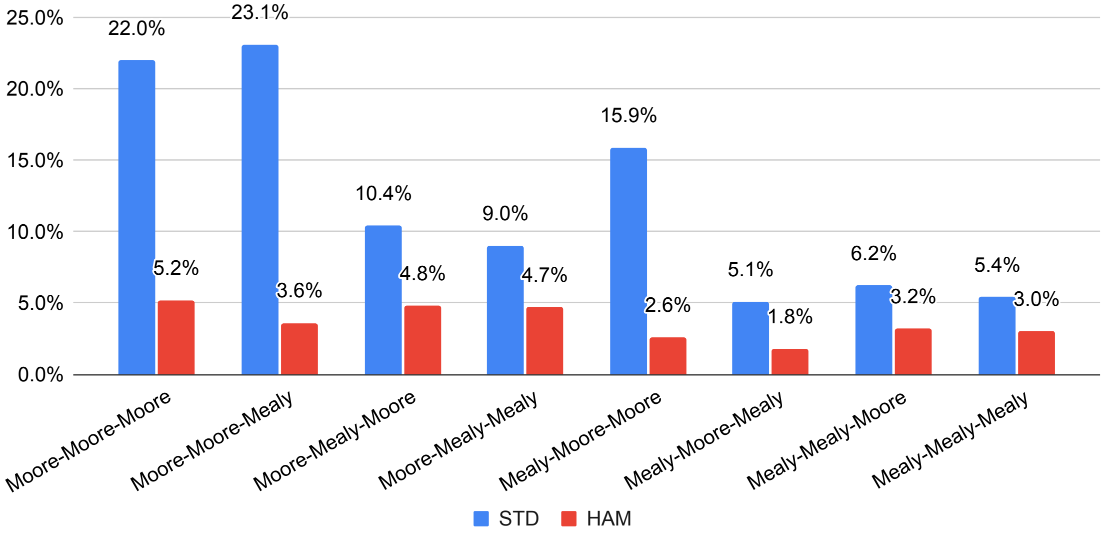


Fig. 9. Error rate of standard and hardened buffers.

Despite the high silicon overhead inferred by the hardening techniques, the router's resilience significantly increases in this configuration when comparing with the baseline router. As we can see in Fig. 11, the error propagation of a full-hardened router is less than 2%, on average.

E. Discussion

The experimental results enable designers and system integrators to understand some aspects related to the impact of the architectures of the controllers and the fault-tolerant techniques in the indicators of cost, performance, and reliability of the router, as follows:

- The adoption of the Mealy machine in the controllers increases the router's performance when applied to the flow controllers and resilience when applied to the routing controllers, because of the reduced number of states in the FSM compared to Moore implementations.
- Although the TMR technique is admittedly expensive, the impact of its use in controllers is not significant since most of the router's cost is due to buffers. On the other hand, the hardening of the controllers alone has low effectiveness in reducing the error propagation precisely because most of the faults are on the buffers.
- The Hamming code technique is highly effective in reducing the propagation of errors. However, this technique significantly degrades the router's performance, mainly when the routing controller is based on the Mealy machine because this combination results in a longer critical path than those of the other configurations.

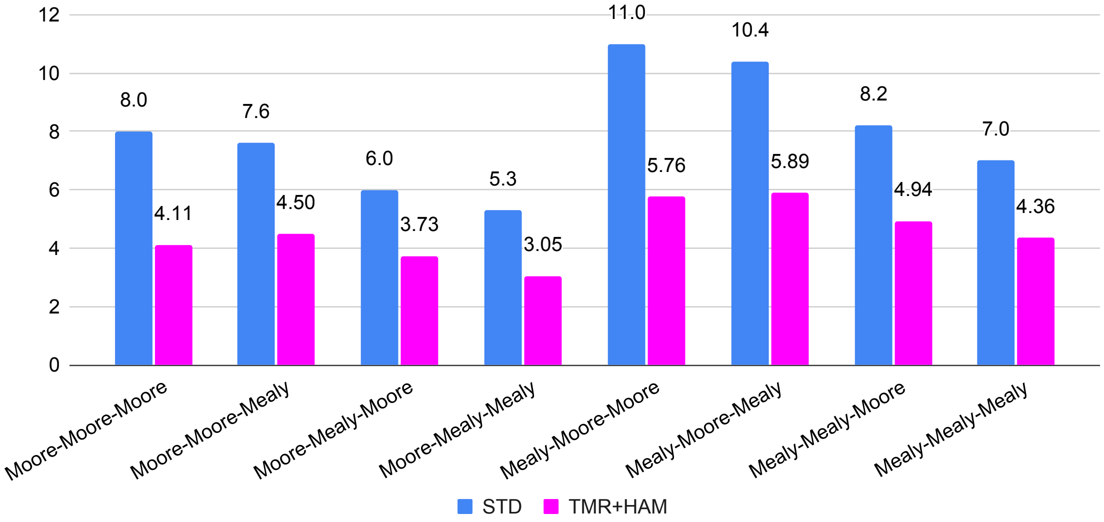


Fig. 10. Throughput of standard and hardened controllers and buffers (Gbit/s).

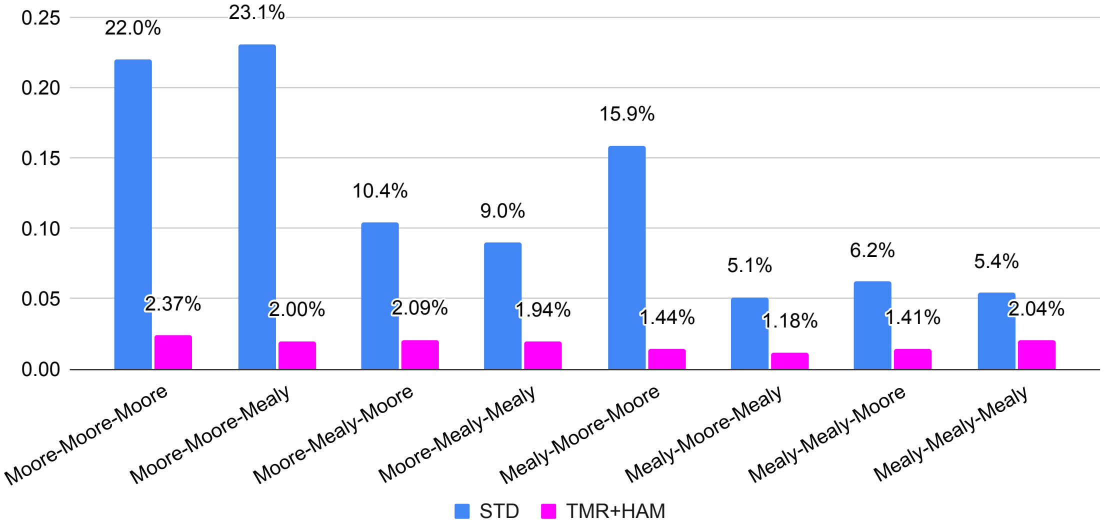


Fig. 11. Error rate of standard and hardened controllers and buffers

Despite presenting a cost and performance degradation, the use of Hamming code is justified in applications concerning reliability. Configurations that use Mealy in the flow regulation and Moore in the routing controller are those that present the highest throughput. When configured with a Mealy FSM for arbitration, it offers the lowest error propagation.

Although being designed to handle transient faults, the developed router may tolerate permanent single-bit stuck faults since their effect is covered by the embedded fault tolerance techniques. Furthermore, the router provides some protection against multi-bit faults, as long as they occur on different controllers and different buffer flits. However, a physical test campaign is required to obtain realistic fault coverage for given environment such as atmosphere or space.

V. Conclusion

This paper presented the design and assessment of a fault-tolerant router for an NoC. The experimental results enable a system designer to select the configurations of the NoC that best fit the application's requirements and costs. For example, an on-board satellite computer requires high reliability in communication, while signal processing applications prioritize throughput.

As future work, we plan to extend the verification campaign to analyze the error propagation rate when injecting multiple faults. We also intend to conduct the physical assessment in a particle accelerator to deepen the reliability analysis. The source code of the proposed architecture is available at [21].

Acknowledgments

This work was financed in part by CNPq – the Brazilian National Council for Scientific and Technological Development – Processes 315287/2018-7 and 436982/2018-8.

References

[1] D. J. Sorin, “Fault tolerant computer architecture,” *Synthesis Lectures on Computer Architecture*, vol. 4, no. 1, pp. 1–104, 2009.

[2] H. Zimmer and A. Jantsch, “A fault model notation and error-control scheme for switch-to-switch buses in a network-on-chip,” in *Int. Conf. on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. IEEE/ACM/IFIP, 2003, pp. 188–193.

[3] A. P. Frantz, F. L. Kastensmidt, L. Carro, and É. Cota, “Dependable network-on-chip router able to simultaneously tolerate soft errors and crosstalk,” in *Int. Test Conf. (ITC)*. IEEE, 2006, pp. 1–9.

[4] D. Park, C. Nicopoulos, J. Kim, N. Vijaykrishnan, and C. R. Das, “Exploring fault-tolerant network-on-chip architectures,” in *Int. Conf. on Dependable Systems and Networks (DSN)*. IEEE, 2006, pp. 93–104.

[5] A. Kohler, G. Schley, and M. Radetzki, “Fault tolerant network on chip switching with graceful performance degradation,” *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 6, pp. 883–896, 2010.

[6] H. Bokhari, H. Javaid, M. Shafique, J. Henkel, and S. Parameswaran, “Supernet: multimode interconnect architecture for manycore chips,” in *Design Automation Conf. (DAC)*. ACM/EDAC/IEEE, 2015, pp. 1–6.

[7] T. F. Pereira, D. R. Melo, E. A. Bezerra, and C. A. Zeferino, “Mechanisms to provide fault tolerance to a network-on-chip,” *IEEE Latin America Trans.*, vol. 15, no. 6, pp. 1034–1042, 2017.

[8] É. Cota, F. L. Kastensmidt, M. Cassel, M. Herve, P. Almeida, P. Meirelles, A. Amory, and M. Lubaszewski, “A high-fault-coverage approach for the test of data, control and handshake interconnects in mesh networks-on-chip,” *IEEE Trans. on Computers*, vol. 57, no. 9, pp. 1202–1215, 2008.

[9] S. Tosun, V. B. Ajabshir, O. Mercanoglu, and O. Ozturk, “Fault-tolerant topology generation method for application-specific network-on-chips,” *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 9, pp. 1495–1508, 2015.

[10] C. Chen, Y. Fu, and S. Cotofana, “Towards maximum utilization of remained bandwidth in defected noc links,” *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 2, pp. 285–298, 2017.

[11] I. Loi, F. Angiolini, S. Fujita, S. Mitra, and L. Benini, “Characterization and implementation of fault-tolerant vertical links for 3-d networks-on-chip,” *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 1, pp. 124–134, 2011.

[12] A.-M. Rahmani, P. Liljeberg, K. Latif, J. Plosila, K. R. Vaddina, and H. Tenhunen, “Congestion aware, fault tolerant, and thermally efficient inter-layer communication scheme for hybrid noc-bus 3d architectures,” in *Int. Symp. on Networks-on-Chip (NoCs)*. IEEE/ACM, 2011, pp. 65–72.

[13] A. Eghbal, P. M. Yaghini, N. Bagherzadeh, and M. Khayambashi, “Analytical fault tolerance assessment and metrics for tsv-based 3d network-on-chip,” *IEEE Trans. on Computers*, vol. 64, no. 12, pp. 3591–3604, 2015.

[14] A. C. Pinheiro, J. A. Silveira, D. A. Tavares, F. G. Silva, and C. A. Marcon, “Optimized fault-tolerant buffer design for network-on-chip applications,” in *Latin American Symposium on Circuits & Systems (LASCAS)*. IEEE, 2019, pp. 217–220.

[15] S. Pasricha and Y. Zou, “A low overhead fault tolerant routing scheme for 3d networks-on-chip,” in *Int. Symp. on Quality Electronic Design (ISQED)*. IEEE, 2011, pp. 1–8.

[16] C. Feng, Z. Lu, A. Jantsch, M. Zhang, and Z. Xing, “Addressing transient and permanent faults in noc with efficient fault-tolerant deflection router,” *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 6, pp. 1053–1066, 2013.

[17] D. R. Melo, C. A. Zeferino, L. Dilillo, and E. A. Bezerra, “Analyzing the error propagation in a parameterizable network-on-chip router,” in *Latin-American Test Symposium (LATS)*. IEEE, 2019, pp. 1–6.

[18] ——, “Maximizing the inner resilience of a network-on-chip through router controllers design,” *Sensors*, vol. 19, no. 24, p. 5416, 2019.

[19] R. W. Hamming, “Error detecting and error correcting codes,” *The Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, 1950.

[20] R. Travessini, P. R. Villa, F. L. Vargas, and E. A. Bezerra, “Processor core profiling for seu effect analysis,” in *Latin-American Test Symposium (LATS)*. IEEE, 2018, pp. 1–6.

[21] XARC. eXtensible ARChitecture. [Online]. Available: http://xarc.org/