Implementation of fault tolerance techniques for integrated network interfaces

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

▶ **To cite this version:**

Douglas Rossi de Melo, Cesar Zeferino, Antonio Ramos, Luigi Dilillo, Eduardo Augusto Bezerra. Implementation of fault tolerance techniques for integrated network interfaces. 3rd IAA Latin American CubeSat Workshop (IAA-LACW 2018), Dec 2018, Ubatuba, Brazil. lirmm-02008453

HAL Id: lirmm-02008453

https://hal-lirmm.ccsd.cnrs.fr/lirmm-02008453

Submitted on 7 Oct 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

This is a self-archived version of an original article.
This reprint may differ from the original in pagination and typographic detail.

Title: Implementation of Fault Tolerance Techniques for Integrated Network Interfaces

Author(s): Douglas Melo, Cesar Zeferino, Antonio Ramos, Luigi Dilillo, and Eduardo Bezerra

DOI:

Published:

Document version: Post-print version (Final draft)

Please cite the original version:
Douglas Rossi de Melo, Cesar Zeferino, Antonio Ramos, Luigi Dilillo, Eduardo Augusto Bezerra. "Implementation of fault tolerance techniques for integrated network interfaces," 3rd IAA Latin American CubeSat Workshop (IAA-LACW), Dec 2018, Ubatuba, Brazil.

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.

IMPLEMENTATION OF FAULT TOLERANCE TECHNIQUES FOR INTEGRATED NETWORK INTERFACES

Douglas Melo[(1),(2),(3)], **Cesar Zeferino**[(1)], **Antonio Ramos**[(4)], **Luigi Dilillo**[(3)], **Eduardo Bezerra**[(2),(3)]

[(1)]*Laboratory of Embedded and Distributed Systems, University of Vale do Itajaí, Brazil, {drm, zeferino}@univali.br*

[(2)]*Laboratory of Communications and Embedded Systems, Federal University of Santa Catarina, Brazil, {drm, eduardo.bezerra}@eel.ufsc.br*

[(3)]*Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier, France, {drm, dilillo, eduardo.bezerra}@lirmm.fr*

[(4)]*Department of Science and Industry Systems, University of South-Eastern, Norway, antonio.ramos@usn.no*

Keywords: *Networks-on-Chip, Network Interface, Fault Tolerance*

The increasing demand for safety and mission-critical computational systems to operate in hostile environments, along with the component miniaturization for such systems, have triggered the need for developing fault tolerance techniques to mitigate the incidence of system failures and increase reliability. Communication architectures used in computers for aerospace applications are made up of a growing number of processing cores. Some approaches that are proposed in the literature do not meet the communication requirements of future on-board computers composed of multiple cores. Networks-on-Chip are the successors of the bus for multicore interconnection, integrating cores by the means of Network Interfaces. This work discusses the implementation and evaluates the performance of the Hamming encoding, the Triple Modular Redundancy (TMR), and temporal redundancy into the eXtensible Interface for Routing Unit (XIRU) Network Interface. Five scenarios are considered based on the combination of these techniques. Results show that the TMR in FIFO buffers is the most costly technique in terms of area usage, the Hamming code has the highest power dissipation, while the temporal redundancy has the lower operation frequency. Finally, a modified version of XIRU to integrate cores into Network-on-Chip systems with reliability requirements is devised. We intend to use the interface to integrate the cores of future cubesats developed in the FloripaSat project.

1. Introduction

The demand for real-time embedded computing has been increasing significantly in recent years. The design of such systems is always subject to a specific set of requirements that includes performance, real-time execution capability, response time, with reduced size and costs [1]. Modern embedded systems usually require processors with increased performance, while keeping physical dimensions and power dissipation as low as possible.

Recent advances in integrated circuits manufacturing technology allowed the construction of complete computing systems in a single chip, known as Systems-on-Chip (SoCs). The components of a SoC can be designed independently of each other and made available by manufacturers in the form of IP (Intellectual Property) cores. In SoCs with few cores, components are interconnected by shared buses. However, this communication architecture does not meet the requirements of SoCs composed of several dozen cores, which demand parallelism and performance scalability [2].

A solution to overcome the bus limitations is represented by NoCs (Networks-on-Chip), scalable and reusable architectures that provide parallelism in communication. In SoCs based on NoCs, each core of the system is connected to one router, which is connected to neighboring routers. However, a Network Interface (NI) between cores and routers is necessary for an effective communication [3].

Miniaturization of the circuits and the increasing operating frequency result in a higher incidence of faults in SoCs [4]. The occurrence of faults can be a serious problem, especially in the presence of external interference. For such critical cases, SoC designers have to use fault tolerance techniques to ensure proper operation of these systems.

In general, fault tolerance techniques are based on redundancy and can be classified into spatial, temporal, and information. Spatial redundancy consists of component replication and the insertion of a voter that compares the outputs of these components and chooses the value on a majority principle. In temporal redundancy, a component performs its operation more than once and a voter compares the results of those executions. Information redundancy consists of adding bits to the message to detect and correct possible errors. Therefore, the implementation of fault tolerance mechanisms results in some overhead, whether in performance, silicon area, or dissipated power [5].

Several cores for SoCs employ fault tolerance techniques in their design. Examples are the soft-core LEON3 [6], commonly used in space applications, and the works of [7] and [8] on System-on-Chip Interconnection Network (SoCIN) [2]. SoCIN uses the XIRU (eXtensible Interface for Routing Unit) [9] to integrate cores based on the Avalon bus. In [10], the authors added support for the AMBA-AHB bus to XIRU, without applying any fault tolerance technique.

This work aims at implementing and evaluating the use of fault tolerance techniques on XIRU. We have implemented Hamming encoding, the TMR (Triple Modular Redundancy), and temporal redundancy techniques, as well as combinations of those. We observed a significant area overhead when combining TMR and the Hamming encoding technique. The combination of techniques presents the highest fault coverage. A significant reduction in operating frequency was observed using temporal redundancy, while no significant changes were noticed in terms of power dissipation.

The remainder of this paper is organized as follows. Section 2 presents a brief background on Network Interfaces, Section 3 describes the proposed techniques, while Section 4 discusses the performance evaluation results in various scenarios. Finally, Section 5 presents conclusions and suggestions for future works.

2. Background

Network Interface (NI) is the component that provides communication services and translates protocols between core and router. An NI comprises a front-end and a back-end [4], as shown in Fig. 1. The front-end can be represented by the session layer, whereas the lower layers of the OSI (Open System Interconnection) model represent the back-end. The front-end communicates with the core and it is responsible for facilitating reuse across platforms owing to its standardized point-to-point protocol. Conversely, the back-end communicates with the router and provides high-level communication services (session layer), reliable data transfer (transport layer), packaging and routing (network layer), error detection (link layer), and resolution of physical problems (physical layer) [3].

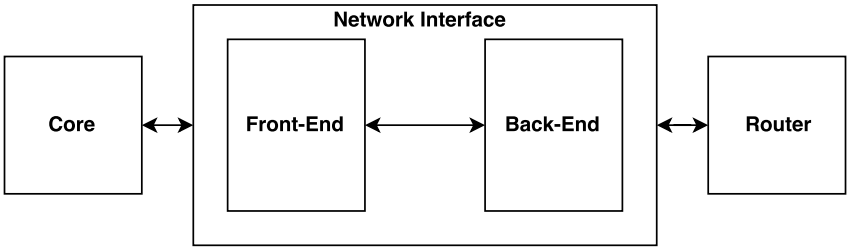

Figure 1: Network Interface [4].

3. Development

In this work, we implement and add fault tolerance components to the XIRU Network Interface (Fig. 2). We used VHDL for hardware description on the Intel Quartus II (version 13.0sp1), the Mentor Graphics ModelSim (version 10.1d) for simulation, and the Altera DE2 kit with the EP2C35F672C6 FPGA for prototyping. We structured this development into five incremental scenarios, as described below.

3.1. Scenario 1: Hamming encoding

The first scenario consists of integrating the Hamming encoder and decoder at the edge of the Network layer, as shown in Fig. 3. Fault tolerance occurs in end-to-end communication, i.e., encoding packets going out to the router, and decoding and correcting packets arriving from the router. The input of the Hamming encoder is 34 bits long, while the output has 41 bits.

The Hamming code detects up to two errors and can correct a single one (SECDED – Single Error Correction, Double Error Detection). In an error-free scenario, the result of the Hamming calculation and the parity calculation is equal to 0. In the case of a single error, both Hamming and parity result in a different value, indicating an error. In the case of a double error, the Hamming calculation results in a value other than zero, while the parity bit is inverted again.

3.2. Scenario 2: Triple Modular Redundancy (TMR) on controllers

The second scenario is the application of TMR on the XIRU controllers. We triplicated the Flow Controller (FC) components of the Network layer, and the controllers of the Generic layer, as shown in Fig. 4. We have also added a voter for each replicated component.

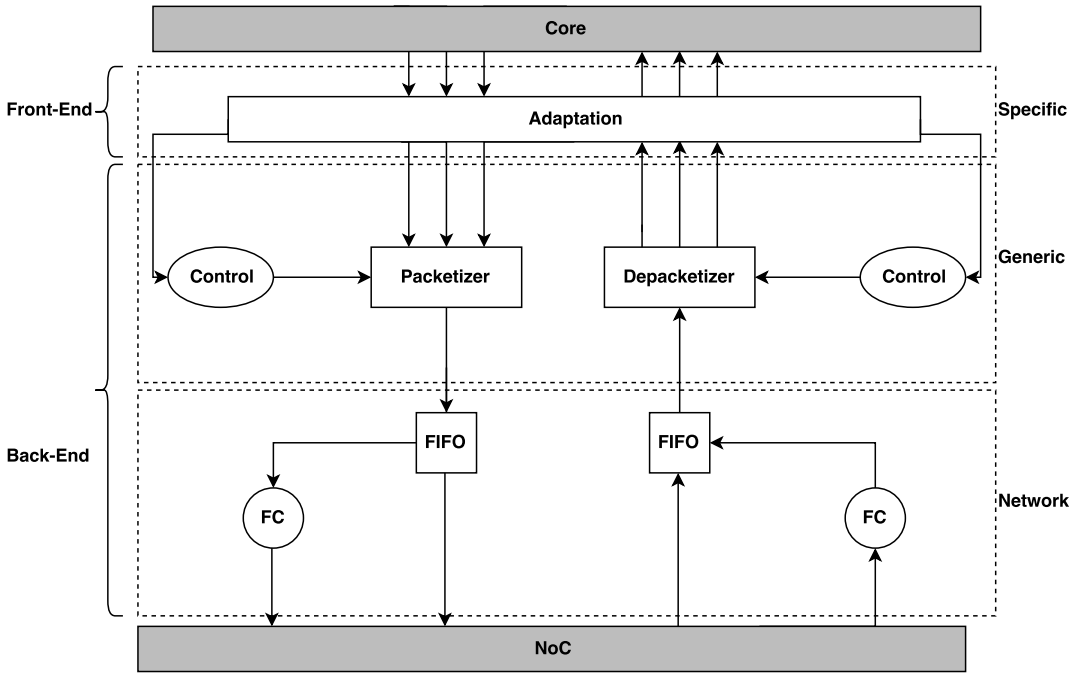

Figure 2: XIRU Network Interface [9].

3.3. Scenario 3: TMR on FIFOs

In this scenario, we applied TMR on the XIRU memorization components (FIFOs), as shown in Fig. 5. These FIFOs are in the Network layer, and are the only components that have been modified for this scenario.

3.4. Scenario 4: Hamming and TMR

This scenario applies end-to-end protection using the Hamming encoding and TMR in the controllers and memorization components, as shown in Fig. 6.

3.5. Scenario 5: Temporal redundancy on controllers

This scenario applies temporal redundancy on the controllers of the Generic layer. The execution is processed three times and, in the end, a voter compares the results of the executions and sets the output signal. Fig. 7 represents the technique that have been applied to the XIRU network interface.

4. Results

After synthesis and verification, we have analyzed and compared the different implementations with respect to use of FPGA resources, i.e, look-up tables (LUTs) and flip-flops (FFs), and performance in terms of maximum operation frequency and average power dissipation. Table I shows the obtained results. As can be observed, Scenario 4 is the one with highest resource utilization. This is expected since it implements the Hamming encoding and TMR on the controllers and on the FIFOs. Indeed, there was an increase of 170.82% of LUTs in the master unit against 135.63% in the slave unit in comparison to the original implementation. Concerning the flip-flops, the overhead of this scenario was 143.60% in the master unit against 137.95% in the slave unit. The lowest performance was verified in Scenario 5, with a decrease of 34.40% of the maximum operating frequency in the master unit and 37.30% reduction in the slave unit. Moreover, the addition of new circuits to the system increases the critical path, thereby causing a degradation in the maximum frequency of operation. Finally, no significant variation in terms of dissipated power was observed across the different scenarios, presenting an average consumption of 135 mW.

There are some differences among the scenarios related to fault coverage rate. Hamming code (Fig. 3) works as an end-to-end solution, so it can't detect nor correct any fault in the NI infrastructure. The TMR on controllers (Fig. 4) and TMR on FIFOs (Fig. 5) can mask single event and also single permanent faults. The combination of the former techniques (Fig. 6) protects control and memorization structures. In contrast, the temporal redundancy (Fig. 7) protects from transient faults but is ineffective against permanent ones.

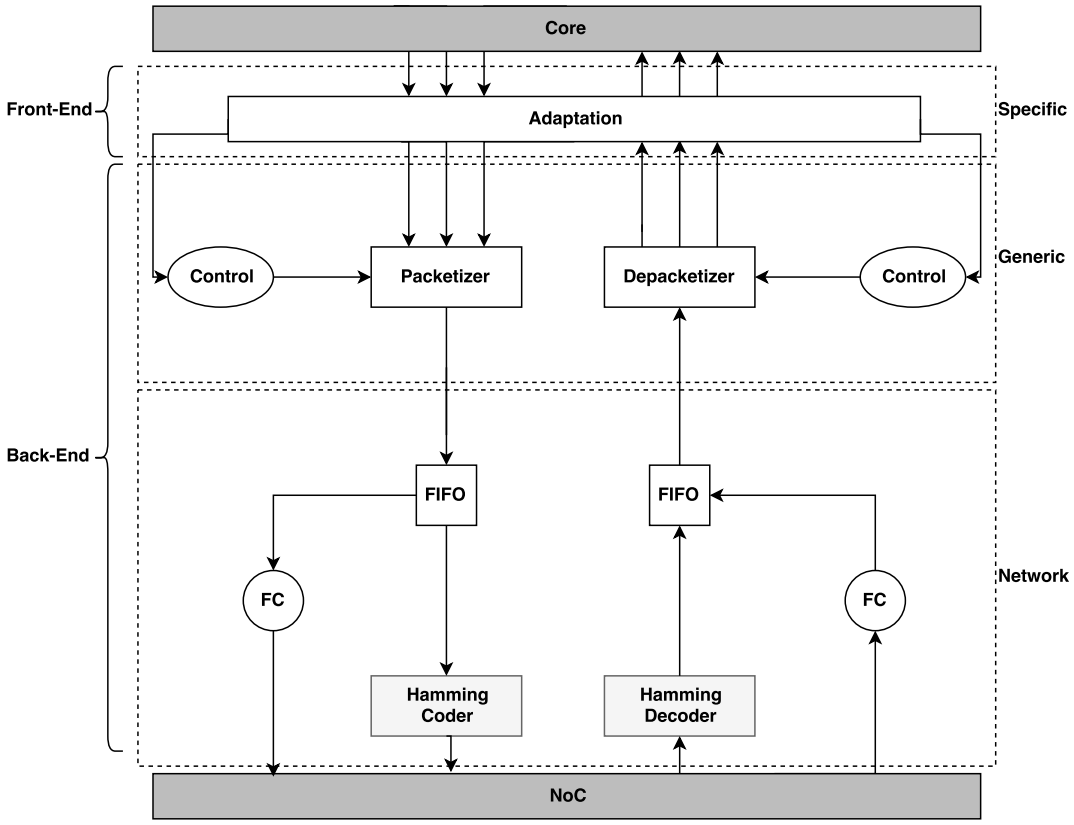

Figure 3: Scenario 1: Hamming encoding.

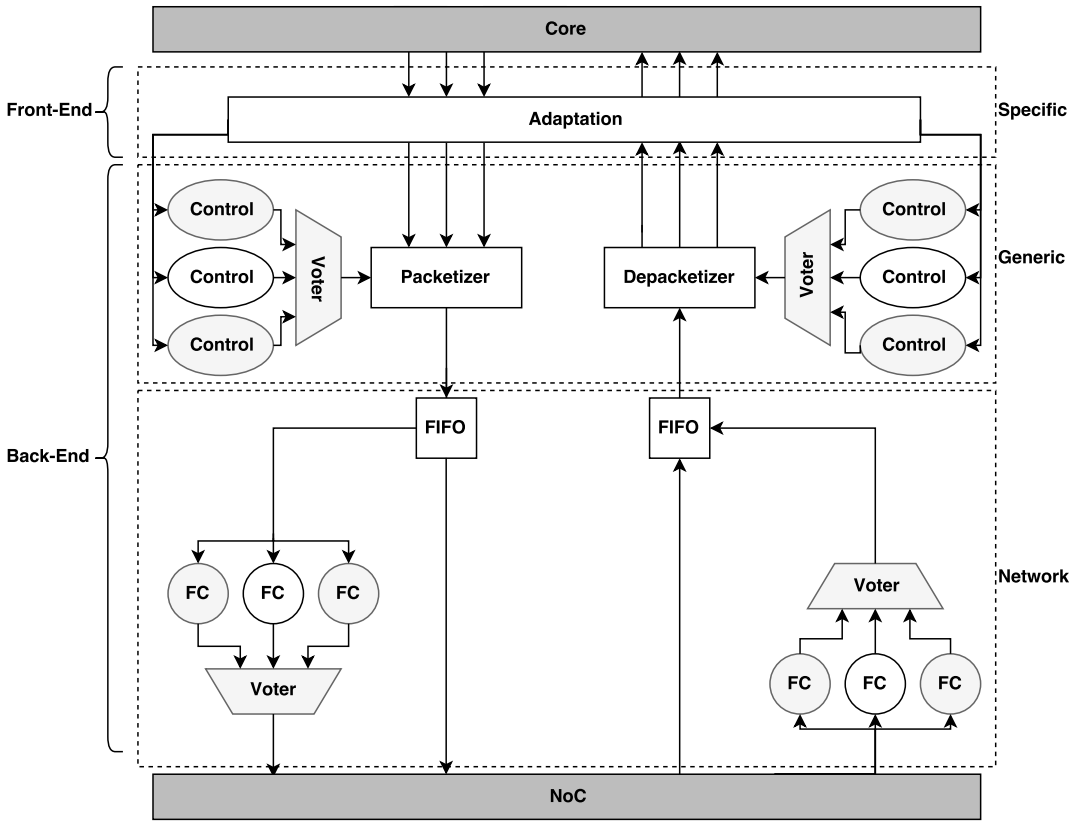

Figure 4: Scenario 2: TMR on controllers.

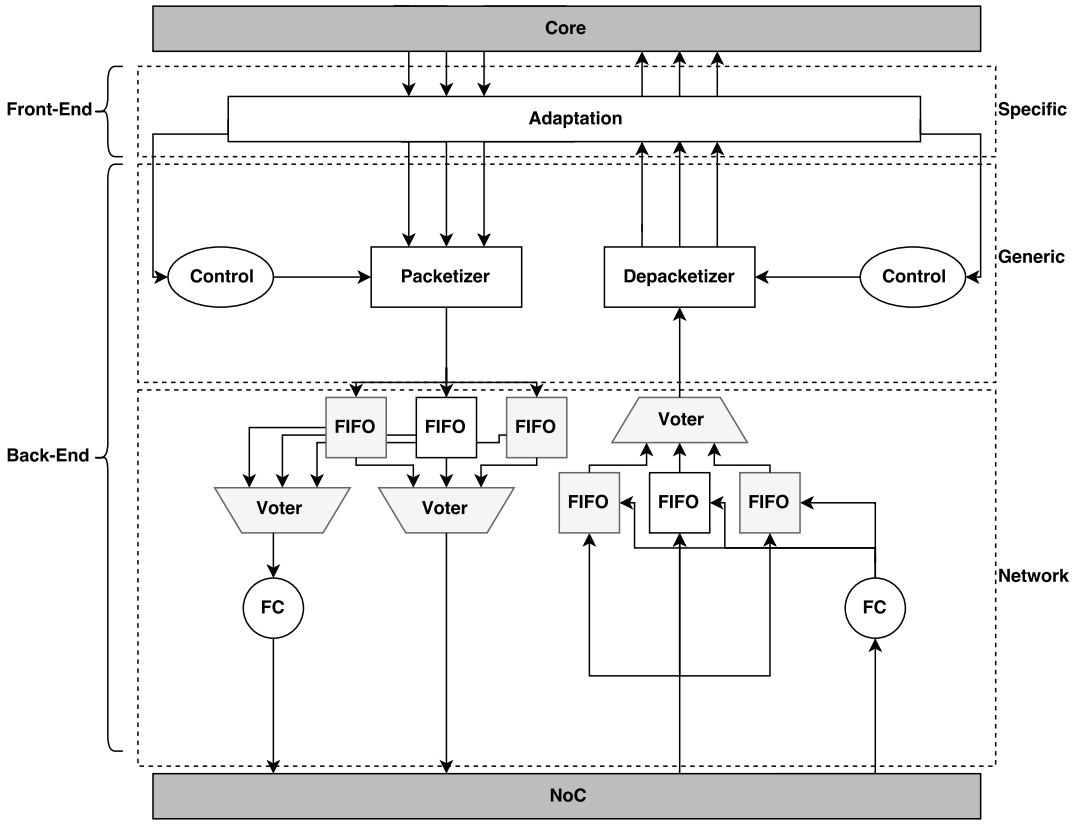

Figure 5: Scenario 3: TMR on FIFOs.

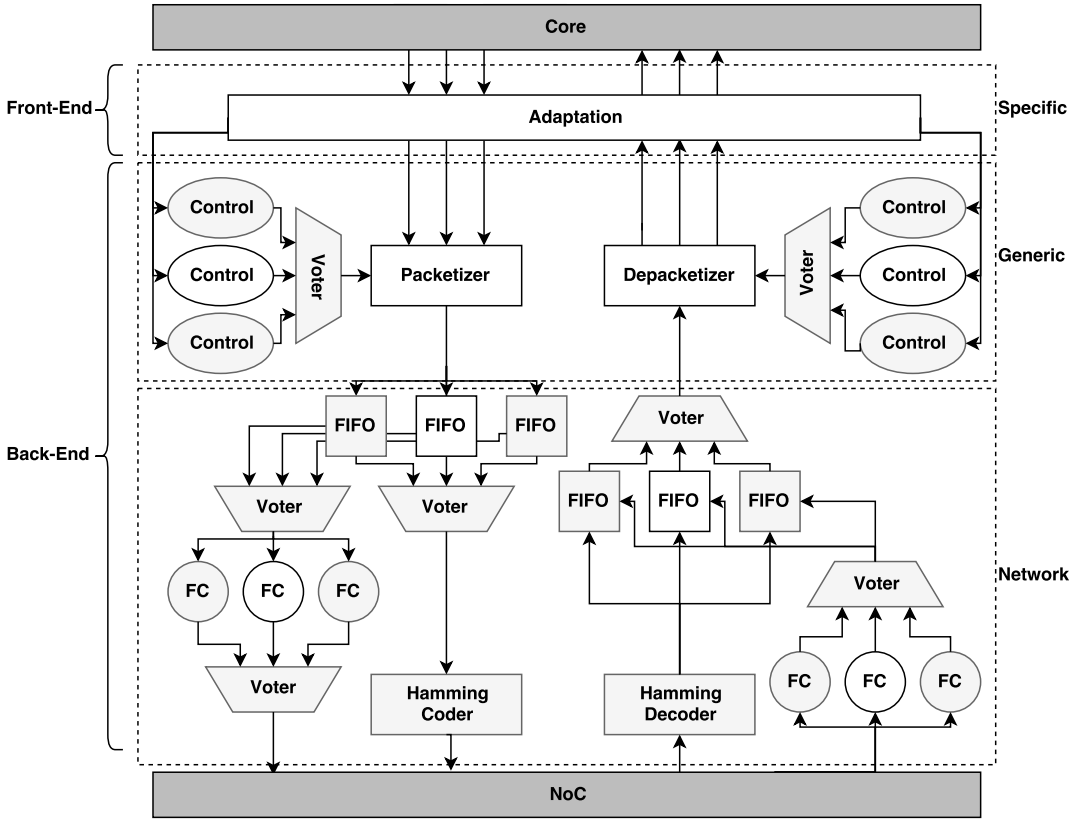

Figure 6: Scenario 4: Hamming and TMR.

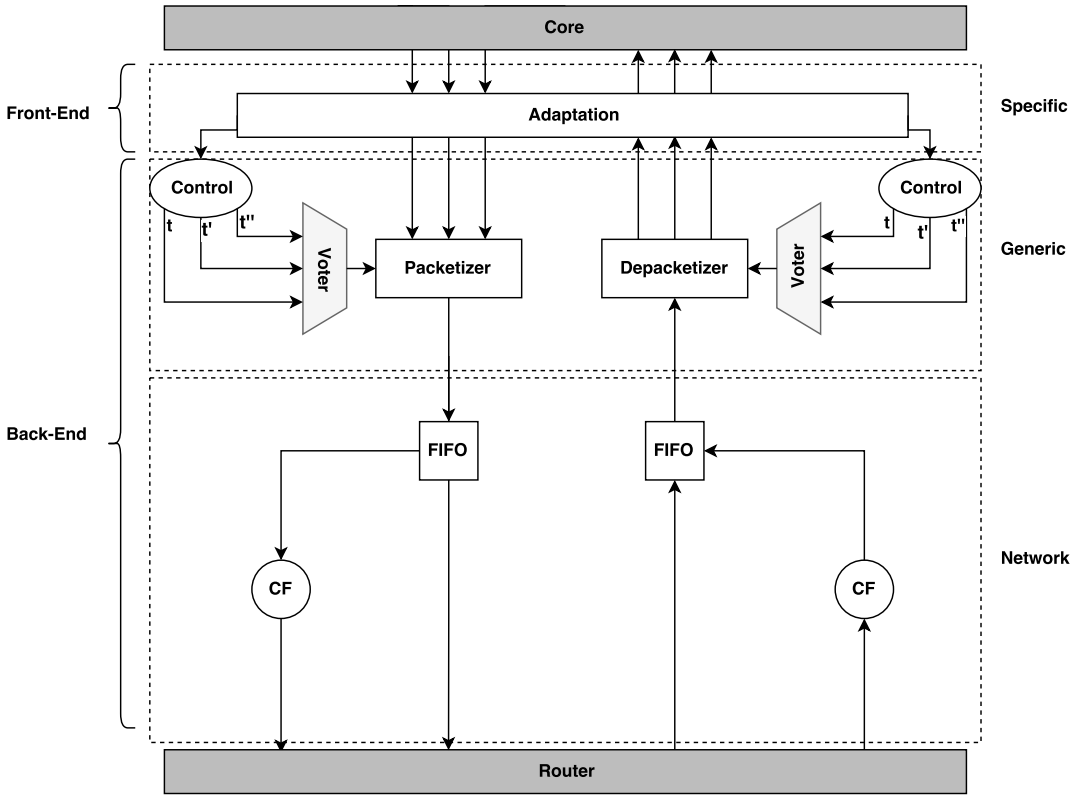

Figure 7: Scenario 5: Temporal redundancy on controllers.

Table 1: Synthesis Results

Scenario	Component	LUTs	FFs	Fmax(MHz)
Original	Master	281	422	212.04
	Slave	407	448	221.73
1	Master	287	422	212.27
	Slave	410	448	216.26
2	Master	327	448	178.44
	Slave	517	486	187.06
3	Master	701	994	195.96
	Slave	823	1020	202.10
4	Master	761	1028	165.84
	Slave	959	1066	192.34
5	Master	311	448	139.10
	Slave	503	486	139.02

5. Conclusion

This work discusses the application of fault tolerance techniques in a network interface for use in reliable systems based on NoCs. As expected, the use of redundancy techniques resulted in an increase in the utilization of logical resources and in the performance degradation, especially in the scenario combining the Hamming encoding and TMR.

We are currently computing the fault coverage and evaluating the communication latency and energy consumption in each scenario. As future work we intended to adapt the network interface for the next generation of FloripaSat platform [11], using a SoC in which all components (processor, network interface, and routers) are protected by reliability techniques.

Acknowledgments

The authors would like to thank the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES), the University of Vale do Itajaí (UNIVALI), and the Norwegian Centre for International Cooperation in Education (SIU), for supporting this work.

References

[1] F. Vahid, T. Givargis, Embedded system design: a unified hardware/software introduction, New York: Wiley, 2002.

[2] C. A. Zeferino, A. A. Susin, Socin: a parametric and scalable network-on-chip, in: Integrated Circuits and Systems Design, 2003. SBCCI 2003. Proceedings. 16th Symposium on, IEEE, pp. 169–174.

[3] G. De Micheli, L. Benini, Networks on chips: technology and tools, Academic Press, 2006.

[4] D. Bertozzi, The data-link layer in noc design, Micheli, G.; Benini, L." Networks on Chips: Tecnology and Tools", New York: Morgan Kaufmann Publishers (2006).

[5] D. J. Sorin, Fault tolerant computer architecture, Synthesis Lectures on Computer Architecture 4 (2009) 1–104.

[6] A. Gaisler, S. Göteborg, Leon3 multiprocessing cpu core, Aeroflex Gaisler, February (2010).

[7] F. Veiga, C. A. Zeferino, Implementation of techniques for fault tolerance in a network-on-chip, in: Computing Systems (WSCAD-SCC), 2010 11th Symposium on, IEEE, pp. 80–87.

[8] T. F. Pereira, D. R. de Melo, E. A. Bezerra, C. A. Zeferino, Mechanisms to provide fault tolerance to a network-on-chip, IEEE Latin America Transactions 15 (2017) 1034–1042.

[9] D. Melo, M. Wangham, C. Zeferino, Xiru: Interface de rede extensível para integração de núcleos a uma rede-em-chip, Revista de Informática Teórica e Aplicada 21 (2014) 10–31.

[10] F. L. Gaya, C. A. Zeferino, D. R. Melo, E. A. Bezerra, Amba-ahb network interface for core interconnection in a network-on-chip, Iberchip (2017).

[11] P. Villa, L. Slongo, J. Salamanca, V. Martins, F. Silva, S. Martinez, L. Mariga, B. Eiterer, I. Vidal, V. Menegon, L. Coelho, X. Travassos, K. Paiva, A. Spengler, F. Souza, L. Becker, D. Lettnin, E. Bezerra, A complete cubesat mission: the floripa-sat experience, in: 1st IAA Latin American Cubesat Workshop, volume 2, pp. 307–314.