AXI Lite redundant on-chip bus interconnect for high reliability systems

Jesús Lázaro Armando Astarloa Aitzol Zuloaga José Ángel Araujo Jaime Jiménez

Nowadays, system-on-chips have become critical since they support more and more safe applications due to their flexibility. However, they are susceptible to single-event upsets because the memory cell size has significantly shrunk. This article presents a triple redundant on-chip interconnect bus that provides low-speed peripherals with high reliability. In addition to correcting single errors and detecting duplicated ones, the proposed circuit offers zero latency and is transparent for both the embedded processor and the peripherals. These characteristics make it suitable for hard real-time applications. At the same time, the impact on area and power consumption is minimal. [1]

Keywords: FPGA, SoC, Redundancy, AXI

1 Introduction

Nowadays, SRAM FPGAs are widely used in many applications, such as autonomous vehicles, signal processing, industry 4.0, and embedded applications [1], thanks to their growth in capabilities.

Due to the nature of SRAM-based FPGAs, they are susceptible to SEU induced by high-energy particles [2, 3], which limits their usage in safety and mission-critical applications. In SRAM-based FPGAs, the programmability is controlled by the SRAM cell-based configuration memory. With every advance in reducing voltage and dimensions of the SRAM cell, its capacitance is decreased. This decrement increases the vulnerability of the cell to particles of lower energy. When an event occurs, the actual behavior of the circuit may get altered until the FPGA is programmed again [4].

The most common SEU hardening techniques include TMR [5], Partial TMR [6], and Dynamic Partial Reconfiguration [7]. The TMR implementation uses three identical logic blocks performing the same task in parallel and compares the outputs by a majority voter. In safety-critical applications, TMR techniques are essential.

TMR may be applied with different granularities. The finest granularity of TMR is achieved when voting is applied for all registers in the design [8]. A more coarse granularity can be achieved if triplication and voting are applied through the device based on larger modules and their respective outputs [5]. The robustness increases when voting is performed over smaller modules, but so does the implementation cost in terms of FPGA utilization. Furthermore, a fine granularity reduces the highest achievable system speed because voters are added to the critical path [9].

In this article, the reliability of FPGA systems is increased by introducing coarse-grained TMR in AXI4-Lite on-chip peripherals. Additionally, IP core — AXILiteRedundant — has a minimal area, power impact, and maximum speed. At the same time, the IP core reduces the processor's overhead by presenting a single peripheral structure rather than the triple system produced by the TMR. This characteristic allows more straightforward software to process the incoming data, allowing hard real-time operations.

The scope of this research project includes implementing the AXILiteRedundant architecture in a ready-to-use HDL-described IP. It will be packaged in the format specified by the Silicon vendors [10], allowing the engineers to integrate this block into the SoC design seamlessly. The implementation includes the basic software driver to configure and manage the IP.

Redundancy can be exploited through an on-chip building block in SoC designs for critical systems, where the failure rate shall be maintained into shallow values. This IP enhances the reliability of the user IP that wraps. An example of a typical user IP that can be rugged using this wrap is an IP that implements the standard communication protocol for satellite platforms, SpaceWire [11]. This wrap and the redundant implementation for a given IP shall be combined with other mechanisms, such as lock-step CPU operation or SRAM memory scrubbing, to build a complete and robust SoC solution. The primary sector that benefits from these innovations is Space [12]. However, the demand for robust SoC platforms is increasing in other sectors, such as Transportation, Defense, and Medical.

The rest of the paper is divided into the following sections: Section 2 presents work done previously in the field. Section 3 describes the proposed solution. Section 4 describes the obtained results. Section 5 compares it with some similar ones. Section 6 concludes the paper.

[1] This work has been supported, within the fund for research groups of the Basque university system IT1440-22, by the Department of Education and, within PILAR ZE-2020/00022 and COMMUTE ZE-2021/00931 projects, by the Hazitek program, both of the Basque Government; the latter also by the Ministerio de Ciencia e Innovación of Spain through the Centro para el Desarrollo Tecnológico Industrial (CDTI) within the

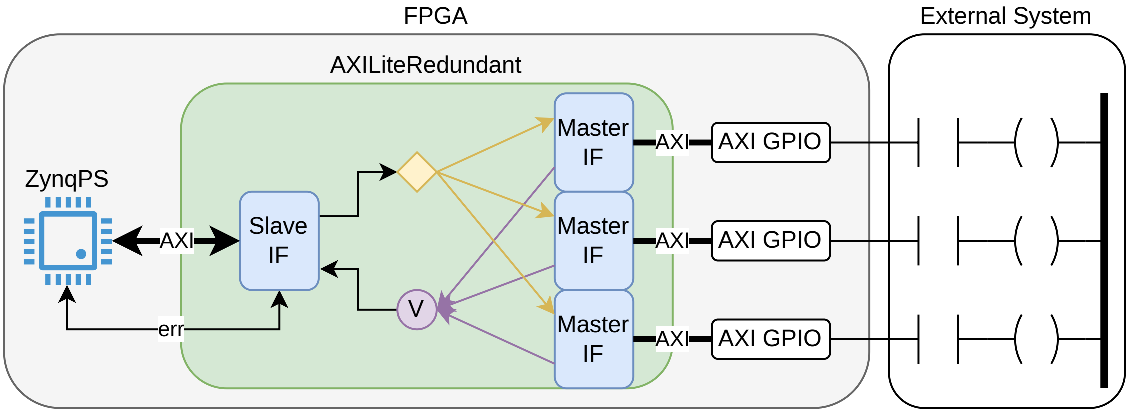


Figure 1: Simplified diagram of the IP core. Slave to Master is triplicated. Master to Slave is majority-voted. Error backchannel (*err*) is also depicted. It will inform the processor of any issues.

2 Previous Work

Redundancy in SoC can take multiple approaches and exists at different levels. The most common ones are: (a) System architecture, (b) Microprocessor, (c) Communications, (d) IO, and (e) FPGA.

System architecture level redundancy is a common approach in SoC [2, 13, 14]. Key areas where it is used include flight control, industry, and satellite design. The main approaches are component redundancy with TMR and redundant intra-chip communication buses.

Microprocessor level redundancy [15, 16, 17] uses multiple processors inside the same SoC. The most common approach is lockstep. In other words, several processors run the same program with the same input data. Newer approaches include using heterogeneous SoCs —systems with more than one type of processor— to mitigate potential hidden errors in the processor's design.

Communications level redundancy [18, 19] uses multiple communication channels to connect to external elements.

IO level redundancy [20, 21, 22] is related to the electrical level redundancy of the signals. Common approaches are routing the same signal to multiple inputs or joining the signals from multiple outputs with some circuitry to avoid short circuits.

FPGA-level redundancy engulfs a variety of issues. FPGAs are incredibly flexible, which allows internal redundancy [23, 24]. They also enable some previous redundancy schemes [15].

At the same time, FPGAs suffer from specific issues such as SEUs due to numerous SRAM cells. In order to mitigate this issue, several approaches, such as scrubbing, have been proposed [25, 26, 27].

This paper presents a new approach to enhancing reliability in SoC. It is based on including redundancy of internal buses for low-speed peripherals. The literature has several examples in this area of research.

Bertozzi et al. [23] proposed a redundant bus coding to increase redundancy in power-constrained systems. They tested known algorithms such as Hamming and CRC. One interesting point is their focus on power, which leads to the conclusion that retransmission is, in many cases, more efficient. However, this approach is not valid for time-critical systems. In addition, it is not transparent to the processor, requiring extensive recoding.

Benevenuti et al. [24] deal with the redundancy of AXI Stream interfaces. Their solution includes using redundant bus inputs to every IP Core. Each IP Core can efficiently decide the correct one by knowing the kind of information it is receiving. The main problem of this approach is that it requires recoding of the IP Core to have triple internal redundancy.

3 Solution description

Since redundancy is an overall requirement, it must be addressed at the IP and system levels. Therefore, our solution is divided into two main elements: (a) `AXILiteRedundant` IP Core, and (b) Redundant system.

3.1 AXILiteRedundant

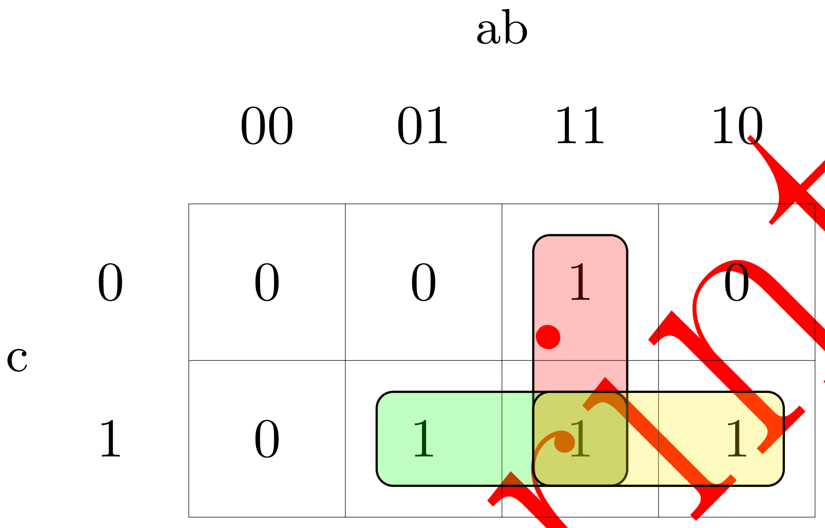
This IP Core is in charge of triplicating elements from the Slave IF to the Master IFs. It also decides the values of the signals that go from the multiple Masters IF to the Slave IF. To do so, it will vote for the correct value. Figure 1 shows the diagram of the IP Core. The Slave IF is the one connected (directly or indirectly) to the processor. The Master IF is the one connected to the external world. Since we are dealing with redundancy, there are three Master IF.

The core manages five channels per interface, as the AXI specification proposes. The following modifications to the AXI standard have been applied to implement the redundancy mechanism.

project IDI-20201264 and IDI-20220543, and through the Fondo Europeo de Desarrollo Regional 2014-2020 (FEDER funds).

- **Read Address**: This bus is triplicated to all the Master IF. Return ready signals are majority voted.
- **Read Data**: Ready signal is triplicated to all Masters. Data and valid are majority voted.
- **Write Address**: This bus is triplicated to all the Master IF. Return ready signals are majority voted.
- **Write Data**: This bus is triplicated to all Masters. Return ready signals are majority voted.
- **Write Response**: Ready signal is triplicated to all Masters. The response is majority voted.

The voting scheme uses a three-way majority voting [28]. This circuit outputs the most abundant value. In case of all three values are different, the circuit also outputs an error signal. The process of triplicating and voting is combinational. This ensures 0 latency in clock cycles. Since the process is highly optimized, it is also very efficient and adds a minor frequency penalty, as shown in the result section. Figure 2 shows the Karnaugh table of the circuit.



	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Figure 2: Tree-way majority voting Karnaugh table.

The IP core is also in charge of feeding some information to the processor. It will state if all the Masters are providing the same values. If not, and can be corrected (two Masters provide the same information), it will do so and inform of the error. If it cannot be corrected (errors in multiple bits of a bus but in different Masters), it will inform of the error, and the output value is irrelevant. This process is done independently for every channel in the interface.

3.2 System

The system is composed of several IP cores. In our example, apart from the minimal system required to implement a bare SoC device, we have added a firewall IP core. The main elements of the system are (see Fig. 3):

- `Zynq`: It is the processor present in the system [29]. In our example is in charge of accessing the peripherals. The processor has a single peripheral from the software point of view, even if they are triplicated.
- `AXILiteRedundant`: It is in charge of triplicating the incoming AXI transactions so that the processor sees a single peripheral.
- `Firewall`: A bridge between two portions of an AXI memory-mapped network protects one portion from issues caused by the opposite portion, such as protocol violations or timeout hangs [30]. In our case, it has been added as extra protection for a case where one of the cores has been compromised and is not working correctly. There are other ways of implementing this extra security, for example, using a `Firewall` per peripheral core.
- `Interconnect`: It is the block responsible for connecting several AXI networks. Even if only a single Slave is connected to the Zynq processor, this block is required because of the different natures of AXI on both sides. In this case, AXILite downstream and AXI 3 upstream [31].
- `GPIO`: They are the peripheral blocks [32]. In order to have redundancy, each IO is connected so that there is a physical majority vote [33].

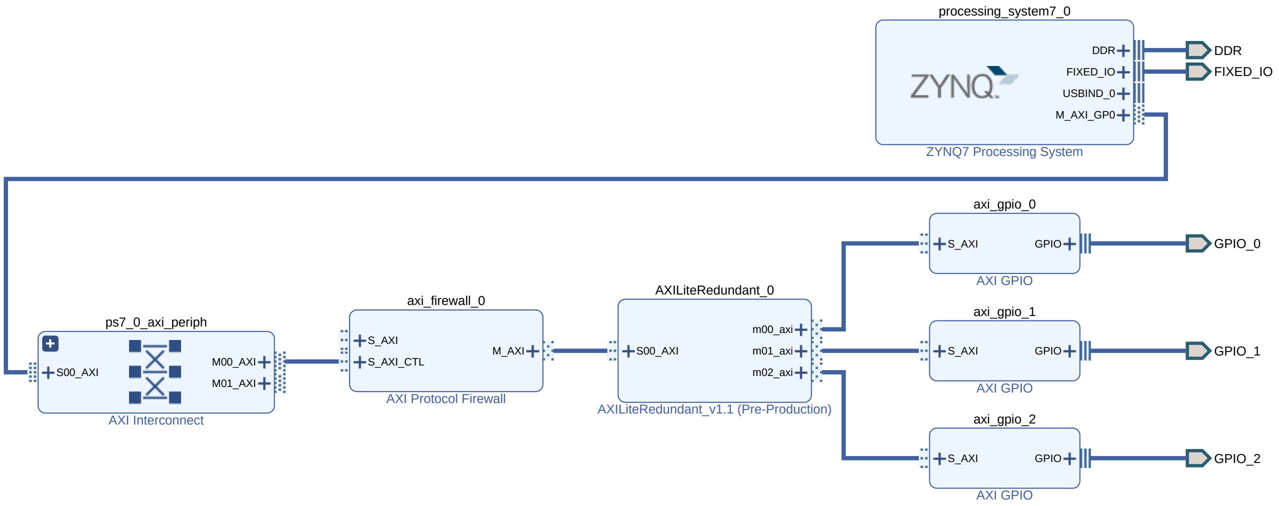


Figure 3: Overall system. The PS is connected to the redundant system through an AXI Interconnect and the proposed core.

4 Results

In order to verify the system, we used a simulation environment. This simplified system allows for verifying the correctness of the IP Core and can generate errors to test the resilience of the whole architecture. The simulation system is depicted in Fig. 4.

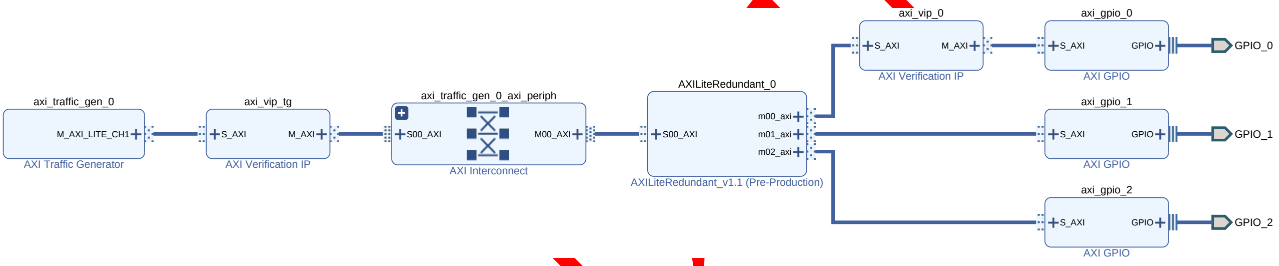


Figure 4: Simulation system. A traffic generator replaces the processor system. A couple of AXI verification cores are added to check for erroneous transactions. The AXI infrastructure is simplified compared to Fig. 3 to more easily check the correct functioning of the system.

The key elements, apart from those described in the previous section, are:

- `AXI Traffic Generator`: Generates traffic over the AXI4. It generates a wide variety of AXI transactions based on the core programming and selected mode of operation. [34]
- `AXI Verification IP`: Checks that all transactions comply with the AXI standard. It is useful when creating new IP to guarantee that it will not cause any issues on the communication channel [35]

The result of the simulation is depicted in Fig. 5. The IO has been simulated in reading and writing, including high impedance. The process that can be seen is:

- W_0: Configuration of the IO as output — write '0'.
- W_1: Write all '1' to the outputs.
- W_0: Configuration of the IO as input — write '1'.
- R_0: Read the input.

The read input process is done four times with different results:

1. All inputs have the same value, so there is no error.
2. GPIO 3 has a different value, so there is a recoverable error.
3. GPIO 1 has a different value, so there is a recoverable error.
4. All GPIO have different values, so there is an unrecoverable error.

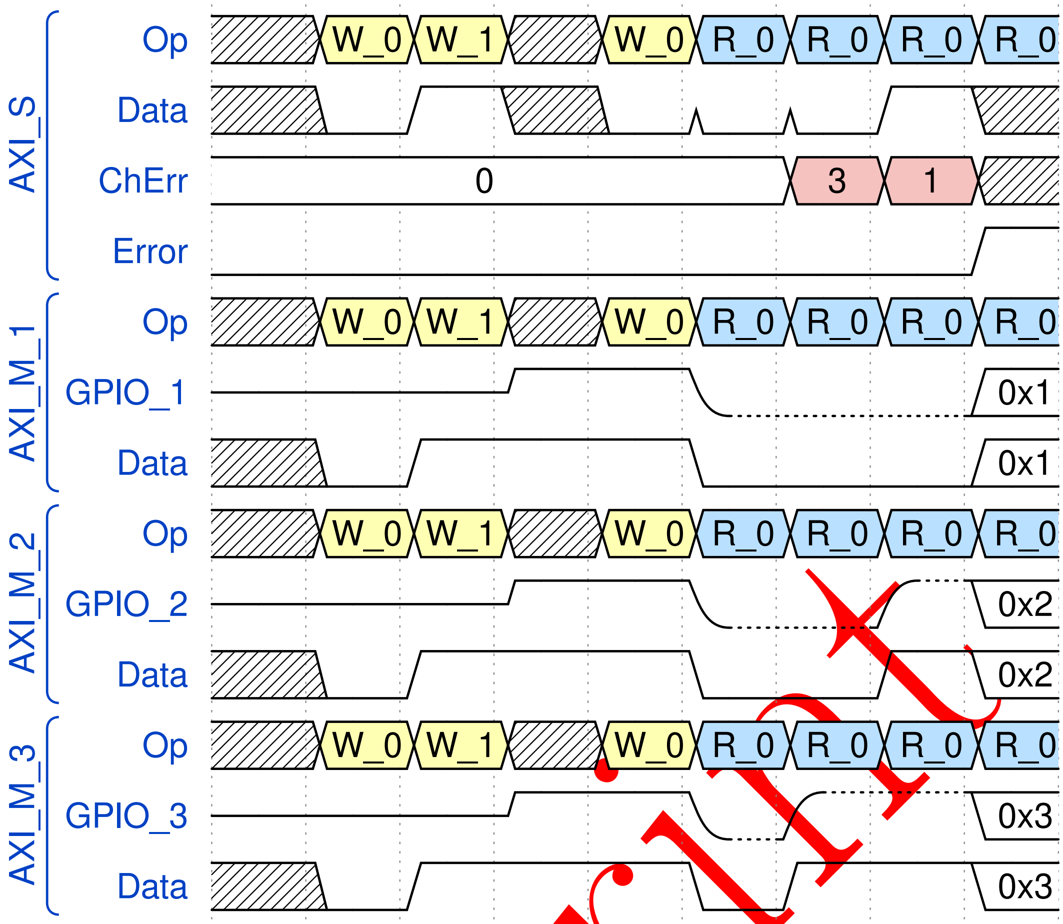


Figure 5: Simulation result. Initially, W_0 and W_1 configure the GPIO as output and write the value ‘1’. Next, the GPIO is configured as an input by writing into W_0, and four consecutive reads (R_0) are performed. In the first one, all three cores respond with the same value. Thus, there is no error. In the second read, the third GPIO is erroneous. In the third read, the first GPIO is wrong, while in the last one, all three values are different, leading to an unrecoverable error (Error = ‘1’).

In addition to increasing reliability, the IP design has focused on minimizing FPGA resources and energy consumption. Moreover, simultaneously, without introducing a significative penalty on timing results. Table 1 shows the area and energy results. As can be seen, the area required to implement AXILiteRedundant is 0.42 % of the LUTs present in the xc7z020 of the Zedboard. From the energy results, it is clear that energy consumption is low compared to the rest of the IPs in the system. For example, the IP core requires a fourth of the energy of the GPIO core.

Table 1: Resource utilization of different cores in the system. The proposed core requires less than 0.42 % of xc7z020 present in the Zedboard. The power results are those provided by Vivado post-implementation.

Name	Slice LUTs	Slice Registers	LUTs % of total	Power mW
axi_firewall_0	626	810	1.18	4.35
ps7_0_axi_periph	543	692	1.02	6.66
AXILiteRedundant	223	81	0.42	0.48
axi_gpio_2	131	382	0.25	1.65
axi_gpio_1	131	382	0.25	1.66
axi_gpio_0	131	382	0.25	1.74
axi_gpio_result	96	318	0.18	2.90
processing_system7_0	24	0	0.05	1530.38
rst_ps7_0_100M	19	40	0.04	0.23

From the timing point of view, as stated before, the IP is designed to introduce 0 clock latency. This simplifies the design and allows it to be used in real-time systems. From the frequency point of view, the overall system can work at speeds over 166 MHz. This shows that the overall performance is not affected by the IP core.

5 Comparison

Due to the specific nature of the IP, it is not easy to compare it with other IPs. Other approaches are specific to other interface definitions and, thus, difficult to compare with the proposed IP.

Benevenuti et al. [24] propose a redundant system for a single processing core using AXI Stream. The main difference is that the system has a single peripheral with a triplicated input interface and a single output interface. The input voter is included inside the IP core, thus requiring redesigning the IP. The input data comes from three different DMAs that transfer the information from the processor to the peripheral. The output is transferred to the DMAs that copy the data to the three different memory sections where the processor can get it. The processor must be aware of the redundancy to vote the results. The paper does not provide any area, power, or timing information.

Bertozzi et al. [23] propose a resilient IF not by redundancy but by including parity bits in the bus. This system is transparent for the processor and the peripheral. This approach does not use redundant peripherals and can only cope with errors in the bus and internal to the FPGA. The paper provides comprehensive power information depending on the redundancy algorithm (hamming, CRC,...). The proposed algorithms require very little power but may have high latency. The latency is (0.88–4.56) ns and is different for encoding and decoding. Area usage is $(2.7 \times 10^3–11.0 \times 10^3)$ gates. The energy consumption is (1.2–617) µW.

Table 2 shows the comparison between the proposed system and the other ones. The proposed system is the most efficient in terms of area and has the lowest latency. In terms of energy, it is better than the most power-hungry version provided by Bertozzi et al., although it requires more energy than the most basic ones presented in the article. Furthermore, it is the sole one that can handle both IF and peripheral errors. It also possesses critical characteristics, such as using unmodified IP cores and software.

Table 2: Comparison of the proposed paper with others present in the literature. The table shows the redundant elements, whether the CPU and the IP must be aware of the redundancy to function, and the impact on the area, energy, and timing.

Paper	Red.	CPU indep.	IP indep.	Area (gates)	Energy (mW)	Timing (ns)
[24]	IF	✗	✗	—	—	—
[23]	IF	✓	✓	20,826	0.62	4.85
This Paper	IF + IP	✓	✓	4376	0.48	2.50

6 Conclusions

This paper presents a TMR voter for the AXI4-Lite Standard. As FPGA systems are more and more usual in safe applications, at the same time that susceptible to SEUs, they must be hardened using redundancy. Our core protects the interconnection as well as the peripheral IPs. In addition to correcting single errors and detecting duplicated ones, the resulting IP requires very few FPGA resources for the implementation. The increment in power consumption is negligible, and it does not impact the latency due to the combinational nature of the IP core. This proposal is transparent both for the peripheral IPs and the CPU, allowing the usage of standard peripheral IPs and eliminating any overhead to the processing software. Further work in this area includes extending the IP range to other standard interfaces, such as AXI Stream. A redundant AXI Stream IP would be useful for systems that require high bandwidth and low latency for data processing and for multichip systems.

References

[1] George Lentaris, Ioannis Stratakos, Ioannis Stamoulias, Dimitrios Soudris, Manolis Lourakis, and Xenophon Zabulis. High-performance vision-based navigation on SoC FPGA for spacecraft proximity operations. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(4):1188–1202, apr 2020.

[2] Xunying Zhang and Xiaodong Zhao. Architecture design of distributed redundant flight control computer based on time-triggered buses for UAVs. *IEEE Sensors Journal*, pages 1–1, 2020.

[3] Qi Shao, Shunkun Yang, and Xiaodong Gou. Formal analysis of multiple-cell upset failure based on common cause failure theory. *IEEE Transactions on Reliability*, 70(4):1495–1509, dec 2021.

[4] Aaron Stoddard, Ammon Gruwell, Peter Zabriskie, and Michael J. Wirthlin. A hybrid approach to FPGA configuration scrubbing. *IEEE Transactions on Nuclear Science*, 64(1):497–503, jan 2017.

[5] Luis A. C. Benites, Fabio Benevenuti, Adria B. De Oliveira, Fernanda L. Kastensmidt, Nemitala Added, Vitor A. P. Aguiar, Nilberto H. Medina, and Marcilei A. Guazzelli. Reliability calculation with respect to functional failures

induced by radiation in TMR arm cortex-m0 soft-core embedded into SRAM-based FPGA. *IEEE Transactions on Nuclear Science*, 66(7):1433–1440, jul 2019.

[6] A. J. Sanchez-Clemente, L. Entrena, and M. Garcia-Valderas. Partial TMR in FPGAs using approximate logic circuits. *IEEE Transactions on Nuclear Science*, 63(4):2233–2240, aug 2016.

[7] Xin Wei, Yi Z Xie, Yu Xie, and He Chen. Dynamic partial reconfiguration scheme for fault-tolerant FFT processor based on FPGA. *J. eng.*, 2019(21):7424–7427, oct 2019.

[8] Xilinx. Triple module redundancy design techniques for virtex fpgas — xapp197. Application note, Xilinx, jul 2006.

[9] U. Kretzschmar, A. Astarloa, J. Lazaro, M. Garay, and J. Del Ser. Robustness of different TMR granularities in shared wishbone architectures on SRAM FPGA. In *2012 International Conference on Reconfigurable Computing and FPGAs*. IEEE, dec 2012.

[10] AMD-Xilinx Inc. Vivado Design Suite User Guide: Creating and Packaging Custom IP, 2022.

[11] System-on-Chip engineering S.L. SpaceWire IP, 2023.

[12] Felix Siegle, Tanya Vladimirova, Jørgen Ilstad, and Omar Emam. Mitigation of radiation effects in SRAM-based FPGAs for space applications. *ACM Computing Surveys*, 47(2):1–34, jan 2015.

[13] Marcos Santana Farias, Nadia Nedjah, and Paulo Victor R. de Carvalho. Active redundant hardware architecture for increased reliability in FPGA-based nuclear reactors critical systems. *Microprocessors and Microsystems*, 90:104495, apr 2022.

[14] Khurram Kazi, Eswin Anzueto, Timothy Henderson, Michael Kass, Wolf Johnson, Brianna Klingensmith, and Kurt Winikka. Design and validation architecture of the dream chaser® fault tolerant flight computer. In *2019 IEEE Space Computing Conference (SCC)*. IEEE, jul 2019.

[15] Cristiano Rodrigues, Ivo Marques, Sandro Pinto, Tiago Gomes, and Adriano Tavares. Towards a heterogeneous fault-tolerance architecture based on arm and RISC-v processors. In *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, oct 2019.

[16] Jiemin Li, Shancong Zhang, and Chong Bao. DuckCore: A fault-tolerant processor core architecture based on the RISC-v ISA. *Electronics*, 11(1):122, dec 2021.

[17] Haomiao Su, Tiejun Lu, Changlei Feng, and Lei Chen. Triple module redundancy reliability framework design based on heterogeneous multi-core processor. *Procedia Computer Science*, 183:504–511, 2021.

[18] Sebastian Hiergeist and Georg Seifert. Implementation of a SPI based redundancy network for SoC based UAV FCCs and achieving synchronization. In *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*. IEEE, sep 2018.

[19] Armando Astarloa, Jesus Lazaro, Unai Bidarte, Aitzol Zuloaga, and Mikel Idirin. System-on-chip implementation of reliable ethernet networks nodes. In *IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, nov 2013.

[20] William R. Tonti, Jack A. Mandelman, Anthony R. Bonaccio, Claude L. Bertin, Howard L. Kalter, and John A. Fifield. Redundant input/output driver circuit, 1 2001.

[21] Yong Li, Jingyan Xue, and Qilin Gai. Design for input and output card of triple redundant control system. In *2014 IEEE Symposium on Computer Applications and Communications*. IEEE, jul 2014.

[22] Armando Astarloa, Jesús Lázaro, Unai Bidarte, Aitzol Zuloaga, and José Luis Martín. An autonomous fault tolerant system for CAN communications. In *Trends in Applied Intelligent Systems*, pages 281–290. Springer Berlin Heidelberg, 2010.

[23] D. Bertozzi, L. Benini, and G. De Micheli. Error control schemes for on-chip communication links: the energy-reliability tradeoff. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24(6):818–831, jun 2005.

[24] Fabio Benevenuti and Fernanda Lima Kastensmidt. Analyzing AXI streaming interface for hardware acceleration in AP-SoC under soft errors. In *Applied Reconfigurable Computing. Architectures, Tools, and Applications*, pages 243–254. Springer International Publishing, 2018.

[25] Laurent Gantel, Quentin Berthet, Emna Amri, Alexandre Karlov, and Andres Upegui. Fault-tolerant FPGA-based nanosatellite balancing high-performance and safety for cryptography application. *Electronics*, 10(17):2148, sep 2021.

[26] C. De Sio, S. Azimi, and L. Sterpone. On the analysis of radiation-induced failures in the AXI interconnect module. *Microelectronics Reliability*, 114:113733, nov 2020.

[27] Josef Borcsok, Waldemar Muller, Eike Hahn, Michael Schwarz, and Mohamed Abdelawwad. Safe-system-on-chip for functional safety. In *2021 18th International Multi-Conference on Systems, Signals & Devices (SSD)*. IEEE, mar 2021.

[28] B. Parhami. Voting algorithms. *IEEE Transactions on Reliability*, 43(4):617–629, 1994.

[29] Xilinx. Processing system 7 — pg082. Product guide, Xilinx, may 2017.

[30] Xilinx. Axi protocol firewall ip — pg293. Product guide, Xilinx, feb 2022.

[31] Xilinx. Axi interconnect — pg059. Product guide, Xilinx, may 2022.

[32] Xilinx. Axi gpio — pg144. Product guide, Xilinx, oct 2016.

[33] Atin Mukherjee and Anindya Sundar Dhar. Triple transistor based triple modular redundancy with embedded voter circuit. *Microelectronics Journal*, 87:101–109, may 2019.

[34] Xilinx. Axi traffic generator — pg125. Product guide, Xilinx, feb 2019.

[35] Xilinx. Axi verification ip — pg267. Product guide, Xilinx, dec 2021.