On the Analysis of Radiation-induced Failures in the AXI Interconnect Module

On the Analysis of radiation-induced Failures in the AXI Interconnect Module

C. De Sio, S. Azimi, L. Sterpone*

Dipartimento di Automatica e Informatica, Politecnico di Torino, Torino, Italy

Abstract

Due to the increasing demand for high performance in embedded systems, devices such as SRAM-based programmable devices are becoming an appealing solution to reach high performance with limited costs. However, SRAM-based programmable devices are subjected to various sources of radiation-induced faults that affect their reliability, such as ionizing radiation and particles, even at sea-level. In this paper, we evaluate the reliability of the interconnection module, implemented on the programmable hardware, against radiation-induced faults in the configuration layer. To do so, we performed a fault injection campaign in order to emulate the radiation-induced effects impacting the configuration memory of AP-SoC Zynq 7000, specifically targeting the configuration memory section programming the interconnection module implemented on the programmable logic. This interconnection module is a crucial element for a wide range of applications and mitigation techniques such as hardware-accelerated designs, Dynamic Partial Reconfiguration, or Triple Modular Redundancy, especially if they aim to meet high performance and high bandwidth. The fault injection results have been analyzed and classified accordingly with the effect observed on the processor-system side in terms of availability and fault model affecting data computed by cores implemented on the programmable logic side.

1. Introduction

Nowadays, All-Programmable System-on-Chips (AP-Soc) have become interesting to a wide range of applications. Due to the integration on a single chip of a microprocessor and programmable hardware, they are a fulfilling choice for meeting the requirements of high-performance systems. This feature along with high flexibility, low costs, and low power consumption made these devices attractive also in fields that require high reliability such as avionics, aerospace, and automotive [1][2]. The architecture of AP-SoC allows the designer to exploit the programmable logic moving computationally demanding tasks to be executed by customizable hardware. To support a high-bandwidth on-chip communication, the AP-SoC typically offers the Advanced Microcontroller Bus Architecture (AMBA) as the major communication standard since it is supporting high-performance and high-frequency communications [3]. Moreover, the vendor provides a module, directly implementable on the AP-SoC hardware, for enabling high-performance communication between programmable logic and processor system through the AMBA. The typical configurations for high-performance architectures rely on this module to manage the communication between the master, usually the processor system, and the slaves (e.g, custom and IPs cores). The Advanced eXtensible Interface 4 (AXI-4) communication interface introduced in AMBA by ARM is adopted by Xilinx as the standard interface for cores. It allows designers to easily combinate applications involving both software running on the processor system and cores implemented on programmable logic through the high-performing communication infrastructure [4][5].

However, the advantages offered by the AP-SoCs architecture come along with ionized particles as a dominant source of error for the programmable logic of the AP-SoCs [6][7][8].

In particular, Single Event Upsets (SEUs) are soft errors that are caused by a single ionizing particle striking a sensitive node of the device. As a result of the interaction between particles and silicon, the particles release their energy which can lead to a change of the state of a logic element or memory cell. Considering the recent technology scaling, SEUs are not a concern for applications and systems reliability only regarding the avionic field but, due to the secondary particles at the sea level, they are becoming crucial also for ground-level applications such as automotive [9][10]. SEUs occurring in the configuration memory (CRAM) of a hardware-programmable device can cause a bitflip in the memory cells. The content of the CRAM cells programs the basic elements of programmable logic such as LUTs, Flip-Flops, DSP, etc. Hence, if the corrupted memory cell of the configuration layer was configuring a critical resource of the implemented design, the fault will affect the application until the configuration memory is rewritten with the correct content, causing a single-event functional interrupt (SEFI).

This work aims to analyze the impact of radiation-induced SEUs affecting the AXI Interconnect module. The analysis is performed by injecting SEUs in the CRAM of a Zynq-7000 AP-SoC programmable logic targeting only the specific module under the test and the unused resources in its proximity. The experimental analysis showed various effects

* Corresponding author. luca.sterpone@polito.it

touching the communication architectures such as the unavailability and erroneous computation of the modules connected via the AXI Interconnect module. Induced errors on data have been classified accordingly with their behavior (i.e., stuck-at faults) and classified according to multiplicity and type.

This paper is divided into seven sections. In Section 2, related works are elaborated. Section 3 provides background on CRAM and SEUs. Section 4 describes the reliability evaluation approach and the fault injection environment. Section 5 is dedicated to the experimental analysis description, while section 6 reports the obtained results. In Section 7, the conclusions and future work are discussed.

2. Related Works

Several works have been dedicated to the effects of SEUs affecting configuration memory of programmable devices [11][12][13]. However, only a few works regard the AXI Interconnect module used for communication between the processor system and the programmable logic. In [14][15], the authors evaluate the reliability of various AXI interfaces connected to the AXI Interconnect core. They perform a reliability evaluation for different modules by accumulating faults in the configuration memory using fault injection. In particular, the AXI Interconnect has been tested along with the AXI-DMA IP Cores as a single module. The analysis reported that it is a weak link for the design even with the hardening of the AXI DMA module. However, the characteristics of errors and experienced misbehaviors have not been evaluated.

3. Background

AP-SoCs are integrated circuits characterized by a processor system and programmable logic. The programmable logic can be configured by the user to implement custom hardware designs. The configuration of the programmable logic is achieved by downloading the bitstream in an SRAM memory, called configuration memory (CRAM). The bitstream is a stream of data containing the configuration information for the basic programmable elements that compose the reconfigurable logic, such as point-to-point connections, look-up-tables, and flip-flops. The configuration of the device is kept in the configuration memory until the device is reconfigured or power cycled.

Single Event Upset (SEU) is one of the dominant sources of errors for this type of device. SEUs are caused by the interaction between ionized particles and silicon both at sea-level (atmospheric neutrons) and in space (cosmic rays, ionizing radiation) [16][17]. As a result of this interaction, the content of a memory cell can be flipped. If the memory cell affected by the phenomenon belongs to the CRAM, the faulty bit will affect the configuration of the device until the next reconfiguration or power cycle. If this memory cell is a cell programming a resource used by the design deployed on the device, the SEU can introduce an undesired modification in the netlist actually implemented in the programmable logic. Typically, only a subset of the resources available on the programmable logic is used by the design implemented on the device. Therefore, only an unknown subset of the configuration memory content, depending on the implemented netlist and the specific device, will generate errors and misbehaviors when corrupted.

On the programmable logic side, the communication of the modules implemented on hardware with the processor system is typically demanded to the AXI Interconnect module, provided by the vendor for the developers [5]. The AXI Interconnect module is intended to be part of the design implemented in the programmable logic in order to provide a bridge between modules supporting AXI Interface and the processor system through the AXI ports it is provided of. Therefore, the AXI-Interconnect module is a critical element for solutions

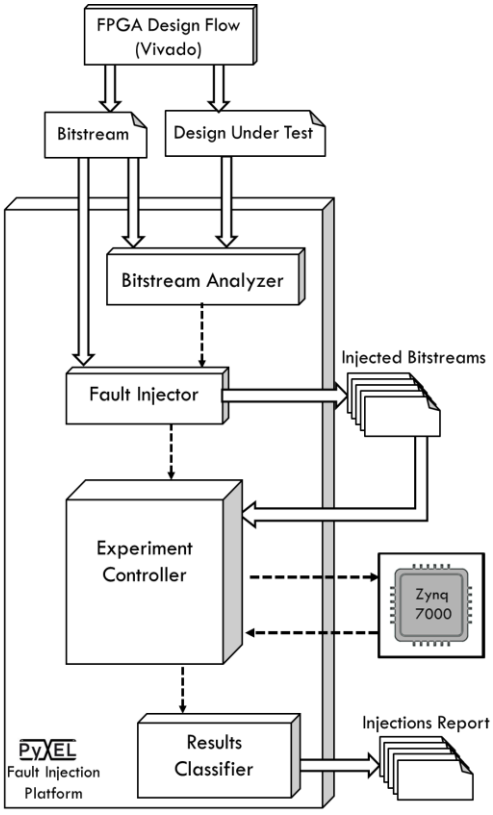

Fig. 1. Schema of the reliability evaluation environment.

relying on the programmable hardware for enhancing their performance.

4. The Reliability Evaluation Environment

A reliability evaluation platform consisting of a fault injection framework and a test environment is proposed for carrying out the experimental analysis. The fault injection platform, presented in [18], has been enhanced to support fault injection on the programmable logic of the Zynq-7000 device [19]. The structure of the reliability evaluation environment is represented in Figure 1.

Starting from the bitstream and the implemented netlist provided as an output by the standard FPGA design flow, we used the PyXEL [18] tool to analyze the bitstream structure for detecting the subset of the whole bitstream related to the AXI Interconnect module implemented within the design under test. The goal of this step is to constrain the injection space to a subset of the whole space. This allows us to inject only faults affecting the module under test. The chosen subset of bits regards only the bits related to the hardware resources implementing the module under test and the unused resources near it. Indeed, the unused resources in the proximity of the module under test could create conflicts with the module itself if they are activated by the injection. Using the info obtained by the bitstream analyzer offered by PyXEL, the fault injector is able to generate a set of faulty bitstreams with specific characteristics, such as the subspace of the configuration data where to inject and the kind of fault to inject. The fault injector can be instrumented to execute the injections in the configuration memory accordingly with the desired fault model, such as single or multiple bits corruption and desired bit transition (set, reset, or flip). The set of faulty bitstreams generated by the fault injector module is used by the experiment controller to configure the programmable logic of Zynq-7000. The experiment controller is able to automatically perform error detection steps to evaluate if injected faults eventually generate errors in the design. In particular, the controller configures the programmable logic, triggers the software for detecting the errors in the design under test, and produces a report in terms of availability and correctness the data obtained from the computations units implemented on the hardware and connected to the processor system by means of the module under test.

5. Experimental Analysis

Using the reliability evaluation environment described in Section 4, a fault injection campaign has been carried out to analyze the errors induced by SEUs on the AXI Interconnect module. The analysis has been executed on a Zynq-7000 AP-SoC.

The injection of faults has been constrained to a subset of 338,446 bit out of the 32,345,856 bits configuring the whole configuration memory. Using the PyXEL tool, this subset has been identified to include the resources used by the AXI Interconnect Module and the unused resources in its proximity. A subsection of the configuration memory, including both AXI Module and unused nearby resources, has been chosen. Indeed, the unused resources can be activated as results of the configuration memory corruption leading to errors.

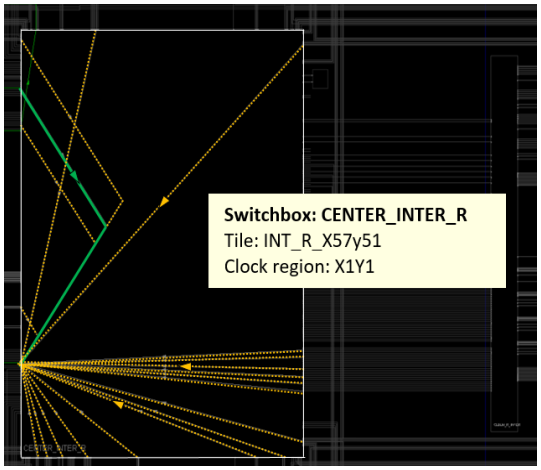

Fig. 2. Conflicting resources of an interconnection box

To better clarify the interaction, Figure 2 shows an interconnection box in the implemented design under evaluation as presented by the Vivado device view. The interconnection box, also named switch matrix, has an enabled point-to-point connection highlighted with a continuous green line. Additionally, there are several disabled interconnections, highlighted by dashed yellow lines, that if activated, can cause a conflicting undesired shortcut between the original connection endpoint and other sources. Many of the conflicted connections represented in Figure 2 can be activated as the effect of a single corrupted bit [16]. However, not all the conflicting interconnections will generate errors on the output due to electrical and logical masking.

We generated 10,000 faulty bitstreams through the corruption of a single bit in each bitstream. Corrupted bits have been chosen randomly among the bits composing the subset, allowing both 0 to 1 and 1 to 0 transitions. The design under test consists of an AXI Interconnect IP Core shared between several instances of an HLS-implemented IP Core. Figure 3 shows a schema of the architecture of the system implemented on the AP-SoC. In particular, the processor system,

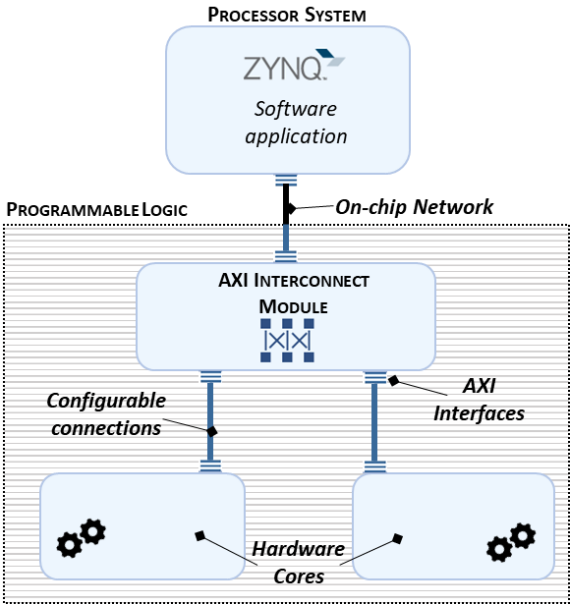

Fig. 3. Architectural Schema of the design under evaluation.

the AXI Interconnect Module, and the IP Cores connected to the processor system through the AXI Interconnect module are shown. The AXI Interconnect module is implemented on the programmable logic and configured as a single master with multiple slaves. The processor system acts as a master while the hardware cores act as slaves. The hardware cores compute a 32-bit deterministic non-linear signature from four 32-bit parameters. When demanded by the experiment controller, an error detection software runs on the processor system. The error detection software test the availability of hardware cores to check if the AXI Interconnect module is working properly. If the hardware cores are available for the computation test, a test vector is sent through the AXI Interconnect module and the results are collected by the testing software and forwarded to the fault injection platform. The collected results are compared with the expected results by the fault injection platform to detect the errors.

Due to the intrinsic characteristics of AP-SoC programmable logic, not all the faults injected in the configuration memory will cause an error on the output. Indeed, some injected faults will activate or modify unused resources and not all the unused resources will cause conflicts with the implemented designs. Additionally, some errors could be masked by the circuit or electrical behavior of the device not reaching the output. The results analyzer of the fault injection platform is able to detect stuck-at bits in the results received through the AXI Interconnect, and distinguish induced faults in the system basing on the availability of the salves and the correctness of their computations.

6. Results

The errors detected have been classified into four categories, accordingly with the effects they generated. The detected categories are *unavailability*, *silent data corruption*, *detectable data corruption*, and *unavailability and data corruption*. *Unavailability* error is caused by a total failure of the communication core which is not able to route data and commands from master to the slaves or vice versa due to injected faults. *Detectable data corruption* occurs when the cores implementing the same computation and stimulated with the same input vector logic generate different outputs. In this case, the master is able to detect misbehaviors without knowing the golden results. *silent data corruption* occurs when the outputs of the modules subjected to the same input return the same faulty output, making it detectable only through comparison with the golden results. *Unavailability and data corruption* include a mix of unreachable and faulty cores. There has not been any occurrence of a mix of unreachable and correctly working cores.

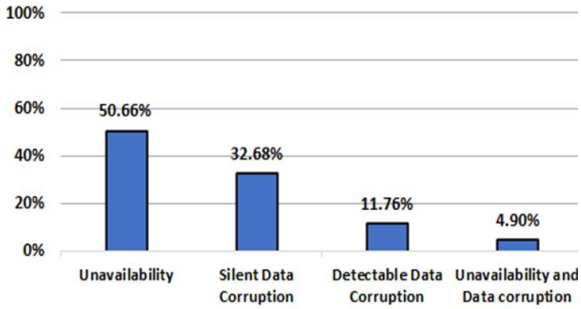

Fig. 4. Classification of detected errors.

The 3.06% of the injected bitstreams have generated misbehaviors, while the rest of them did not cause errors on the output of the system. According to [20], the chosen amount of injection assures to estimate with 95% confidence level and a margin of error of 0.01 the probability that a single bit corrupted, among the memory section under test could lead to an error on the output of the application.

Among the detected errors generated by the evaluated faulty configurations, 50.66% provoked the *unavailability* of the IP Cores in the programmable logic. An additional 4.90% resulted in partial unavailability, where one IP Core cannot be reached by the master and the communication while the other IP Core is affected by errors (*unavailability and data corruption*). The rest of the error-inducing configurations had errors in the data transferred

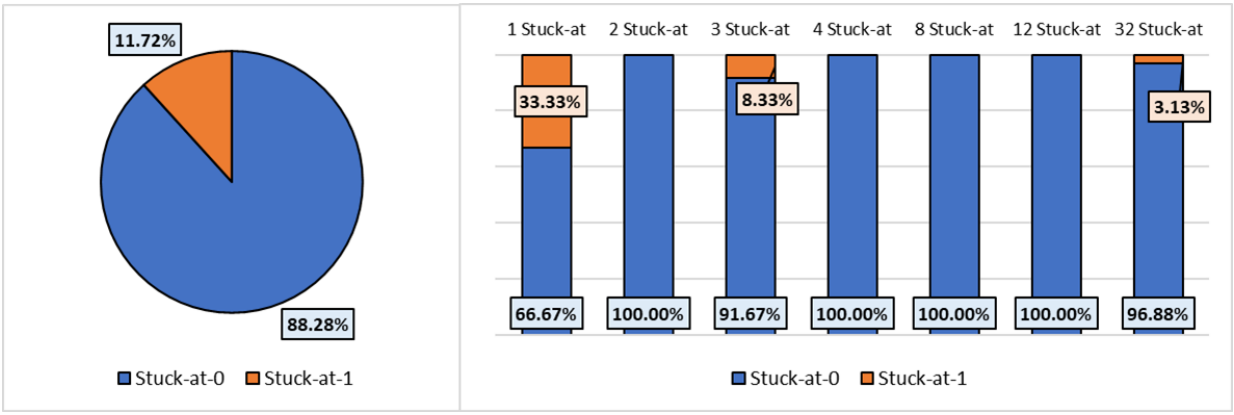

Fig. 6. Distributions of stuck-at-0 and stuck-at-1 faults affecting data words

between the master and the slaves. In particular, only 11.76% have generated *detectable data corruption*, while 32.68% produced *silent data corruption*. Figure 4 summarizes the detected kind of faults and their percentages.

Further analysis has been carried on categories reporting errors on transferred data. In particular, 78.80% of the errors introduced by communication core over exchanged data have been identified to be stuck-at faults. The stuck-at faults always affect the communication with both the hardware cores and specifically the same bits of the data words for both the cores. We would like to emphasize that stuck-at faults introduced by the AXI Interconnect Module will lead to silent data corruption even with duplication of the hardware cores if the cores are connected to the master system through the same AXI Interconnect, as typically happens.

The multiplicity of stuck-at bits affecting erroneous data are reported in Figure 5. In particular, it shows more than 75% of faulty words present only one faulty bit out of 32 composing the data words. Additionally, the stuck-at faults have been also classified by type (i.e. stuck-at-0 or stuck-at-1). From the whole set of detected stuck-at, only 11.72% were

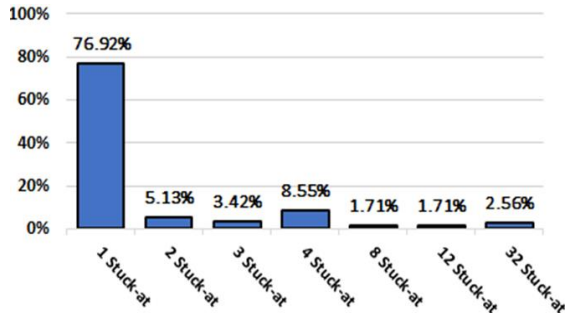

Fig. 5. Stuck-at multiplicity distribution over data words

stuck-at-1. It is reported in Figure 6 along with the distribution of stuck-at-0 and stack-at-1 for each stuck-at multiplicity. In particular, stuck-at-1 faults have been observed mostly on words with a single faulty bit, with few cases in 3-stuck-at and 32-stuck-at categories. In detail, 33.33% of the stuck-at detected in the results experiencing only a single stuck-at on the whole bits composing the word was stuck-at-1.

7. Conclusions and Future Works

In this paper, we analyzed the impact of SEUs on the AXI Interconnect module. We emulated the single event upset in the bitstreams configuring the programmable logic of a Zynq-7000 AP-SoC device using a reliability evaluation environment. The fault injection campaign has been carried out targeting the resources related to the module under test. The errors generated in the system as effects of the injected faults have been classified both in terms of data transfer errors and availability of the cores connected via the injected module to the processor system. The errors affecting the data have been furtherly analyzed, identifying the stuck-at fault model. Detected stuck-at faults have been categorized accordingly with their multiplicity and type.

As future works, a reliability analysis against multiple-bit upsets affecting the CRAM of the device is of interest to verify if it leads to different kinds of faults, especially regarding the type and multiplicity of generated stuck-at faults. Additionally, we plan to analyze the type of errors that can be mitigated by a hardened version of the AXI Interconnect module a further investigation is planned to verify if the hardening increases the cases where at least one of the two module function correctly that is currently only 11,76% of the cases.

References

[1] S. Sabogal, A. George and G. Crum, "ReCoN: A Reconfigurable CNN Acceleration Framework for Hybrid Semantic Segmentation on Hybrid SoCs for Space Applications," *2019 IEEE Space Computing Conference (SCC)*, Pasadena, CA, USA, 2019, pp. 41-52, doi: 10.1109/SpaceComp.2019.00010.

[2] Y. Han and E. Oruklu, "Real-time traffic sign recognition based on Zynq FPGA and ARM SoCs," *IEEE International Conference on Electro/Information Technology*, Milwaukee, WI, 2014, pp. 373-376, doi: 10.1109/EIT.2014.6871793.

[3] ARM Limited, "AMBA AXI and ACE Protocol Specification", February 2013, Issue E, ARM IHI 0022E.

[4] Xilinx Inc., "AXI Reference Guide", San Jose, CA, USA, July 2017, User Guide, UG1037.

[5] Xilinx Inc., "AXI Interconnect v2.1: LogiCORE IP Product Guide", San Jose, CA, USA, December 2017, Product Guide, PG059.

[6] C. De Sio, S. Azimi, L. Bozzoli, B. Du, and L. Sterpone, "Radiation-induced Single Event Transient effects during the reconfiguration process of SRAM-based FPGAs", in *Microelectronics Reliability*, Volumes 100–101, 2019, 113342, ISSN 0026-2714.

[7] C. De Sio, S. Azimi, L. Sterpone and B. Du, "Analyzing Radiation-Induced Transient Errors on SRAM-Based FPGAs by Propagation of Broadening Effect," in *IEEE Access*, vol. 7, pp. 140182-140189, 2019, doi: 10.1109/ACCESS.2019.2915136.

[8] S. Azimi, L. Sterpone, B. Du, and L. Boragno, "On the analysis of radiation-induced Single Event Transients on SRAM-based FPGAs", in *Microelectronics Reliability*, Volumes 88–90, 2018, Pages 936-940, ISSN 0026-2714, https://doi.org/10.1016/j.microrel.2018.07.135.

[9] R. C. Baumann, "Landmarks in Terrestrial Single-Event Effects, " Part B of the Short Course presented at the 2013 *Nuclear and Space Radiation Effects Conference*, San Francisco, CA, 8 July 201

[10] G. Hubert, L. Artola, D. Regis, "Impact of scaling on the soft error sensitivity of bulk, FDSOI and FinFET technologies due to atmospheric radiation", in *Integration*, Volume 50, 2015, Pages 39-47, ISSN 0167-9260, https://doi.org/10.1016/j.vlsi.2015.01.003.

[11] B. Du, S. Azimi, C. de Sio, L. Bozzoli and L. Sterpone, "On the Reliability of Convolutional Neural Network Implementation on SRAM-based FPGA," *2019 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, Noordwijk, Netherlands, 2019, pp. 1-6, doi: 10.1109/DFT.2019.8875362.

[12] M. Ceschia *et al*., "Identification and classification of single-event upsets in the configuration memory of SRAM-based FPGAs," in *IEEE Transactions on Nuclear Science*, vol. 50, no. 6, pp. 2088-2094, Dec. 2003, doi: 10.1109/TNS.2003.821411.

[13] T. Li, H. Yang, H. Zhao, N. Wang, Y. Wei and Y. Jia, "Investigation into SEU Effects and Hardening Strategies in SRAM Based FPGA," *2017 17th European Conference on Radiation and Its Effects on Components and Systems (RADECS)*, Geneva, Switzerland, 2017, pp. 1-5, doi: 10.1109/RADECS.2017.8696177.

[14] F. Benevenuti and F. L. Kastensmidt, "Reliability evaluation on interfacing with AXI and AXI-S on Xilinx Zynq-7000 AP-SoC," *2018 IEEE 19th Latin-American Test Symposium (LATS)*, Sao Paulo, 2018, pp. 1-6, doi: 10.1109/LATW.2018.8347233.

[15] F. Benevenuti and F. L. Kastensmidt, "Analyzing AXI Streaming Interface for Hardware Acceleration in AP-SoC Under Soft Errors", Applied Reconfigurable Computing. Architectures, Tools, and Applications (2018), 10.1007/978-3-319-78890-6_20.

[16] E. Normand, "Single event upset at ground level," in *IEEE Transactions on Nuclear Science*, vol. 43, no. 6, pp. 2742-2750, Dec. 1996, doi: 10.1109/23.556861.

[17] H. Quinn, "Radiation effects in reconfigurable FPGAs", in *Semiconductor Science and Technology*, Volume 32, pp 044001, April, 2017. 10.1088/1361-6641/aa57f6.

[18] L. Bozzoli, C. De Sio, L. Sterpone and C. Bernardeschi, "PyXEL: An Integrated Environment for the Analysis of Fault Effects in SRAM-Based FPGA Routing," *2018 International Symposium on Rapid System Prototyping (RSP)*, Torino, Italy, 2018, pp. 70-75, doi: 10.1109/RSP.2018.8632000.

[19] Xilinx Inc., "Zynq-7000 All Programmable SoC: Technical reference manual", San Jose, CA, USA, July 2018, User Guide, UG585.

[20] R. Leveugle, A. Calvez, P. Maistri and P. Vanhauwaert, "Statistical fault injection: Quantified error and confidence," *2009 Design, Automation & Test in Europe Conference & Exhibition*, Nice, 2009, pp. 502-506, doi: 10.1109/DATE.2009.5090716.