Non-intrusive fault tolerance in soft processors through circuit duplication

Frederico Ferlini, Felipe A. da Silva, Eduardo A. Bezerra, and Djones V. Lettnin
Federal University of Santa Catarina
Department of Electrical Engineering
Florianópolis – SC, Brazil
E-mail: {fferlini, felipes, Eduardo.Bezerra, lettnin}@eel.ufsc.br

Abstract—**The flexibility introduced by Commercial-Off-The-Shelf (COTS) SRAM based FPGAs in on-board system designs make them an attractive option for military and aerospace applications. However, the advances towards the nanometer technology come together with a higher vulnerability of integrated circuits to radiation perturbations. In mission critical applications it is important to improve the reliability of applications by using fault-tolerance techniques. In this work, a non-intrusive fault tolerance technique has been developed. The proposed technique targets soft processors (e.g. LEON3), and its detection mechanism uses a Bus Monitor to compare output data of a main soft-processor with its redundant module. In case of a mismatch, an error signal is activated, triggering the proposed fault tolerance strategy. This approach shows to be more efficient than the state-of-the-art Triple Modular Redundancy (TMR) and Software Implemented Hardware Fault Tolerance (SIHFT) approaches in order to detect and to correct faults on the fly with low area overhead and with no major performance penalties. The chosen case study is an under development On-Board Computer (OBC) system, conceived to be employed in future missions of the Brazilian Institute of Space Research (INPE).**

Keywords: On-board computer; fault tolerance; fault injection; LEON3; bus monitor; single event upset; space applications.

I. INTRODUCTION

In recent years, the usage of Commercial-Off-The-Shelf (COTS) SRAM based FPGAs in military and aerospace applications have been growing fast. When comparing to the traditional ASIC development process, FPGA technology offers lower development costs, and it is a more interesting solution for small production volumes. Current FPGA devices allow the integration of complete digital systems in a single chip, offering up to two million logic cells, 65 MBytes of embedded memory, and a number of additional features such as high performance arithmetic functions and high speed input/output modules [1]. Moreover, by employing FPGAs the designers can concentrate all their efforts in the application development without worrying with the complex fabrication process of ASIC devices.

The advantages of a COTS SRAM based FPGA are attractive for mission-critical applications like those in the space market. Usually, a satellite is an exclusive product made for a single customer, and as a result the costs and time necessary for developing new ASICs for each new satellite are often not justified. These features make FPGAs an interesting option for the space market.

Despite all the advantages of COTS SRAM based FPGA, they are not widely used for space applications due to their sensitivity to ionizing radiation. In the harsh space environment electronic devices are exposed to different sources of ionizing radiation that can interact with it causing faults and damages. In SRAM FPGAs the main effects caused by ionizing radiation are the Single Event Effects (SEE). SEEs take place when a single particle strikes a certain location within the device. Depending on the strike location and time, the electric fields, and the energy of the incident particle, this effect can produce different functional behaviors. SEEs are usually transient faults (aka. soft errors) that affect the device for a certain period of time, at most until a power cycle is performed. One example of soft error is the Single Event Upset (SEU). Also called upset or bit-flip, the SEU is the effect of a particle that changes the value of a memory element. SEUs are considered soft errors because at the first writing operation of the affected memory element the wrong value will be overwritten. In SRAM FPGAs, memory elements are used to store the configuration of the user application loaded to the FPGA [1][2]. Configuration Memory (CM) cells are the basic blocks of FPGA technology. The CMs can be used in the routing matrix to interconnect logic blocks or to configure logic blocks. When an ionizing particle strikes a CM, causing an upset, it can result in a functional failure changing the state in a logic block or can result in a misrouted or missing signal when the CM is storing a routing matrix [3].

Considering the sensitivity of COTS SRAM based FPGAs to radiation effects, it is necessary to apply mitigation techniques in order to use this kind of device in space applications. In this paper we describe a non-intrusive fault tolerance technique applied to an embedded soft-processor COTS SRAM based FPGA. The technique proposed in this paper aims the reduction of the overhead traditionally associated to other approaches as the Triple Modular Redundancy (TMR). In addition, the proposed technique has the capability of identify run-time errors, thus helping to avoid a fault in the OBC processor to be propagated as an error in the whole system. The proposed technique is validated via fault injection strategies. Meanwhile, a simulation environment has been developed aiming the simulation of the soft processor with SEUs applied in all its signals.

978-1-4673-2356-7/12$26.00 © 2012 IEEE

The remaining of this paper is organized as follows. Section III introduces the case study, which is an On-Board Computer (OBC) for satellites, implemented using FPGAS. The fault tolerance technique is described in Section IV. A fault injection emulator is under development, and it is described in Section V. In Section VI the simulation environment is described together with the experimental results.

II. RELATED WORK

The idea of identifying run-time errors in COTS FPGAs is investigated in [4][5][6]. In these papers the authors describe an on-line checker to identify errors in different applications embedded in FPGAs. The on-line checker is based on the properties of the application running on the FPGA. To identify an error the on-line checker verifies if any of the applications properties is disrespected. More complex applications have a larger number of properties to be verified, what makes the on-line checker more complex implying in a big area overhead.

In [7] a hardware control flow checker for embedded processors was introduced. An 8051 IP core embedded in an FPGA is used as case study to identify errors caused by bit flips in the processor control flow registers. The errors are identified by an on-line checker who analyze the internal processor registers and generates a signature based on their state. Then, this signature is compared with a pre calculated set of possible signatures for the running software. Because of that, this approach depends on the software, because it is necessary to calculate the set of possible signatures and store this set in the on-line checker internal memory.

The concept of control flow through bus observability is used in [8]. In this hybrid approach, fault detection oriented features are implemented in the software execution, these features allow the processor to communicate with an external circuit through the SoC bus. By computing the received information, this circuit determines incorrect executions. However, the addition of instructions to the execution software causes loss of performance, as the processor has to execute these instructions.

Our main contribution in this research work is to use the concepts of control flow with bus observability to identify run-time errors with no modifications in the application (software), with no software dependence and with low area overhead.

III. ON-BOARD COMPUTER ARCHITECTURE

The OBC is the central unit of a satellite control and it is responsible for all data processing, communication and decision-making activities. In other words, the OBC is the "manager" of a satellite, as all control modules are directly connected to the OBC. A single failure on the OBC can cause permanent damage to the satellite, with possible lost of its orbit, loss of communication with the Earth, among others.

One of OBC's most important tasks is the *attitude* control of the satellite. In a satellite, attitude means the orientation of the satellite in space. The OBC is connected to sensors that provide information about the satellite orientation. Based on this information the OBC can estimate the orientation of the satellite and, if necessary, it can send commands to the actuators in order to change its orientation. Common examples of sensors used in satellites are: star sensors, solar sensors, inertial sensors, magnetometers and GPS. The most used actuators on a satellite are: thrusters and reaction wheels.

Another important component of the OBC is the communication system. The communication system of a satellite is the only way of interaction with a Ground Station. That is the only option for the mission control to have information about the health of the satellite and to send commands or requisitions to be executed by the OBC. The commands sent by the Ground Station to the satellite are called Telecommands (TC) and they can contain instructions to the OBC as, for instance, to turn off some instrument, to change the satellite attitude, among others. Every time the OBC receives a TC it should send a feedback message with the information required by the TC or just to inform the Ground Station that the TC was received without errors, this feedback message is called Telemetry (TM).

Figure 1. shows a generic satellite control unit. The OBC is located in the center of the figure, and it is surrounded by sensors, actuators and other satellite subsystems. At the top of the figure the Power Conditioning and Distribution Unit (PCDU) are connected to the batteries and to the solar panels. The PCDU is the subsystem responsible for the power management of the satellite. The Telemetry, Tracking & Command (TT&C) is the unit responsible for receiving the TCs sent by the Ground Station, verify their integrity, correcting possible errors, unpacking the command and delivering it to the OBC. The other modules in Figure 1 are sensors and actuators used by the OBC for the attitude control of the satellite.

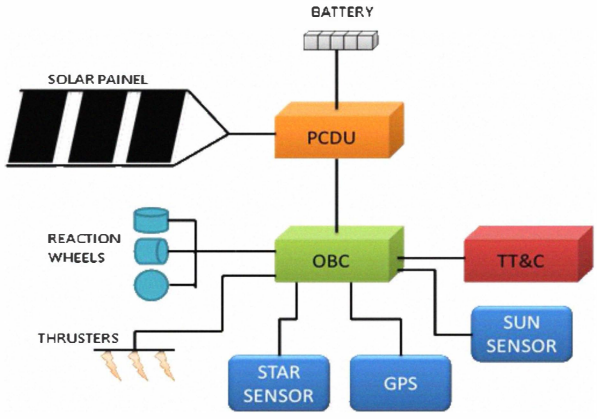


Figure 1. A generic satellite control unit

The choice of processor to be used in the OBC should take into account all the data processing the on-board computer must perform and the dependability this kind of application requires. The LEON3 soft-processor was selected by engineers from the Brazilian Space Program to be used as the OBC's main processing unit. It is important to mention that the Brazilian Space Program is the main interested in the outcomes of this research.

The LEON soft-processor [9] was developed by the European Space Agency (ESA) to be an open-source high

performance processor able to be used in their space missions. Currently, the LEON processor family is developed by the Aeroflex Gaisler and the LEON3 is the most consolidated version in the market and have been used in aviation and commercial applications.

The LEON3 VHDL model implements a 32-bit processor conforming to the SPARC V8 architecture. It is designed for embedded applications presenting the following features: separate instruction and data caches, hardware multiplier and divider, interrupt controller, 24-bit timers, UARTs, power-down function, watchdog, 16-bit I/O port and a flexible memory controller. Additional modules can easily be added using the on-chip AMBA AHB/APB buses. The VHDL model is fully synthesizable with most synthesis tools and can be implemented in both FPGAs and ASICs. Simulation can be done with all VHDL-87 compliant simulators [10].

Figure 2. shows an example of a LEON3 system design. It is possible to notice that the design is bus-centric, as all the IP-cores are connected to AMBA AHB/APB bus.

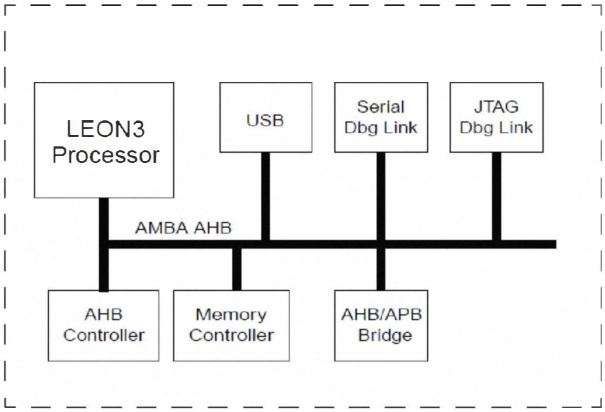


Figure 2. Example of a LEON3 system design

In order to use the LEON3 soft-processor embedded on a COTS SRAM FPGA aiming an OBC for satellites it is necessary to improve the processor mitigation with a fault-tolerance technique. In this paper we changed the LEON3 system design showed in Figure 2. to add the capability of identify processor faults during its execution. The proposed architecture is described next in Section IV.

IV. PROPOSED DUPLICATION APPROACH

The OBC developed in this research work is fully compliant with mission requirements provided by the Brazilian Institute of Space Research (INPE). The main difference between the proposed architecture and INPE's specification is the selected processor. In an ongoing satellite design, INPE is employing ERC32 [11], which is a 32-bits radiation tolerant RISC processor manufactured by Atmel. In the proposed architecture, following INPE's space engineer requests, the LEON3 soft-processor from Gaisler/ESA has been adopted as the main processing unit. Both processors are compliant with the SPARC V8 architecture, and the same software applications can run on both architectures.

At the board level, INPE has defined the OBC to be designed as a redundant architecture. In the ongoing OBC implementation at INPE, there are two redundant ERC32 processors running in parallel, but in a hot spare configuration. All the information is processed by the two processors simultaneously, but only one of them provides data to the on-board subsystems at a time. Following this requirement, as shown in Figure 4, the proposed OBC board has two redundant COTS SRAM based FPGAs and our technique allows the identification of an error in one of the FPGAs. As a consequence, it is possible to avoid this error to be propagated to the rest of the system. The prototyping board employed so far has just one FPGA, but the whole system has been planned targeting a two FPGAs OBC board.

The idea behind the proposed technique is to implement on-line fault detection at run time. This approach allows fault masking through hardware replication, helping to avoid fault propagation to the remaining system components.

Different fault detection techniques have been adopted. On the FPGA embedded processor, for instance, the fault detection is accomplished through "processor control flow". A redundant and non-intrusive soft processor is added to the AMBA bus. In this case, the control flow consists of the comparison between the outputs provided by the two processors. In the event of a mismatch, a fault is detected and announced to the system. Figure 3 shows the proposed LEON3 redundant architecture.

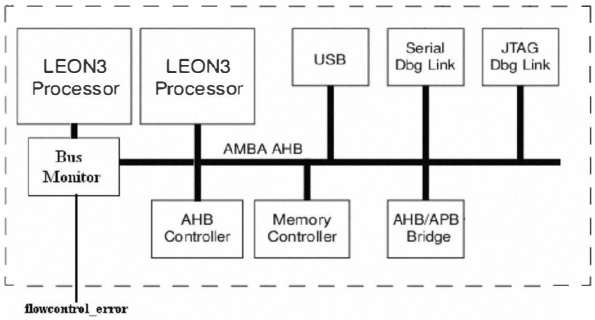


Figure 3. LEON3 proposed architecture with non-intrusive redundant processor

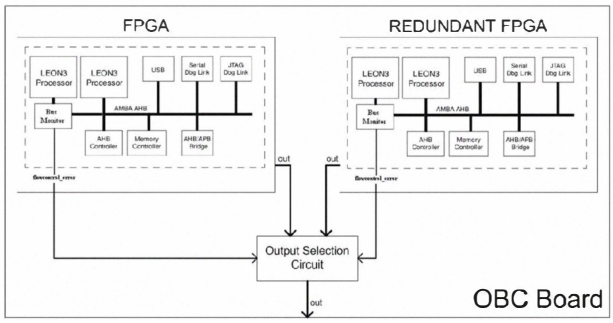


Figure 4. OBC Board, showing the fault tolerance strategy using FPGA redundancy with four LEON processors in total

In the architecture shown in Figure 3, a Bus Monitor has been developed in order to compare the results of both, the main and the redundant processor. The Bus Monitor compares the data written to the bus by the processors. The two processors work in parallel, processing the same data, simultaneously. However, the data processed by the redundant processor does not go to the bus. These data is used only by the Bus Monitor to check whether the main processor has obtained the same result as the redundant one. In case of a mismatch, the "controlflow_error" signal is set, and it is used by the "Output Selection Circuit" (see Figure 4) to provide the correct data to the satellite subsystems.

When compared to other fault mitigation techniques the use of a non-intrusive redundant processor showed to have some advantages. When using only two redundant processors instead of the more traditional Triple Modular Redundancy (TMR) technique, the FPGA area overhead is considerably reduced. As a result, it is possible to use regular COTS FPGA without the need of employing larger FPGAs that are usually much more expensive and even harder to find their radiation hardened versions. Also, a TMR design could become so large that it may not fit in a COTS FPGA. Another advantage is that the software running on the processor does not need to be modified avoiding loss of performance as it happens when hybrid software and hardware fault detection techniques are used [8].

V. VALIDATION VIA FAULT INJECTION

The continuous evolution of the fabrication process of integrated circuits, in terms of technology shrinking and device density, results in increasing the radiation sensitivity of these chips even at the sea level. An efficient approach used in the dependability evaluation of contemporary VLSI devices is through accurate fault injection techniques [12][13]. There are different methods for injecting faults in integrated circuits mimicking hard or soft errors. Some of these methods are being used in order to validate the fault tolerant technique applied to the proposed on-board computer.

Initially, a simulation based fault injection was developed in order to validate the logical functionality of the non-intrusive fault tolerant technique. In this approach, the fault injection is done by a simulation script that changes the value of internal OBC signals and observes the injected fault effects.

The simulation script uses the TCL language and works with the Mentor Modelsim simulation tool [14]. The main idea is to use the "force" function available in the Modelsim to simulate a change of value in one of the internal processor signals, in other words a simulation of a fault injection.

The "force" function, as the name says, can force a value in any of the signals during a simulation. Therefore, the script works as follow: it starts the simulation and runs it until a selected period of time is reached; then, the script selects one of the processor internal signals, checks it value and inserts the fault by inverting a bit of this value; with the fault inserted, the script runs the simulation to the end of the test program checking if an error was detected by the Bus Control; at the end of the simulation the script checks the results of all signals and compares with the results of the simulation with no fault injection.

The simulation script runs the simulation once for each one of the processor internal signal, injecting a fault in a different signal at a time. This way we can simulate the effect of a SEU in all the internal signals of the processor and check what sort of problem this fault generates.

With the script it is possible to apply different fault models to the system controlling the time that the "force" function is applied to a signal. If the signal is forced with a value for the entire simulation time, this value will not change in case of a rewrite, in other words, we are simulating a latch up in that signal. As we are trying to simulate SEU effects it is necessary to cancel the "force" function using the "noforce" function after a certain period of time. This way the signal will change it value in case of a rewrite.

It is necessary to analyze the behavior of the OBC during the fault injections to observe the different effects caused by the SEUs [2]. We found three different behaviors caused by the fault injection.

- Silent: the SEU effects eventually disappear. This is detected when the non-faulty and faulty states become identical for all FFs and memories after fault injection.
- Latent: the SEU effects were not propagated to an output of the circuit, but the internal state of the circuit is not equivalent at the end of the execution.
- Failure: the pulse effects are propagated to at least an output of the circuit under test. In this case the Bus Monitor is able to detect the failure.

Several simulation-based techniques have been developed [10][15] and could be used to estimate SEU sensitivity in the proposed OBC architecture. However, complex projects with hundreds of thousands of gates would represent a huge amount of fault possibilities, and the capability of analyzing a significant number of faults may not be realistic. Therefore, the major drawback of using simulation techniques is the time consuming effort resulting from the huge computational power required to perform circuit simulation under a large number of faulty conditions. Emulation-based fault injection techniques have been proposed in past years with the objective of accelerating fault injections campaigns [16-19]. Features present in FPGAs like reconfiguration and readback capabilities are used to increase the observability and controllability of a circuit under test (CUT), making FPGAs important tools for fault injection emulation.

An emulation-based fault injection strategy is under development for a more accurate SEU sensitivity estimation of the proposed OBC. Initially, a basic fault injector was developed based on the FT-UNSHADES [20] platform. It is a non-intrusive technique that automates the SEU emulation at any flip-flop and at any moment. The operation flow of the fault injector developed consists of the steps listed next:

1. Download the original OBC (without the proposed fault tolerant technique) to the FPGA device and run a fault free emulation.
2. The outputs of the fault free emulation are saved as golden results to be used later on in the fault campaigns.
3. Download the proposed OBC to the FPGA device and run a fault free emulation in order to compare the outputs against the original one. This step shows that the processor duplication and the comparator added to the bus have no influence in the outputs.
4. Creation of a fault location list.
5. The fault campaign is performed. The location of the fault to be injected is randomly chosen from the generated list before. The injection time is also defined in a random way.
6. The outputs observed in the fault campaign are compared to the golden output, and the faults are classified as follows: ***undetected***, if differences to the golden output are found without any Bus Monitor signaling; ***detected***, if the checker detects the fault; ***silent/latent***, if the outputs are the same and Bus Monitor signal does not go high; otherwise, the fault is classified as ***problem***.

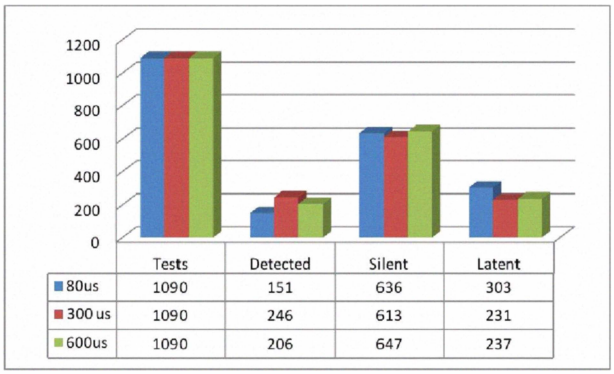
There are several other techniques and optimizations for fault emulation [21], and some of them would be considered for adoption in a future work.

Hardware fault injection is a widely accepted approach to evaluate the behavior of a circuit in the presence of faults. Thus, it plays a key role in the design of robust circuits [13]. In addition to the on-going fault emulation activities, a new partnership is under discussion with the Physics Institute at University of São Paulo (Instituto de Física, USP) and with the Industrial Engineering Faculty (Faculdade de Engenharia Industrial, FEI). USP has an electrostatic accelerator Pelletron 8UD of 8 MV, a proton/alpha particle accelerator Tandem of 1.7 MV. *FEI* has an X-Ray Diffractometer XRD-7000 Shimadzu. These facilities will allow the realization of physical hardware fault injection campaigns. The results obtained from the fault injection campaigns will be useful in both, the validation of non-intrusive fault tolerant techniques applied to the OBC, and also in the emulation-based technique. The expected results to be obtained using FEI/USP facilities [21-23] could be useful in order to show that the proposed OBC architecture is eligible as a fault tolerant device.

VI. Simulation Results

During the execution of the software on a processor, different parts of the processor are used at different times. To improve the odds of an injected fault to affect the processor generating a failure, the simulation was executed with the fault been injected at different simulation times. First the SEUs were injected at the beginning of the software execution, at 80 us of simulation time. Then, a second simulation was executed with the fault been injected at 300 us. And last, the fault was injected at 600 us. The total simulation time is 800 us.

Figure 5 shows the results after running the simulation scripts inserting the SEU at the three different simulation times.



	Tests	Detected	Silent	Latent
80us	1090	151	636	303
300 us	1090	246	613	231
600us	1090	206	647	237

Figure 5. Simulation results

As shown in Figure 5, we have run 1,090 simulation tests to each one of the fault injection times. The reason for this amount of tests is that there are 1,090 internal signals in the processor and we run simulation tests inserting a fault in each one of the internal signals.

It is possible to notice that, on average, 18% of the inserted faults propagate to an output of the processor and become a failure. In all the cases that a failure occurred the Bus Monitor was able to detect it and to activate the "controlflow_error" signal, allowing this failure to be mitigated by the redundant FPGA data.

On the other cases, 82% on average, the fault did not propagate to the processor output. These faults are silent in the cases they disappeared after a rewrite, or latent in the cases it changes some internal signal of the processor.

We can also notice the different behavior of the faults when they are injected at different simulation times. The percentage of faults injected at 300 us that became failures is 22.5% against 13.8% of the faults injected at 80 us. That shows that the processor is more susceptible to SEUs effects in the middle of the running program, when more of its resources are being used.

VII. Conclusions and Future Work

Brazil is one of the few and oldest countries in the world with an active space program. In order to have technology to build and to launch satellites, it is important to keep developing competence in this area of hardened electronic designs for space applications. Therefore, research works concerning fault tolerance techniques, hardening satellite modules like OBC and validation techniques of components are of fundamental importance.

This paper presents a non-intrusive fault tolerant technique applied to an on-board computer for space applications. This OBC has been designed following specifications provided by the Brazilian Institute of Space Research (INPE).

The entire validation flow of the project was shown starting by the simulation of the proposed non-intrusive fault tolerant technique going through the emulation-based fault injection,

and targeting a future exposition of the design to radiation environments, in order to stress the components for final qualification. The preliminary simulation results show the functionality and capability of the fault tolerant technique proposed for the OBC. The emulation-based fault injection platform is being developed to run longer fault campaigns looking for better fault tolerant estimation. Finally, the whole environment of hardware-based fault injection is being prepared to qualify the components manufactured in Brazil for space applications and for validation of the developed fault tolerant and fault injection techniques.

As a future work, additional fault models will be applied to the validation chain in order to better estimate radiation sensitivity. A framework will be developed to integrate all phases of the validation process aiming the simplification of the validation of other fault tolerant techniques. Thus, diminishing the necessity of the expensive time exposition of the design in radiation chambers, and increasing the number of successful fault tolerant projects.

REFERENCES

[1] N. Battezzati, L. Sterpone. M. Violante, *Reconfigurable Field Programmable Gate Array for Mission-Critical Applications.* New York, NY: Springer, 2011.

[2] L. Entrena, M. Garcia-Valderas, R. Fernandez-Cardenal, A. Lindoso, M. Portela, and C. Lopez-Ongil, "Soft Error Sensitivity Evaluation of Microprocessors by Multilevel Emulation-Based Fault Injection," *IEEE Transactions on Computers*, no. 99, pp. 1–1, 2010.

[3] J. George, R. Koga, G. Swift, G. Allen, C. Carmichael, and C. W. Tseng, "Single Event Upsets in Xilinx Virtex-4 FPGA Devices," *2006 IEEE Radiation Effects Data Workshop*, pp. 109-114, Jan 2007.

[4] M. Straka, Z. Kotasek, and J. Winter, "Digital Systems Architectures Based on On-line Checkers," 2008 11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools, Ieee, 2008, pp. 81-87.

[5] M. Straka, J. Tobola, and Z. Kotasek, "Checker Design for On-line Testing of Xilinx FPGA Communication Protocols," 22nd IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems (DFT 2007), Ieee, 2007, pp. 152-160.

[6] M. Straka, J. Kastil, and Z. Kotasek, "Modern Fault Tolerant Architectures Based onPartial Dynamic Reconfiguration in FPGAs," IEEE 13th International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS), 2010, 2010, pp. 173-176.

[7] J. B zalke and S. Pandey, "Dynamic Partial reconfigurable embedded system to achieve Hardware flexibility using 8051 based RTOS on Xilinx FPGA .," 2009 International Conference on Advances in Computing, Control, and Telecommunication Technologies, 2009, pp. 2009-2011.

[8] P. Bernardi, L. Bolzani, M. Rebaudengo, M.S. Reorda, F. Vargas, M. Violante, "A new hybrid fault detection technique for systems-on-a-chip," IEEE Transactions on Computers, pp. 185-198, Feb. 2006.

[9] Aeroflex Gaisler, LEON3 Multiprocessing CPU Core Product Sheet, URL: http://www.gaisler.com/doc/leon3_product_sheet.pdf

[10] A. da Silva and S. Sanchez, "LEON3 ViP: A Virtual Platform with Fault Injection Capabilities," *2010 13th Euromicro Conference on Digital System Design: Architectures, Methods and Tools*, pp. 813-816, Sep. 2010.

[11] Atmel Corporation, TSC695F SPARC 32-bit Space Processor, Atmel 2003, URL: http://www.atmel.com/dyn/resources/prod_documents/doc4148.pdf

[12] F. Kastensmidt, L. Carro, and R. Reis, *Fault-Tolerance Techniques for SRAM-based FPGAs*. Boston, MA: Springer US, 2006.

[13] M. Nicolaidis, Ed., *Soft Errors in Modern Electronic Systems*, vol. 41. Boston, MA: Springer US, 2011.

[14] Mentor Graphics Corporation, ModelSim User Manual, Mentor 2010, URL: http://www.actel.com/documents/modelsim_ug.pdf

[15] H. Ziade, R. Ayoubi, and R. Velazco, "A Survey on Fault Injection Techniques," *Int. Arab J. Inf. Technol.*, vol. 1, no. 2, pp. 171–186, 2004.

[16] L. Entrena, M. Garcia-Valderas, R. Fernandez-Cardenal, A. Lindoso, M. Portela, and C. Lopez-Ongil, "Soft Error Sensitivity Evaluation of Microprocessors by Multilevel Emulation-Based Fault Injection," *IEEE Transactions on Computers*, no. 99, pp. 1–1, 2010.

[17] C. Lopez-Ongil, M. Garcia-Valderas, M. Portela-Garcia, and L. Entrena, "Autonomous Fault Emulation: A New FPGA-Based Acceleration System for Hardness Evaluation," *IEEE Transactions on Nuclear Science*, vol. 54, no. 1, pp. 252-261, Feb. 2007.

[18] P. Vanhauwaert, R. Leveugle, and P. Roche, "Reduced Instrumentation and Optimized Fault Injection Control for Dependability Analysis," *2006 IFIP International Conference on Very Large Scale Integration*, pp. 391-396, Oct. 2006.

[19] M. a Aguirre, V. Baena, J. Tombs, and M. Violante, "A New Approach to Estimate the Effect of Single Event Transients in Complex Circuits," *IEEE Transactions on Nuclear Science*, vol. 54, no. 4, pp. 1018-1024, Aug. 2007.

[20] J. Tombs et al., "The implementation of a FPGA hardware debugger system with minimal system overhead," *Field Programmable Logic and Application*, pp. 1062–1066, 2004.

[21] M. A. G. Silveira et al., "Performance of electronic devices submitted to X-rays and high energy proton beams," *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, vol. 48, no. 6, pp. 2080-2080, Jul. 2011.

[22] K. Cirne et al., "Comparative study of the proton beam effects between the conventional and Circular-Gate MOSFETs," *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, vol. 53, no. December, pp. 2006-2006, Jul. 2011.

[23] J.A. De Lima et al., X-Ray Radiation Effects in Overlapping Circular-Gate MOSFET's, Proceedings of the Conference on Radiation Effects on Components and System, IEEE Transactions on Nuclear Science, 2011, in press.