electronics

MDPI

Article

Fault-Tolerant FPGA-Based Nanosatellite Balancing High-Performance and Safety for Cryptography Application

Laurent Gantel [1,*], Quentin Berthet [1], Emna Amri [2,*], Alexandre Karlov [2] and Andres Upegui [1,*]

[1] inIT, Hepia, University of Applied Sciences and Arts of Western Switzerland, 1202 Geneva, Switzerland; quentin.berthet@hesge.ch

[2] CYSEC SA EPFL Innovation Park, 1015 Lausanne, Switzerland; alexandre.karlov@cysec.systems

* Correspondence: laurent.gantel@hesge.ch (L.G.); emna.amri@cysec.systems (E.A.); andres.upegui@hesge.ch (A.U.)

Abstract: With the growth of the nano-satellites market, the usage of commercial off-the-shelf FPGAs for payload applications is also increasing. Due to the fact that these commercial devices are not radiation-tolerant, it is necessary to enhance them with fault mitigation mechanisms against Single Event Upsets (SEU). Several mechanisms such as memory scrubbing, triple modular redundancy (TMR) and Dynamic and Partial Reconfiguration (DPR), can help to detect, isolate and recover from SEU faults. In this paper, we introduce a dynamically reconfigurable platform equipped with configuration memory scrubbing and TMR mechanisms. We study their impacts when combined with DPR, providing three different execution modes: low-power, safe and high-performance mode. The fault detection mechanism permits the system to measure the radiation level and to estimate the risk of future faults. This enables the possibility of dynamically selecting the appropriate execution mode in order to adopt the best trade-off between performance and reliability. The relevance of the platform is demonstrated in a nano-satellite cryptographic application running on a Zynq UltraScale+ MPSoC device. A fault injection campaign has been performed to evaluate the impact of faulty configuration bits and to assess the efficiency of the proposed mitigation and the overall system reliability.

Keywords: FPGA; fault-tolerance; SEU; TMR; nano-satellite; dynamic and partial reconfiguration

check for updates

Citation: Gantel, L.; Berthet, Q.; Amri, E.; Karlov, A.; Upegui, A. Fault-Tolerant FPGA-Based Nanosatellite Balancing High-Performance and Safety for Cryptography Application. *Electronics* **2021**, *10*, 2148. https://doi.org/10.3390/electronics10172148

Academic Editors: Juan M. Corchado and Rui Pedro Lopes

Received: 13 July 2021
Accepted: 27 August 2021
Published: 3 September 2021

1. Introduction

Nano-satellites have been recently under the spotlight due to their versatility in many different applications. Compared to regular satellites, now that nano-satellite can be equipped with a significant amount of computational resources, they offer a cost-efficient way to carry out several applications from diverse market verticals and a faster go-to-market. Applications that nano-satellites can effectively carry out are, most notably, satellite communication and connectivity (e.g., Space-X [1]); Internet-of-Things applications (e.g., Astrocast [2], Hiber [3]); Earth Observation for defense, oil and gas and maritime industries (e.g., Kleos [4], Spire [5]) as well as space exploration and science in general.

Powerful computational resources are provided by FPGA devices which are being increasingly embedded in nano-satellites, not only in radiation-hardened FPGA but also in commercial off-the-shelf devices. They provide high performance, reasonable power consumption and good flexibility. These qualities allow them to be advantageously embedded as a payload board in a nano-satellite system. Payload boards often include specific devices such as sensors or cameras, requiring the execution of dedicated computation tasks like image and signal processing, data mining or sensors fusion.

One of the issues that must be tackled when sending an FPGA device into space is the fact that FPGAs, especially SRAM-based ones, are prone to radiation effects [6]. Single Event Effects (SEE) are caused by the impact of a particle on a memory cell of the FPGA. This can lead to a Single Event Upset (SEU) which will provoke a bit flip of the configuration memory cell content. An SEU can affect both the design state or the design routing, leading to data corruption, unusable functionalities, or even a complete system failure.

The case study in [7] details the implementation of radiation-error mitigation techniques on a Xilinx Kintex 7 FPGA device, especially Triple Modular Redundancy (TMR). TMR consists in triplicating a function that needs to be protected against radiations. The output of each function or module is compared with the two others and a voter selects the output that is identical for at least two of the modules [8,9].

Another solution against SEE is memory scrubbing. A memory scrubber is a module which periodically checks the FPGA configuration memory in order to detect bit flip caused by SEE. The aim is to detect and correct, avoiding a complete failure of the device operations. An external memory scrubber using a rad-hard FPGA to check the configuration memory of the SRAM payload FPGA is described in [10]. Considering the importance of this mechanism to detect and be able to correct radiation-induced errors, Xilinx provides a hardware IP core to rapidly and seamlessly include this mitigation technique in a design [11]. This IP allows internal checking of the FPGA configuration memory with low development cost, with the drawback that, being implemented in the FPGA logic, it requires an upper layer to protect this scrubber against an SEU.

A third method to handle the radiation issue is to take advantage of dynamic and partial reconfiguration (DPR). This feature provided in Xilinx and Intel FPGAs allows reconfiguration of a partition of the FPGA without disturbing the rest of the design. Thus, it can be used to reload a damaged partition, load a new partial configuration avoiding the permanently damaged areas or reconfigure a partition with a more robust implementation. In the latter case, the implementation could use TMR or another fault correction mechanism to tackle the radiation effects more efficiently. It has the benefit of reducing FPGA resources usage, which in turn reduces the size of the area vulnerable to SEEs. It also decreases power consumption and allows for faster and easier transfer of a new applications to a device. An example is the work presented in [12], where they used TMR and dynamic reconfiguration for rewriting faulty frames after comparing them to identically routed redundant blocks.

A more sophisticated fault mitigation technique is presented in [13]. The authors describe a framework called HARFT (Hybrid Adaptive Reconfigurable Fault Tolerance) that proposes different operating modes that can be activated depending on the necessary level of fault mitigation against application performances. The framework implements scrubbing techniques and takes advantage of the partial reconfiguration capability of the FPGAs. Additional techniques such as ECC, parity schemes and TMR are also used. The mitigation mechanisms implemented in the processor embedded in the FPGA (Zynq) allows switching from a Symmetric Multi-Processor (SMP) mode to an Asymmetric Multi-Processor (AMP) mode after a reset of the system. The system allows the user to choose a third mode in which one or several Microblaze processors operate in lockstep in the programmable logic, potentially coupled with hardware accelerators. The three modes have been tested in different scenarios with simulated SEEs sent to a Zynq device containing soft processors but no hardware accelerators.

The combination of these mitigation techniques leads to the design of complex systems able to adapt themselves to the level of radiation to which they are subjected. The ability to balance between resources usage and operational safety is a key point to reduce the cost of the high-performance FPGA-based nano-satellite board, while maintaining device adaptability and long-term reliability. In this work, we propose a payload computation platform based on a Zynq Ultrascale+ MPSoC FPGA. The platform provides different reconfigurable partitions (RPs) connected with streaming interfaces in order to offer efficient computation acceleration to the application. A memory scrubbing mechanism based on the Soft Error Mitigation Controller provided by Xilinx is integrated into the platform. This scrubber provides information about the level of radiation to which the device is currently subject and, based on this information, the application is automatically and dynamically deployed in two different modes: a “High-Performance Mode”, where each partition hosts a different hardware co-processor, or in a “Safe Mode”, where streamed partitions are dynamically reconfigured in order to act as a TMR version of a single hardware core. A Low-Power mode in which RPs are disabled thanks to the DPR is also provided to fulfill nano-satellite application requirements regarding energy-saving constraints.

Compared to [13], our platform offers the ability to efficiently integrate reconfigurable hardware accelerators that can take advantage of the streaming structure of the interconnect. It provides a controller for the configuration memory scrubber and upper software layers to handle correctable but also non-correctable soft errors due to radiation. Finally, the platform leverages the heterogeneity provided by the MPSoC using one of the real-time processors to perform both platform monitoring and the fault injection campaign.

2. Materials and Methods

2.1. Nano-Satellite System Overview

The proposed platform is a FPGA-based payload board intended to be embedded in a nano-satellite along with an on-board computer (OBC). From the Mission Control Center (MCC) point of view, the OBC is the entry point for communication with the Earth, especially with the Attitude Determination and Control System (ADCS) sub-module responsible for satellite navigation. Messages for the payload board are transferred by the OBC through a serial link whereas scientific instruments can use dedicated high-speed interfaces with the FPGA board (Figure 1). An example of a nano-satellite is the CubeSat format [14]. In this specification, the different boards that compose such a miniaturized satellite are stacked on top of each other inside multiple 10 cm cubes.

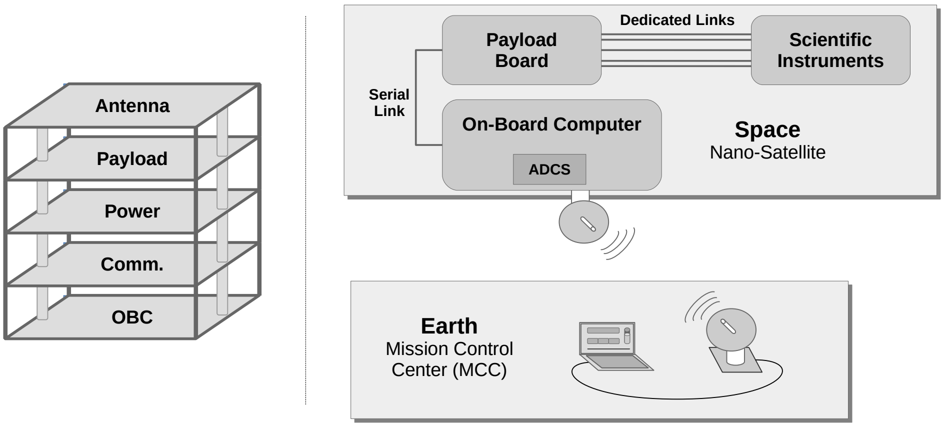

Figure 1. Nano-satellite system overview.

2.2. Payload Board Prototype

The prototype of the payload board is built upon the Zynq Ultrascale+ MPSoC FPGA device. This MPSoC includes, in addition to the reconfigurable logic, a quad-core Cortex-A53 processor, a dual-core Cortex-R5 micro-controller and 2GB of DDR4 memory (Figure 2). Communication is articulated around an AXI-Stream Interconnect (AXIS-IC) allowing the processor and the reconfigurable partitions to exchange data information through the DDR4 memory. A bridge performs stream packet conversion to permit communication between the memory-mapped domain (i.e., the processor and its DDR memory) and the stream domain (i.e., the reconfigurable partitions).

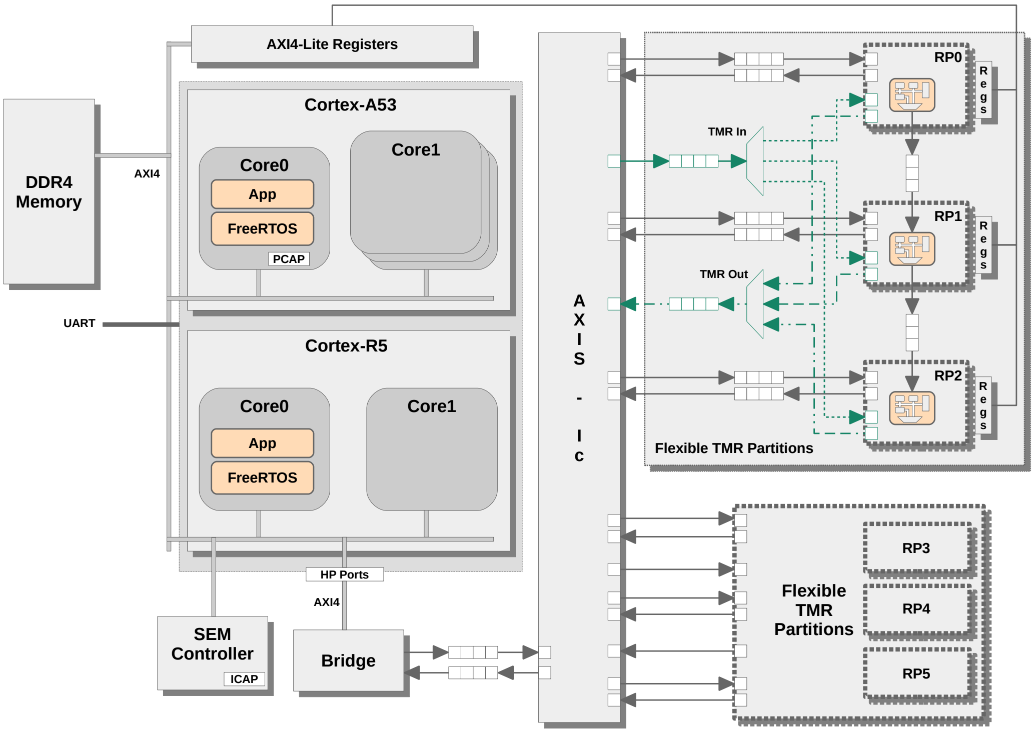

Figure 2. Payload Board Prototype.

On the Cortex-A53, the first core hosts the payload application which runs on top of a FreeRTOS operating system [15]. Three more cores are available to run other software tasks or operating systems if necessary. The main core manages the communication with the OBC through the serial link (UART), the application execution and the hardware tasks deployment over the reconfigurable partitions.

RPs are packed into flexible TMR partitions providing the datapath and voters logic to enable the TMR mechanism (TMR In and TMR Out). In addition to the individual stream of each partition, the three partitions inside a TMR set are linked via direct streams allowing implementation of efficient dataflow processing. This communication interface with the reconfigurable partitions has been thought to provide the ability to switch from a High-Performance mode to a Safe mode, and vice-versa.

In High-Performance mode, each partition in the set can run a different hardware tasks and communicate with the rest of the system though its dedicated stream interface. In Safe mode, the three partitions share the TMR interface. The switch between the two modes is controlled by the processor via dedicated AXI4-Lite registers. The decision to switch is based on the monitoring information provided by the Soft-Error Mitigation (SEM) Controller.

2.3. Fault Mitigation Mechanisms

In addition to the TMR applied to the reconfigurable partitions, the configuration memory is protected against SEU using the scrubber IP provided by Xilinx for their FPGAs. This IP, called Soft-Error Mitigation (SEM) IP, is integrated in the design and controlled by the rest of the logic. A wrapper has been added around the SEM IP in order to access important information from the processor AXI4 interface. This controller allows the Cortex-R5 to send commands to the IP for state changes or fault injection, and to parse monitoring reports generated by the IP during its activity. Information from these reports is used to feed several counters, counting the number of injected faults, how much has been corrected or how many are uncorrectable. It also indicates the current SEM state.

The detected faults can be categorized, or classified, as correctable or uncorrectable, and as essential or non-essential (Figure 3). The later information is generated during the bitstream generation phase by the Xilinx tools in the form of an ASCII file containing a one or a zero character for each bit in the design, indicating if it is essential or not. It represents several million bits for the UltraScale+ MPSoC family devices. This information is requested by the SEM each time a fault has been detected. A module has been added to the controller to provide the correct information to the SEM regarding the frame where the fault has been detected (Figure 4). If the bit is essential, the SEM intends to repair it; otherwise the fault will be discarded and the scrubber will continue to scan the configuration memory.

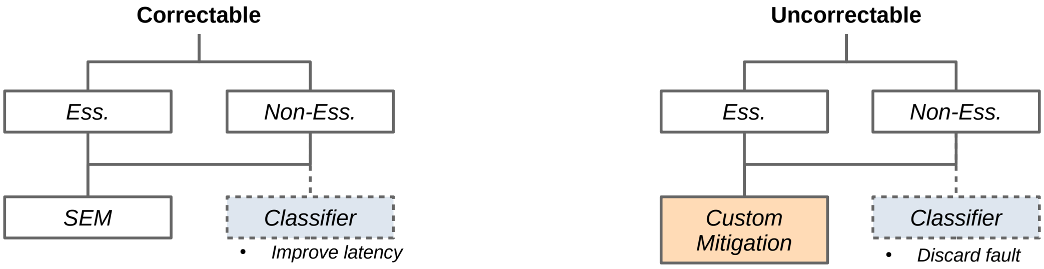

Figure 3. Fault Classification.

In the case of an uncorrectable error, a custom mitigation procedure has been setup: a backup memory containing the original frames is used to dynamically restore the faulty frame using the DPR. Frames content is extracted using the RapidWright utility [16] and stored in the DDR4 memory at boot time (Frames DataBase).

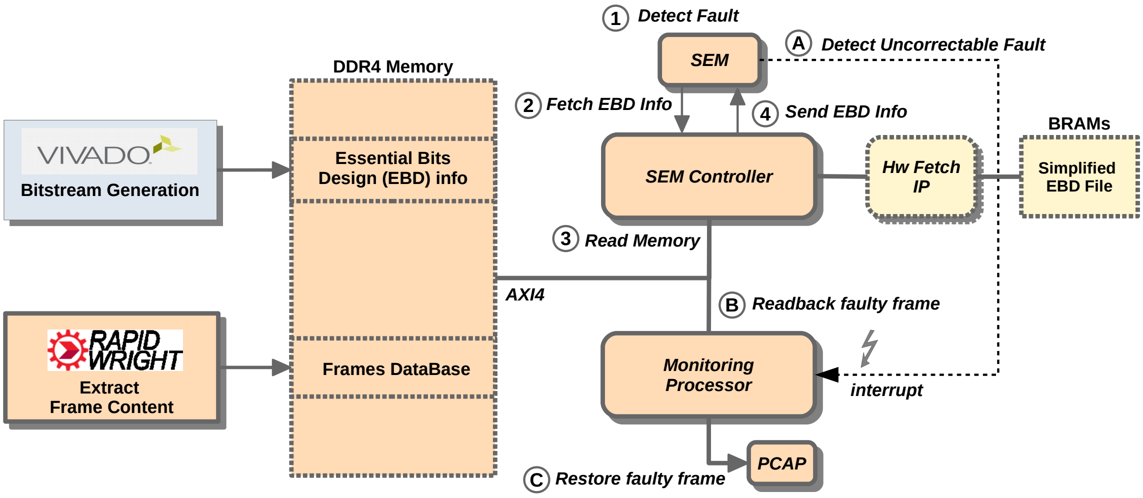

Figure 4. Custom Mitigation Procedure.

The EBD info file generated by the Vivado tools has a major drawback: it is highly memory-consuming (several MBytes) and so it must be stored in a memory distant from the SEM controller. In order to leverage this issue, an optional hardware module can be added (Hw Fetch IP). For each frame, instead of storing the information of each bit contained in the frame, we store a bit indicating whether the frame contains an essential bit or not. This allows storing the EBD information in several BRAM blocks, simplifying the content of this file. With most of the essential bits being concentrated in about 15% of the frames, the overhead to check a frame—even if the faulty bit is not essential—is compensated by the improvement of the read memory latency when performing a fetch of EBD information for the SEM Controller.

2.4. Platform Monitoring

The platform reliability is evaluated thanks to a dedicated testbench permitting simulated faults injection (Figure 5). The Cortex-R5 is used to monitor the proper functioning of the application running on the Cortex-A53 and the programmable logic. It also executes the fault injection task allowing the efficiency of the mitigation mechanisms to be measured.

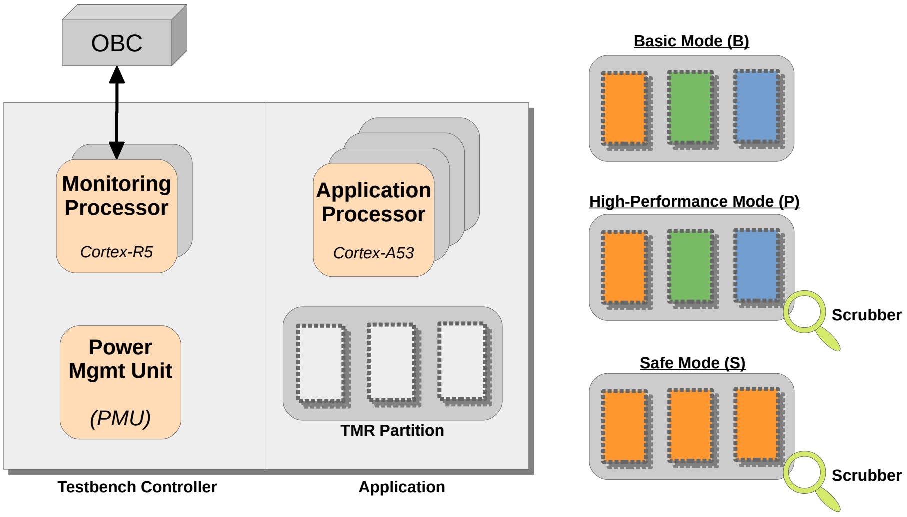

Figure 5. Prototype Testbench.

On this monitoring processor, a watchdog timer is used to ensure that the Cortex-A53 is still alive and a task periodically receives the processing result from the application processor (Figure 6). If the processing result is erroneous or if the watchdog expires, the monitoring processor concludes that a failure has occured and stops the testbench for the current run. The elapsed time is used to measure the system reliability before the occurrence of a critical failure and the system is reset thanks to the Power Management Unit (PMU) contained in the MPSoC. Information exchange between the three cores is done via the Inter-Processor Interrupt (IPI) mechanism.

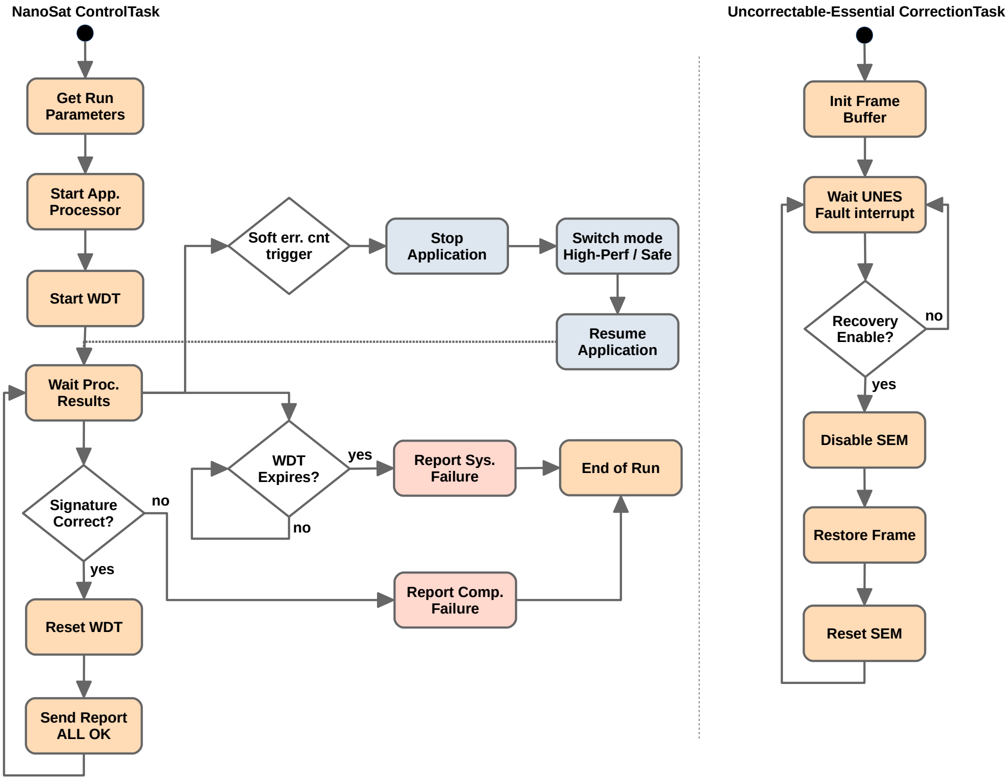

Figure 6. Monitoring Diagram.

The Cortex-R5 also monitors the radiation level using the information provided by the SEM, especially the number of soft errors detected by the configuration memory scrubber.

This information is compared to a threshold that should be defined regarding the number of soft errors that could lead to a system failure. The number of soft errors is periodically reset in order to obtain an error rate. If this rate is greater than the threshold, the application is stopped and the platform is reconfigured to switch to the Safe mode, and the application is resumed. When this rate decreases below the threshold, the platform returns to the High-Performance mode.

2.5. Fault Injection Mechanism

Fault injection can be done using two sources of random frames addresses. The first source is internal to the SEM Controller. A LFSR PRNG is used to generate random linear frame addresses that will target all platform. The second source is a FIFO that can be fed by the monitoring processor via the AXI4 interface. This interface can be used to generate constrained LFA in software and so to target precise regions in the platform. It allows the tester to inject faults only in the areas that are covered by a mitigation mechanism, and so evaluate its efficiency without being disturbed by failures on non-protected areas.

The efficiency of the platform is evaluated measuring the reliability. As the application running on the platform is deterministic, it was necessary to have a different set of addresses for each run to expect a different result. In this way, a random seed is provided by the PC connected to the UART link at the beginning of each run. This seed is used by the platform to generate the random addresses that target the desired area.

2.6. Testbench Use-Case

A use-case featuring a communication between the nano-satellite and the Mission Control Center is introduced in Figure 7. The nano-satellite is equipped with a camera and takes pictures of the Earth. For each picture, it processes the image using a Grayscale filter followed by a Sobel filter. The result is encrypted with AES-256 and stored in the internal memory. In order to secure the communication with Earth, the image is then signed using the SPHINCS$^+$ post-quantum signature scheme to allow robust data authentication [17]. This signature scheme has been chosen because it is a good candidate to secure the future communication of nanosatellite systems. Being designed to prevent attacks from near-coming quantum computers, it provides an additional level of security to authenticate information such as configuration data or firmware updates coming from the Mission Control Center.

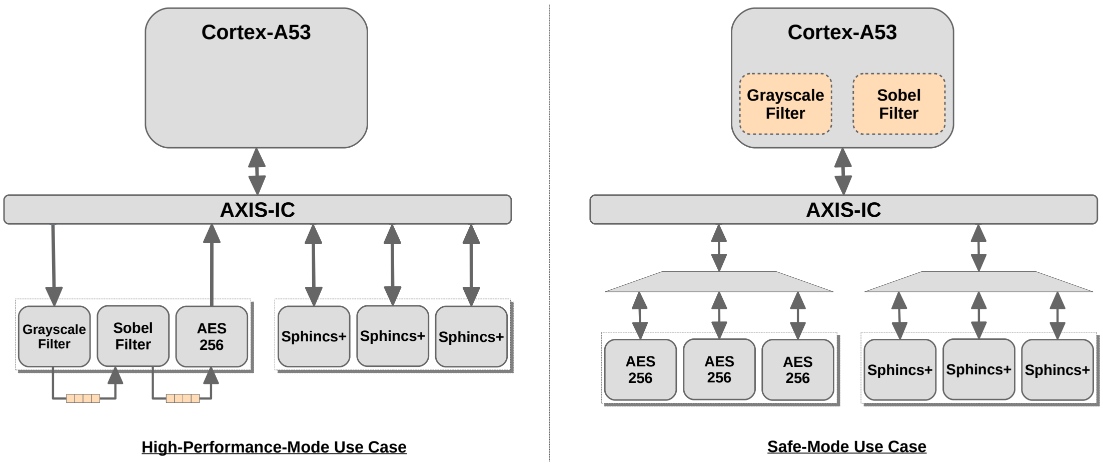

Figure 7. Operation modes use cases.

In Safe mode the first flexible TMR partition is used to host a TMR version of the AES-256 IP. This IP has been chosen over the Grayscale and Sobel filters because it benefits the most from the hardware acceleration. Both filters are executed in software on the processor. The second flexible TMR partition is used to host three instances of a hardware implementation of the SPHINCS$^+$ algorithm [18].

3. Results

3.1. Platform Resources Usage

Table 1 shows the resources occupied by the static partition and the reconfigurable modules in terms of LUT, BRAM and DSP, as well as the estimated power consumption for each of these partitions. Figure 8 gives the estimated power consumption for the platform in the different supported modes. Among the reconfigurable partitions, the most resource- and energy-consuming are the SPHINCS$^+$ IP and the AES IP. In Safe mode, the resource usage overhead compared to the High-Performance mode will be high, but these modules have the significant advantage of being implemented in hardware [18]. These numbers also show that, in this case, the impact of the dynamic and partial reconfiguration to reduce the overall power consumption of the platform would be minimal. Occupying the largest share of energy consumption is the static partition, especially the processing system which is responsible for more than 90% of the power consumption.

Table 1. Resources occupation and estimated power consumption.

Module	FF	LUT	LUTRAM	BRAM	DSP	Power Consumption (mW)
Static Partition	20568	17140	2382	7	1	2901
Grayscale Filter	344	570	160	0	0	1
Sobel Filter	807	1157	160	3	0	7
AES-256	1077	1452	0	0	0	12
SPHINCS$^+$	4405	4210	219	4	0	21

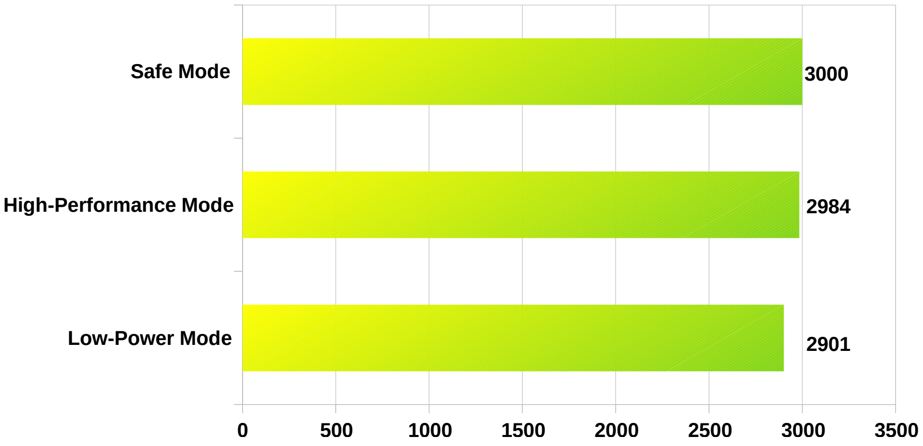

Figure 8. Estimated power consumption for the platform execution modes (mW).

3.2. Platform Reliability

To evaluate the mitigation efficiency, we measured the reliability of the system when injecting faults in the configuration memory at a period of 100 ms, namely one random fault injected every 100 ms. This period has been chosen because the mean time required by the scrubber to perform a complete verification of the FPGA used during these tests is about 50 ms. We allowed time for the scrubber to detect the fault and to perform all the steps for the correction, if possible, including the read of the essential bits information and the fault repair. A system failure occurs if the scrubber does not manage to repair the fault in time, i.e., before the faulty bit is used by the design and causes a computation error or a freeze of the system.

The reliability is computed in four different configurations, or modes. The first mode is the basic mode (B_MODE) in which no mitigation mechanism is enabled. The second mode is the Safe mode (S_MODE) in which both the scrubber and the TMR are enabled. The third mode is the TMR mode (T_MODE) in which only the TMR is enabled. The last mode is the High-Performance mode (P_MODE) in which only the scrubber is enabled.

For every mode, the injection period and the startup time are the same, and fault injection begins just after the application processor startup. Each mode is run until a system or computation failure occurs and is repeated 400 times. The mode, the period and the random seed are sent by the PC through the serial link at the beginning of the run. At the end, the monitoring processor gets access to the watchdog timer and transmits the elapsed time before failure and the number of injected faults to the PC. The results are reported in Figure 9.

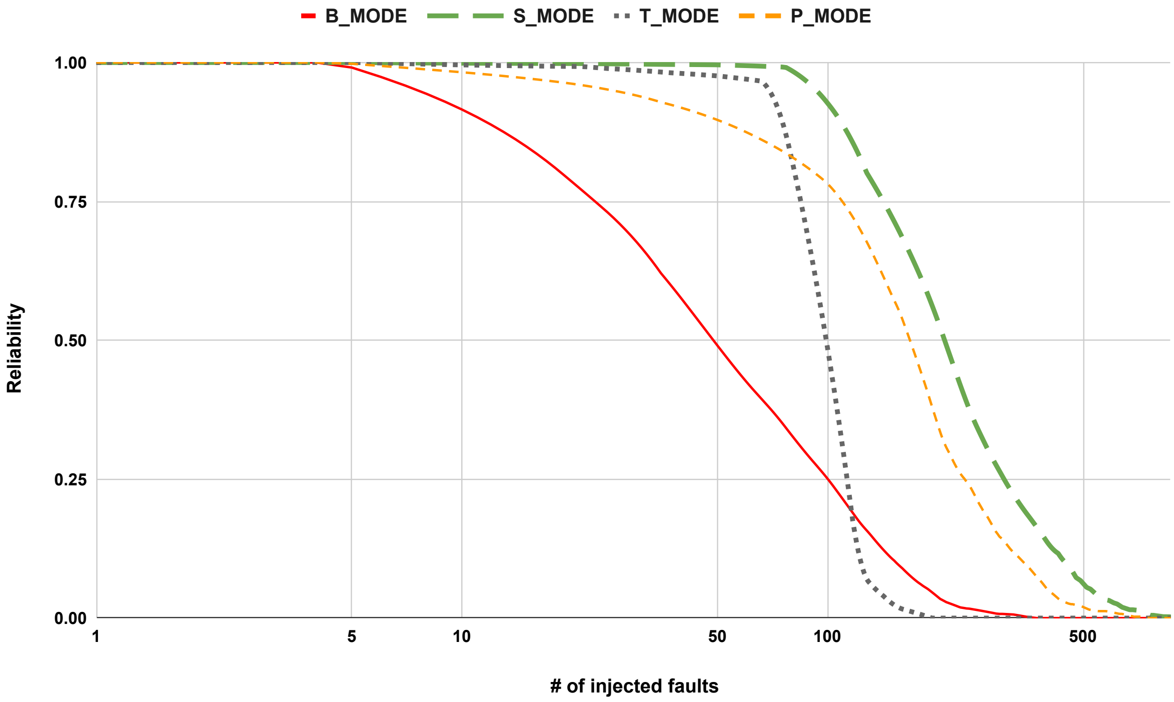

Figure 9. System reliability at 100 ms injection period.

We can observe that, as expected, the different mechanisms improve the overall system reliability compared with the Basic mode (B_MODE). The best mode is the Safe mode (S_MODE) followed by the High-Performance mode (P_MODE). The TMR mode (T_MODE) does not provide any notable interest after the number of injected faults goes beyond a hundred, but coupled with the scrubbing, the performance reduction of the S_MODE is compensated by a significant reliability improvement.

4. Discussion

In this article, we described a platform dedicated to nano-satellite system including fault mitigation mechanisms to deal with SEU due to space radiation, and dynamic reconfiguration capabilities to adapt itself to the level of these radiations. The proposed platform offers a balance between performance and safety when running computationally intensive applications such as post-quantum cryptography. The computation part of the application is deployed on TMR partitions designed to support the mode switching in a flexible way.

This flexibility has a cost in term of hardware resources. It puts pressure on the place and route process but it permits embedding a smaller device, thus a cheaper device, in the nano-satellite payload board without sacrificing functionality. In addition, a smaller device implies a lower power consumption. In order to make the Low-Power mode more interesting, it would be useful to apply software procedures like adjusting the energy-saving settings of the processor when idle to improve the overall system power consumption. Figure 8 also shows that switching from High-Performance mode to Safe mode, and vice-versa, is not detrimental. It means that no extra power budget would be required in case a mode is run more than the other.

Regarding the system reliability, the proposed platform provides an architecture that allows performing early tests of the fault mitigation mechanisms. In our case, fault injection during testbenches has been performed at a period of 100 ms. This period is very pessimistic and has been chosen to speed up the test runs. In real cases, fault occurrence is far more less important. In [19], SEU count per day at 705 km altitude was measured between 170 to 255, with an occasional peak at 1188. Another measure in a Low Earth Orbit of 686 km indicates an average SEU count per day of 100 [20]. We can also notice that using a 16 nm process, Ultrascale+ devices are less sensitive to radiation than previous FPGA generations. This better robustness does not mean that the switch from the High-Performance mode to the Safe mode can be performed at any time. Regarding the radiation level measured by the SEM Controller, if this level is low, the reconfiguration can be done safely. Otherwise, additional mitigation measures must be taken such as checking the reconfigured partition using the DPR to perform a readback of the configuration memory.

When in Safe mode, the choice has been made to not dynamically reconfigure the two filters using the DPR, and to keep the AES IP in place. This choice has been made because, in this case, with the board being assaulted by radiation, it would have been hazardous to perform a dynamic reconfiguration of the FPGA. When in High-Performance mode, regarding the radiation level measured by the SEM, adding new functionalities to the platform and performing a check of the dynamically reconfigured IP could be considered. The system reliability measurement of these different configurations has been depicted in Figure 9. It shows that the High-Performance mode using the scrubber only improves the reliability by x4, whereas the Safe mode, in which the TMR feature is added, improves this reliability by x4.4.

In future work, solutions to enhance platform reliability will be explored. These solutions include features such as repeats of computation failure [21]. Instead of considering that the system has failed and that we need to reboot it, we allow time for the scrubber to perform another round of checking. If, after this round, the fault is still present, we perform a safe recovery of the system. With regards to the fault injection period, it has been setup to keep the same injection scheme for every test run (one fault every 100 ms). For a more realistic simulation, a random distribution of the fault injection will be used in the future platform. Another issue that should be leveraged is the necessity to protect the SEM module against SEU, which could lead in an accumulation of soft errors in the design and the occurence of a computation or system failure. In our case, the monitoring processor will reset the system and a fallback configuration will be loaded from the Flash memory in order to restore the design. The triplication of the logic used by this module [21] is a solution envisioned the future work , as well as the protection of the internal memory of the FPGA using built-in EDAC mechanisms.

Finally, taking advantage of the partial reconfiguration, it could be possible to modify on-the-fly the distribution of the reconfigurable modules over the floorplan [21,22]. This solution would avoid failure in series due to a gathering of the TMR modules in the same configuration memory area. It would be related to the ability to safely perform a dynamic reconfiguration. In this way, the use of Flash memory storage for the reconfigurable bitstream would be necessary and fault prediction mechanisms such as neural networks coupled with a precise model of the radiation at Low Earth Orbit could be considered in order to prevent possible reconfiguration issues.

Author Contributions: Conceptualization, L.G., A.U. and A.K.; methodology, L.G., A.U. and Q.B.; software, L.G. and Q.B.; validation L.G. and A.U.; formal analysis, L.G. and A.U.; investigation L.G., resources, A.U.; data curation, L.G.; writing—original draft preparation, L.G.; writing—review and editing, L.G. and A.U.; visualization, L.G.; supervision, A.U.; project administration, A.U. and E.A.; funding acquisition, A.U. and A.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been funded by Innosuisse under the project "DyReCte: Dynamic Reconfigurable Cryptoengine". Grant number: 32291.1 IP-ICT.

Data Availability Statement: Data used to compute the reliability of the platform are available at this address: https://drive.switch.ch/index.php/s/l3poJGr0rIOc05N (accessed on 2 September 2021).

Conflicts of Interest: The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

FPGA	Field Programmable Gate Array
SEU	Single Event Upset
TMR	Triple Modular Redundancy
RP	Reconfigurable Partition
DPR	Dynamic and Partial Reconfiguration
MPSoC	Multi-Processor System-on-Chip
SEE	Single Event Effects
MCC	Mission Control Center
OBC	On-Board Computer
ADCS	Attitude Determination and Control System
IP	Intellectual Property
SEM	Soft-Error Mitigation
LFSR	Linear Feed Shift Register
PRNG	Pseudo-Random Number Generator
EDAC	Error Detection And Correction
MDPI	Multidisciplinary Digital Publishing Institute

References

1. SpaceX. StarLink Website. 2021. Available online: https://www.starlink.com/ (accessed on 1 September 2021).
2. Astrocast. Astrocast Website. 2021. Available online: https://www.astrocast.com/ (accessed on 1 September 2021).
3. Hiber. Hiber Website. 2021. Available online: https://hiber.global/ (accessed on 1 September 2021).
4. Kleos. Kleos Website. 2021. Available online: https://kleos.space/ (accessed on 1 September 2021).
5. Spire. Spire Website. 2021. Available online: https://www.spire.com/en (accessed on 1 September 2021).
6. Xilinx. Device Reliability Report—Second Half 2019. 2020. Available online: https://www.xilinx.com/support/documentation/user_guides/ug116.pdf (accessed on 1 September 2021).
7. Van Harten, L.; Jordans, R.; Pourshaghaghi, H. Necessity of Fault Tolerance Techniques in Xilinx Kintex 7 FPGA Devices for Space Missions: A Case Study. In Proceedings of the 2017 Euromicro Conference on Digital System Design (DSD), Vienna, Austria, 30 August–1 September 2017; pp. 299–306.
8. Katz, R.; Barto, R.; McKerracher, P.; Carkhuff, B.; Koga, R. SEU hardening of field programmable gate arrays (FPGAs) for space applications and device characterization. *IEEE Trans. Nucl. Sci.* **1994**, *41*, 2179–2186. [CrossRef]
9. Carmichael, C. Xilinx XAPP197: Triple Module Redundancy Design Techniques for Virtex FPGAs. 2006. Available online: https://www.xilinx.com/support/documentation/application_notes/xapp197.pdf (accessed on 1 September 2021).
10. Kumar, M.; Digdarsini, D.; Misra, N.; Ram, T.V.S. SEU mitigation of rad-tolerant Xilinx FPGA using external scrubbing for geostationary mission. In Proceedings of the 2016 IEEE Annual India Conference (INDICON), Bangalore, India, 16–18 December 2016; pp. 1–6.
11. Xilinx. Xilinx PG187: UltraScale Architecture Soft Error Mitigation Controller v3.1. 2019. Available online: https://www.xilinx.com/support/documentation/ip_documentation/sem_ultra/v3_1/pg187-ultrascale-sem.pdf (accessed on 1 September 2021).
12. Upegui, A.; Izui, J.; Curchod, G. Fault mitigation by means of dynamic partial reconfiguration of Virtex-5 FPGAs. In Proceedings of the 2012 International Conference on Reconfigurable Computing and FPGAs, Cancun, Mexico, 5–7 December 2012; pp. 1–6.
13. Wilson, C.; Sabogal, S.; George, A.; Gordon-Ross, A. Hybrid, adaptive, and reconfigurable fault tolerance. In Proceedings of the 2017 IEEE Aerospace Conference, Big Sky, MT, USA, 4–11 March 2017; pp. 1–11.
14. University, C.P.S. CubeSat Website. 2021. Available online: http://www.cubesat.org/ (accessed on 1 September 2021).
15. FreeRTOS. FreeRTOS Website. 2021. Available online: https://www.freertos.org/ (accessed on 1 September 2021).
16. Xilinx. RapidWright—Latest Release Download Page. 2019. Available online: https://github.com/Xilinx/RapidWright/releases/latest (accessed on 1 September 2021).

17. Bernstein, D.J.; Hülsing, A.; Kölbl, S.; Niederhagen, R.; Rijneveld, J.; Schwabe, P. The SPHINCS+ Signature Framework. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, London, UK, 11–15 Novermber 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 2129–2146. [CrossRef]
18. Berthet, Q.; Upegui, A.; Gantel, L.; Duc, A.; Traverso, G. An Area-Efficient SPHINCS+ Post-Quantum Signature Coprocessor. In Proceedings of the 2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Portland, OR, USA, 17–21 June 2021; pp. 180–187. [CrossRef]
19. Poivey, C.; Barth, J.; LaBel, K.; Gee, G.; Safren, H. In-flight observations of long-term single-event effect (SEE) performance on Orbview-2 solid state recorders (SSR). In Proceedings of the 2003 IEEE Radiation Effects Data Workshop, Monterey, CA, USA, 25–25 July 2003; pp. 102–107. [CrossRef]
20. Bentoutou, Y.; Djaifri, M. Observations of single-event upsets and multiple-bit upsets in random access memories on-board the Algerian satellite. In Proceedings of the 2008 IEEE Nuclear Science Symposium Conference Record, Dresden, Germany, 19–25 October 2008; pp. 2568–2570. [CrossRef]
21. Kibar, O.O.; Mohan, P.; Rech, P.; Mai, K. Evaluating the Impact of Repetition, Redundancy, Scrubbing, and Partitioning on 28-nm FPGA Reliability Through Neutron Testing. *IEEE Trans. Nucl. Sci.* **2019**, *66*, 248–254. [CrossRef]
22. Zhang, J.; Han, T.; Li, Y.; Li, J. Real-Time Redundant Scrubbing (RRS) System for Radiation Protection on SRAM-Based FPGA. In Proceedings of the 2020 5th International Conference on Computer and Communication Systems (ICCCS), Shanghai, China, 22–24 February 2020; pp. 905–911. [CrossRef]