# Proton-induced MBU Effects in Real-time Operating System on Embedded Soft Processor

Corrado De Sio[1], Sarah Azimi[1], Andrea Portaluri[1], Daniele Rizzieri[1], Eleonora Vacca[1], Luca Sterpone[1],
David Merodio Codinachs[2]
[1]*Politecnico di Torino, Turin, Italy*
[2]*European Space Agency (ESA), Noordwijk, The Netherland*
{corrado.desio, sarah.azimi, andrea.portaluri, daniele.rizzieri, eleonora.vacca, luca.sterpone}@polito.it
david.merodio.codinachs@esa.int

*Abstract*— **In this paper, we perform an evaluation of the impact of radiation-induced micro-architectural faults affecting the Microblaze soft-processor running a Real-Time Operating System. Fault injection campaigns with a proton-radiation test fault model are presented.**

***Keywords— MBU, Proton Radiation Test, Real-Time Operating System, SEU, Soft Processor.***

## I. Introduction

Field-Programmable Gate Arrays (FPGAs) featuring high flexibility, combined with high performance and complexity became increasingly important also for space applications. With satellite lifetimes increased far beyond 10 years, hardware reconfigurability in flight has become a demanded requirement [1]. Soft-core processors are one of the cores commonly implemented using the programmable logic of the FPGAs [2]. Among the available solutions, Microblaze is an industry leader in FPGA-based soft processing solutions. The highly flexible architecture and configuration options, make it very suitable for embedded applications due to the few resources required to be implemented on programmable hardware. The increasing task complexity required for embedded systems led to the decrease of bare-metal applications and the migration toward the adoption of Real-Time Operating Systems (RTOSs) which provide an efficient solution for meeting stringent real-time requirements [3].

However, when using a soft microprocessor in mission-critical applications, the reliability issues deriving from the exposure of the devices to ionizing radiation, such as Single Event Upsets (SEUs), should be considered [4][5][6]. Differently from hardwired microprocessors, the netlist of soft microprocessors such as Microblaze is implemented in the programmable hardware using the configuration memory (CRAM) of the FPGA. This memory can be corrupted by SEU [7], leading to the hardware micro-architectural faults which can propagate to the application layer and, in the case of the usage of a microprocessor supporting an operating system, it can lead to catastrophic results, especially in mission-critical applications [8].

Several works elaborate on the software-level techniques for evaluating the sensitivity to Single Event Upsets (SEUs) of the embedded operating system. Commonly, these approaches are based on modifying the original kernel of the embedded operating system or altering either the memory that the OS uses or the parameters of system calls.

In [9], the vulnerability of FreeRTOS has been evaluated through a software-based fault injection methodology that targets the most relevant variables and data structures of the OS. An automatic method for fault injection into program and data memory is presented in [10]. The authors of [11] proposed a detailed analysis and hardening architecture based on lockstep synchronization supporting FreeRTOS. The heavy ion irradiation test presented in [12] targets the SRAM and the special purpose registers of an ARM microcontroller to evaluate the impact of the radiation-induced SEU. However, software application-level methods do not take into account the impact of faults occurring at the architectural level of the soft-core processor on the functionality of the operating system.

Other approaches are based on the simulation of HDL description of microprocessors [13]. The advantage of these methods is the feasibility of injecting upsets into any CPU register and structure at any time however, these methods are time-expensive.

The main contribution of this work is dedicated to performing a detailed evaluation of the impact of radiation-induced architectural faults affecting the application benchmarks running on the FreeRTOS of the Microblaze embedded soft processor. The analysis has been performed through a fault injection campaign while an accurate fault model consisting of different clusters patterns of Multiple Bit Upset (MBU) has been identified through proton radiation performed at Paul Scherrer Institute (PSI) radiation facility. We evaluated the effect of faults during the execution of different software applications on FreeRTOS supported by Microblaze implemented on Zynq-7020 FPGA while we performed a deep investigation on the outcome of the software application.

Please notice that the developed platform is not targeting the software-level fault injections but targeting the hardware faults and their impact on the execution of the software running in the operating system.

## II. Proton Radiation Test-based Fault Model

In order to perform the reliability analysis by fault injection campaigns with an accurate fault model, we have performed a proton radiation test at the Paul Scherrer Institute (PSI) Proton Facility in Switzerland. A Zynq-7020 device has been irradiated with proton beams with energies between 29 and 200 MeV. Table I shows the value of energies and fluxes used during the radiation test experiment. During the experiment, the configuration memory of the device has been continuously monitored through a periodic reading of the content every 5 seconds. The snapshots of the configuration memory content have been then analyzed for detecting the occurrence of Multiple Bit Upsets (MBUs). The flux of the particles has been tuned to keep a few bitflips in configuration memory in each snapshot.

TABLE I. RADIATION TEST CONDITIONS: ENERGY, FLUX AND FLUENCE.

| Energy [MeV] | Flux [$cm^{-2}s^{-1}$] | Fluence [$cm^{-2}$] |
|---|---|---|
| 29.31 | $4.124 \cdot 10^7$ | $9.173 \cdot 10^{10}$ |
| 50.80 | $4.024 \cdot 10^7$ | $6.064 \cdot 10^{10}$ |
| 69.71 | $4.110 \cdot 10^7$ | $2.124 \cdot 10^{10}$ |
| 101.34 | $4.319 \cdot 10^7$ | $2.415 \cdot 10^{10}$ |
| 151.18 | $4.094 \cdot 10^7$ | $1.226 \cdot 10^{10}$ |
| 200 | $4.144 \cdot 10^7$ | $3.942 \cdot 10^{10}$ |

The few bitflips and the large size of the configuration memory (more than $10^8$ bits) allowed us to detect groups of SEUs with a strong correlation both in time and space. Therefore, it has been possible to select a group of bits with a high probability to have occurred as a result of a Single Event Multiple Upsets (SEMUs).

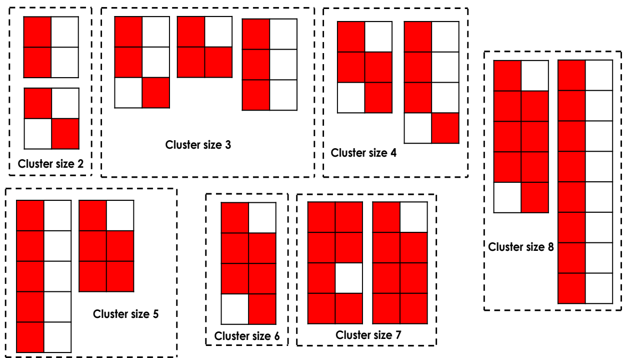

Fig. 1. Detected cluster sizes and shapes during the Zynq-7020 proton test.

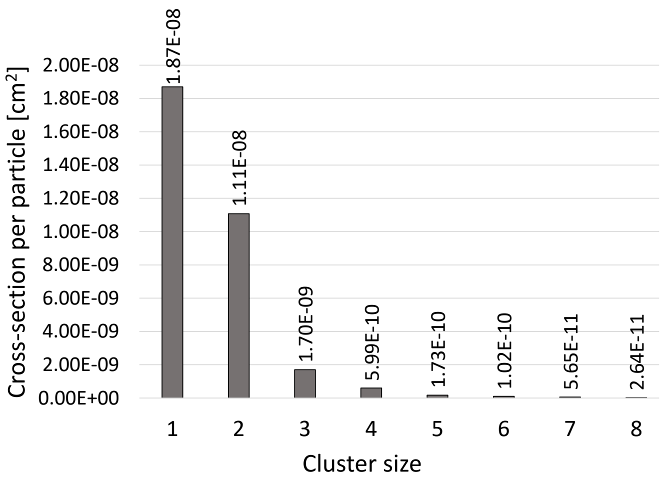

Fig. 2. Cross-section of different cluster sizes.

Figure 1 illustrates the observed pattern for MBUs resulting from the radiation test experiment while Figure 2 shows the cross-section per particle of each cluster size observed during the whole radiation test. It can be noticed how the contribution of SEMUs is not negligible compared to SEUs. Indeed, more than 40% of the detected events have been SEMUs. The proposed cluster of faults has been used as the fault model in the fault injection campaign. In order to obtain an analysis closest as possible to reality, the occurrence rate of the different clusters during the fault emulation has been weighted on the cross-section reported in Figure 2. Since the high occurrence rate of SEMUs, evaluating only an SEU fault model will result in a loose approximation of the observed events.

## III. THE RADIATION ANALYSIS WORKFLOW

The fault model collected through the proton test has been used to perform an accurate radiation analysis on the impact of radiation-induced faults on the Microblaze embedded soft processor porting FreeRTOS through fault injection campaigns.

### A. *The Implemented Hardware/Software Platform*

The current section elaborates on the implemented hardware platform, Microblaze soft-core, supporting FreeRTOS and software benchmarks applications.

#### 1) *Hardware Platform*

The Microblaze embedded soft processor is a Reduced Instruction Set Computer (RISC) optimized for FPGA deployment. It is highly configurable, allowing the selection of a specific set of features required by the design. Therefore, it has been chosen as the platform for supporting FreeRTOS. FreeRTOS, as a deterministic Real-time operating system, allows concurrency among several tasks with different priority levels, supporting a preemption mechanism to switch between task's execution [14].

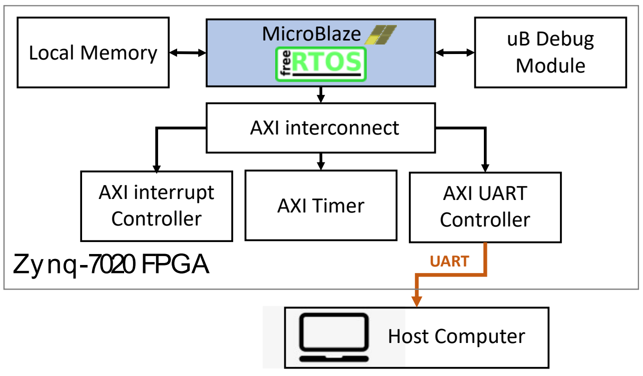

Fig. 3. The implemented hardware platform.

Another important feature that characterizes the Microblaze porting of FreeRTOS is the possibility to instantiate exception handlers to cope with the standard exception conditions defined by Microblaze soft-core [15].

A Xilinx 28 nm CMOS Zynq-7020 FPGA is chosen as a target hardware device.

Figure 3 represents the implemented hardware while Table II reports the device utilization when implementing a Microblaze porting FreeRTOS. As it is can be observed from the table, the implemented design used few resources of the FPGA. Therefore, fault injection campaigns are performed selectively to target only a subset of the whole configuration memory of the FPGA where the circuit is implemented.

TABLE II. RESOURCES UTILIZATION OF THE HARDWARE PLATFORM

| Resources | Used [#] | Available [#] | Usage [%] |
|---|---|---|---|
| **LUTs** | 2,596 | 53,200 | 4.88 |
| **Logic Slices** | 966 | 13,300 | 7.26 |
| **Flip-Flops** | 2,668 | 106,400 | 2.51 |
| **BRAM** | 32 | 140 | 22.86 |

*2) Software Platform*

As a software application suite, a set of software benchmarks have been chosen to run through FreeRTOS, while exploiting its main functionalities. To exploit the capability of FreeRTOS to schedule the execution of different tasks, for each software application, three different tasks (software benchmark) with the same priority have been instantiated. Therefore, three tasks running on FreeRTOS have the same instruction code, however, they are operating on different input data while sharing the processor execution time. The three selected software benchmarks are:

- *matmul*: matrix multiplication between large matrices of integers.
- *matconv*: matrix Convolution between large matrices of integers.
- *dijkstra*: computation of shortest path between nodes in a large graph using the Dijkstra algorithm.

## B. Fault Injection Analysis

Reliability analysis of the applications under test against radiation-induced hardware architectural faults in soft microprocessors has been performed by fault injection campaign. The PyXEL platform has been used as a supporting fault injection framework [16]. PyXEL is a python-based platform easing the execution of FPGA fault injection campaigns. The platform has been instrumented to inject MBU patterns identified during the proton radiation test in the configuration memory of the FPGA. The device under test is connected to the host computer running the PyXEL experiment manager through a serial connection allowing to run the experiments on the platform and collecting results. A timeout mechanism is used to handle the halt and loop of the processor due to the injected faults.

Two fault injection campaigns have been performed. The former has been carried out based on the distribution of SEUs and MBUs represented in Figure 2. In the latter, each detected cluster has been extensively tested in order to estimate the impact of the different MBU clusters on the application failures. As it has been mentioned before, during the fault injection campaigns, only the part of the whole configuration memory implementing the circuit under test is targeted by the fault injection task in order to reduce the injection space to the resources used by the implemented netlist.

## IV. Experimental Analysis and Results

Reliability analysis has been carried out by two fault injection campaigns based on fault models collected through a proton radiation test campaign targeting the Zynq-7020 device. Results have been collected, categorized, and discussed.

## A. Error Classification

As a result of fault injections, different misbehaviors have been observed. Errors are detected by comparison of the outcomes of the fault injection experiments with the outcome of the golden (i.e., without faults) run.

The collected results have been classified into four categories: correct, silent data corruption (SDC), halt, and raising exceptions. They are defined as follows:

- *Correct*: the FreeRTOS succeeded in executing the application and produced an output that matches the golden one.
- *Silent Data Corruption*: the task execution on FreeRTOS terminates but the produced output data does not match with the golden one.
- *Halt*: the FreeRTOS does not complete the task. It can be due to different causes, such as infinite loops and application timeout.
- *Raising Exceptions:* an exception is generated in the FreeRTOS (i.e., at the software level) as a result of a fault affecting Microblaze architecture (i.e., netlist modification due to configuration memory corruption).

Moreover, we have performed a detailed investigation on the cause of each raised exception and classified them as follows:

- *FSL_EXCEPTION*: data bus error exception.
- *UNALIGNED_ACCESS*: attempt to perform unsupported unaligned access to memory.
- *ILLEGAL_OPCODE*: attempt to execute an illegal opcode.
- *AXI_D_EXCEPTION*: data system bus timeout.

Finally, we have reported the benchmark application error rate which is defined as the percentage of results that deviate from the nominal behavior.

## B. Experimental Results

We have performed two fault injection campaigns. In the first campaign, we performed a total of 10,000 fault injections considering the cluster distribution represented in Figure 2. The observed error rate with respect to the cluster distribution is reported in Figure 4. Data show that the three applications have been impacted differently by the fault injections. In particular, the *matconv* has registered the highest number of total errors (including all the four categories) with 2,179 corruptions over 10,000 injections while *matmul* and *dijkstra* collected 1,028 and 1,026 errors, respectively.

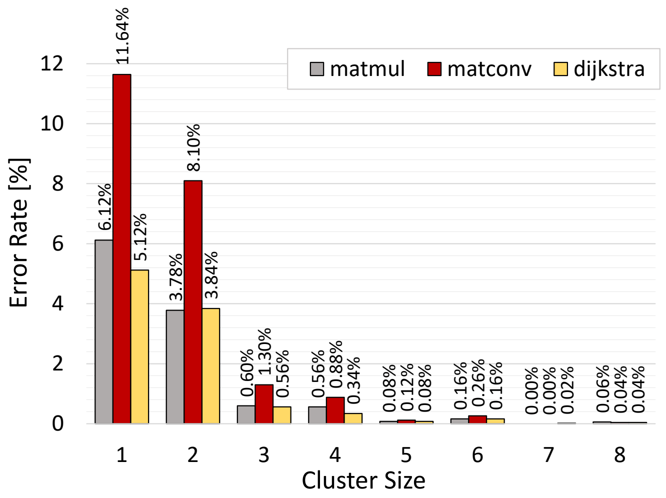

Fig. 4. Distribution of errors for each application over cluster sizes.

For each application, the vast majority of the errors are due to the cluster injection of sizes 1 and 2. Indeed, it is easy to observe that the distribution of errors roughly follows the distribution of the clusters. Table III reports the classification of the observed errors for each application.

TABLE III. ERROR RATE CLASSIFICATION

| Application | SDC [#] | Halt [#] | Exception [#] | Total Errors [#] |
|---|---|---|---|---|
| **matmul** | 227 (19.95%) | 882 (77.50%) | 29 (2.54%) | 1,138 (100%) |
| **matconv** | 1,016 (46.63%) | 1,139 (52.27%) | 24 (1.10%) | 2,179 (100%) |
| **dijkstra** | 119 (11.58%) | 891 (86.67%) | 18 (1.75%) | 1,028 (100%) |

It can be noticed that the highest value always belongs to the *Halt* label, most likely due to the corruption of communication modules within the design.

The second campaign is performed considering 5,000 fault injections for each cluster size. The results are represented in Figure 5.

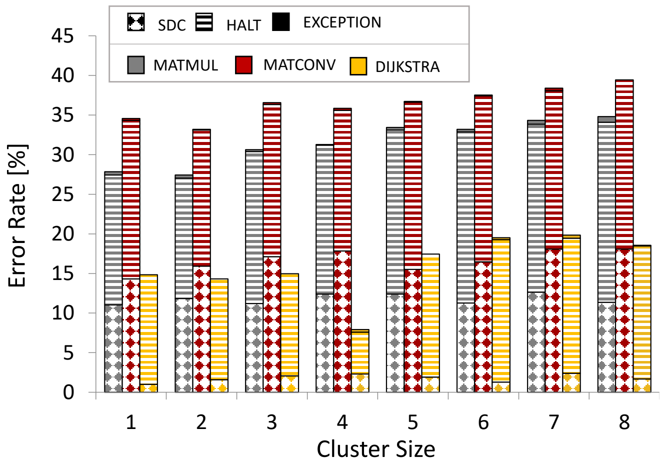

Fig. 5. Error Rate associated with cluster size for each application.

As it can be seen, the cluster size has a marginal effect on the error rate. It may be related to the fact that bits associated with a specific hardware resource are located closely in the configuration memory. If that part of configuration memory is selected as a fault location, the size of the injected cluster will only marginally increase the corruption of the used logic resource.

Finally, a classification of the observed exceptions in the two fault injection campaigns has been performed. Although a negligible value with respect to the other errors, the exceptions are a mechanism that can be used to improve the reliability of soft processors, detecting the occurrence of a hardware fault. The chart in Figure 6 shows the relative frequencies of each exception type.

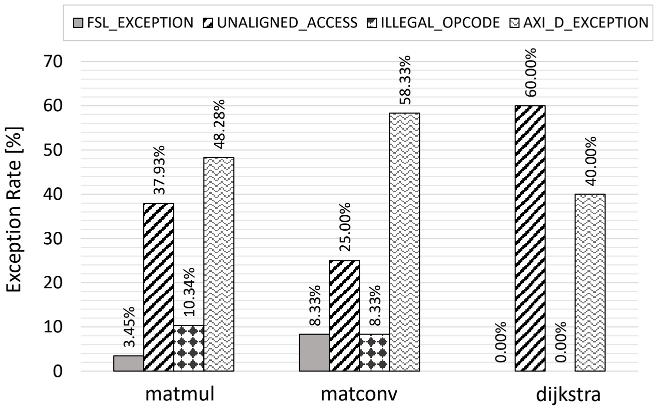

Fig. 6. Relative frequency of the exceptions types resulting from the two fault injection campaigns.

## V. CONCLUSIONS AND FUTURE WORKS

In this paper, the impact of radiation-induced architectural faults identified through a proton radiation test, affecting different applications executing on a Microblaze embedded soft-processor running FreeRTOS has been evaluated. The occurrence and contribution to the error rate of specific MBUs events based on different shapes and sizes have been evaluated in detail. In the future, we plan to consider both software and hardware approaches for mitigating radiation-induced errors on applications running within FreeRTOS.

## REFERENCES

[1] A. Hofmann, et al., "An FPGA based on-board processor platform for space application," 2012 NASA/ESA Conference on Adaptive Hardware and Systems (AHS), pp. 17-22, 2012, DOI: 10.1109/AHS.2012.6268653.

[2] S. Azimi, et al., “Analysis of Single Event Effects on Embedded Processor,” MPDI Electronics, vol 10, 2021, DOI: 10.3390/electronics10243160.

[3] "IEEE Standard for a Real-Time Operating System (RTOS) for Small-Scale Embedded Systems," in IEEE Std 2050-2018, vol., no., pp.1-333, 2018, DOI: 10.1109/IEEESTD.2018.8445674.

[4] De Sio, et al., "SEU Evaluation of Hardened-by-Replication Software in RISC- V Soft Processor," IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), pp. 1-6, 2021, DOI: 10.1109/DFT52944.2021.9568342.

[5] Á. B. de Oliveira et al., "Evaluating Soft Core RISC-V Processor in SRAM-Based FPGA Under Radiation Effects," in IEEE Transactions on Nuclear Science, vol. 67, no. 7, pp. 1503-1510, July 2020, doi: 10.1109/TNS.2020.2995729.

[6] P. Rech, et al., "Reliability Analysis of Operating Systems for Embedded SoC," 2015 15th European Conference on Radiation and Its Effects on Components and Systems (RADECS), 2015, pp. 1-5, doi: 10.1109/RADECS.2015.7365659.

[7] B. Du et al., "Ultrahigh Energy Heavy Ion Test Beam on Xilinx Kintex-7 SRAM-Based FPGA," in IEEE Transactions on Nuclear Science, vol. 66, no. 7, pp. 1813-1819, 2019, doi: 10.1109/TNS.2019.2915207.

[8] P. V. Nekrasov, et al., "Investigation of Single Event Functional Interrupts in Microcontroller with PIC17 Architecture," 2015 15th European Conference on Radiation and Its Effects on Components and Systems (RADECS), pp. 1-4, 2015, DOI: 10.1109/TNS.2019.2915207.

[9] D. Mamone, et al., "On the Analysis of Real-time Operating System Reliability in Embedded Systems," IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), pp. 1-6, 2020, DOI: 10.1109/DFT50435.2020.9250861.

[10] I. O. Loskutov et al., "Investigation of Operating System Influence on Single Event Functional Interrupts Using Fault Injection and Hardware Error Detection in ARM Microcontroller," International Siberian Conference on Control and Communications (SIBCON), pp. 1-4, 2021, DOI: 10.1109/SIBCON50419.2021.9438916.

[11] P. M. Aviles, et al., "Radiation Testing of a Multiprocessor Macrosynchronized Lockstep Architecture With FreeRTOS," in IEEE Transactions on Nuclear Science, vol. 69, no. 3, pp. 462-469, March 2022, doi: 10.1109/TNS.2021.3129164.

[12] I. O. Loskutov, et al., "SEFI cross-section evaluation by fault injection software approach and hardware detection," IEEE 30th International Conference on Microelectronics (MIEL), pp. 251-254, 2017, DOI: 10.1109/MIEL.2017.8190114.

[13] W. Mansour and R. Velazco, "SEU fault-injection in VHDL-based processors: A case study", Journal of Electronic Testing: Theory and Applications (JETTA), vol. 29, no. 1, pp. 87-94, 2013, DOI: 10.1109/LATW.2012.6261258.

[14] FreeRTOS, “Xilinx Microblaze Port” informative webpage. [Online]. Available: https://bit.ly/3r5Y3ph. Accessed on: April 06, 2022.

[15] Xilinx, “MicroBlaze Processor Reference Guide”, UG984 v2021.2, Oct 27, 2021, pp. 80-89. [Online].

[16] L. Bozzoli, et al., "PyXEL: An Integrated Environment for the Analysis of Fault Effects in SRAM-Based FPGA Routing," International Symposium on Rapid System Prototyping (RSP), pp. 70-75, 2018, doi: 10.1109/RSP.2018.8632000.