Multiple Fault Injection Platform for SRAM-based FPGA based on Ground-level Radiation Experiments

Jimmy Tarrillo
Department of Electrical Engineering
Universidad de Ingenieria y Tecnologia - UTEC
Lima, Peru
jtarrillo@utec.edu.pe

Jorge Tonfat, Lucas Tambara, Fernanda Lima Kastensmidt and Ricardo Reis
Instituto de Informatica – PPGC - PGMICRO
Universidade Federal do Rio Grande do Sul - UFRGS
Porto Alegre, Brazil
(jlseclen, latambara, fglima}@inf.ufrgs.br

***Abstract*-SRAM-based FPGAs are attractive to many high reliable applications at ground level due to its high density and configurability. However, due to its high sensitivity to neutron-induced soft errors, the FPGA configuration memory bits may suffer unexpected bit-flips and consequently critical errors may occur. To cope with this problem, authors have proposed several mitigation techniques, which must be verified under the presence of faults. Since ground-level radiation experiments are very costly, fault injection is a suitable method to verify mitigation techniques in early stages of development. In this work, we present a fault injector platform implemented in a FPGA commercial board able to inject multiple bit-flips in the configuration memory bits of SRAM-based FPGAs based on a fault database collected on radiation experiments. We show the accuracy of our proposed fault injection campaign compared to radiation test results. We compare the soft error rate of three designs under the accumulation of multiple faults.**

Keywords— fault injection; single event upsets effects; SRAM-based FPGA

I. INTRODUCTION

Field Programmable Gate Arrays (FPGAs) are highly appropriate to space applications as they present fast time-to-market, high flexibility and reprogrammability. SRAM-based FPGAs store its configuration memory bits into volatile SRAM cells, and consequently radiation effects can generate single or multiple bit-flips in the configuration bitstream. Such single event upsets (SEUs) or multiple bit upsets (MBUs) can induce functional errors in the implemented design. In order to tolerate these errors, many techniques were proposed in the literature. However, it is necessary to validate the efficiency of the selected technique closest to the real effect as possible, but also considering the controllability, observability and cost of the experiment.

Fault injection by emulation is an important tool to predict in early stages of the design phase the susceptibility of the design under upsets. Emulation of SEUs and MBUs by flipping the configuration bits on an FPGA is an attractive technique to evaluate the behavior of a design before it is working in radiation environment. In addition, it is possible in FPGAs that fault injection has partial reconfiguration capabilities, which reduces the time to inject upsets. The main goal of this approach relies on the fact that it allows fast injection campaigns, once the design under test (DUT) executes at speed prototyped in the board, instead of using simulation. Moreover, comparing to radiation tests when the FPGA is placed under particles accelerators, the amount of injected faults per unit of time (upset rate) is much higher, since a bit-flip is directly injected in the memory cell, not depending on the possibility of a particle flips or not a bit in the configuration memory. The control of the test is also superior comparing to a radiation test, since a precise location is flipped (a known bit), which allows the user reproducing a real radiation test.

The injection of upsets in the SRAM-based FPGA bitstream can be implemented in hardware by the use of external or internal programmable port of the FPGA. The most common and fastest interface is the internal one called internal configuration access port (ICAP) [1], which provides some advantages such as the possibility to reconfigure frame by frame without the necessity of using input/output pins. The ICAP can be controlled by SEU controller macro [2] and an embedded soft-core as PicoBlaze, or by a specific control design developed by the user. SEU can be injected in the bitstream randomly, sequential (every configuration bit or configuration control registers is flipped in sequential order), or user defined (provided via text file).

Many fault injection platforms are available to inject SEUs in FPGAs. FLIPPER [3] is based on a mother control board and a design under test (DUT) board, and uses the external configuration memory port to flip the configuration bits. FT-UNSHADES [4] uses the ICAP to inject single faults in the bitstream and user's flip-flops of the design. Other example of fault injector is presented in [5], which is also based on ICAP interface and the injection of SEU in the configuration memory bits. All these platforms inject one fault per time in the bitstream and analyze its effect in the design outputs. When multiple bit upsets (MBU) need to be considered, the challenge is how to distribute the multiple faults in the configuration memory bits.

In this work, we propose a fault injector platform able to emulate SEU and MBU and their accumulation effects in the configuration memory bits of SRAM-based FPGA. This approach is fast (thousands of upsets can be injected in few minutes), it presents a good control of the experiment and it is inexpensive compared with radiation experiments. In our proposal, the bit-flip locations are taken from an external database bank where upset bits location were stored based on previous experiments in neutron radiation test from ISIS [6] neutron facility.

Comparisons between the fault injection results and real neutron radiation experiments are shown. The obtained soft error rate results are very similar, which shows the efficient of using the proposed platform to predict radiation test experiments. The main differences of the available platforms [3, 4, 5] and the one presented here is that the proposed platform aims to inject multiple faults in order to repeat neutron radiation test experiments based on the observed and collected flux of particles and bit-flips. In this way, it is possible to verify and test designs in the laboratory before radiation ground testing having a better prediction of the mitigation technique efficient to cope with multiple and accumulated faults.

II. Proposed Multiple Fault Injection Platform

The proposed fault injection platform is based on SRAM-based FPGA Virtex-5, and is composed by elements out of the FPGA and modules implemented into the FPGA as presented in Fig. 1. The internal configuration access port (ICAP) is able to modify the configuration bits allowing the emulation of a SEU. The ICAP is able to access to the configuration memory through each frame address. Frames are the smallest addressable segments of the FPGA configuration memory bits and are composed by 41 words of 32 bits (1312 bits) in case of Virtex-5.

A PicoBlaze 8-bit soft processor [7] is used to manage all components of the platform. ICAP control module receives the information from PicoBlaze and controls the ICAP to read and write into the configuration memory. The frame buffer is used to store temporally the read frame and modify it. There are two different sources of fault injection locations:

- SEU location database bank: After neutron accelerated experiments in ISIS facilities, we collected the location of bit-flips induced by radiation and stored into an external on-board flash memory. In this way, results obtained from such bank will replicate in trust way the effects of radiation in the design under test. However, SEU location database bank can be filled with a customized SEU distribution.
- LFSR: In case of SEU locations from the database bank are not enough, an on-chip random generator implemented by a linear feedback shift register (LFSR), can be used as source of fault injection. This option could be useful when the circuit under test is bounded to a small area of the FPGA and the available locations are not enough.

As any malfunction in the fault injector may compromise the experiment control, the on-chip elements have a specific placement (forbidden area) under which is not allowed any bit-flip injection. Then, the circuit under test can be placed in the remaining available area (DUT area) of the configurable matrix. Clock lines and connection lines between the control circuit that send the data to the external PC must be taken into account to avoid faults and consequently the loss of connection of the system.

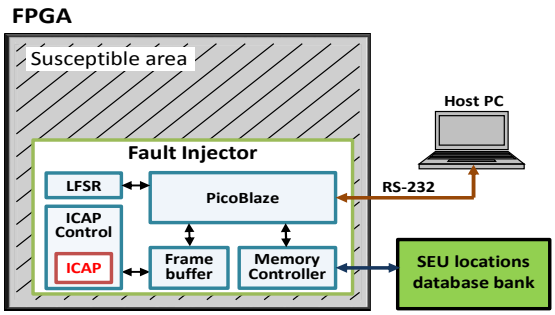


Fig. 1. Fault injector platform scheme.

SEUs are injected consecutively one by one until the user requirements are achieved or if the total of SEU locations from the external bank is reached. As shown in Fig. 2, at the beginning, the PicoBlaze receives the setup of the user: SEU rate, forbidden area (slices), bit-flip location source, and maximum number of injected faults. First, the control request for a SEU location, and if the obtained location belongs to the forbidden area, the request is repeated again. After that, the flipped bit location is sent to the external PC.

The injector was implemented into XC5VLX50T on Genesys Digilent board and in XC5VLX110T Virtex-5 FPGA on ML505 Evaluation Platform board. Synthesis result of the fault injector modules is detailed in Table I.

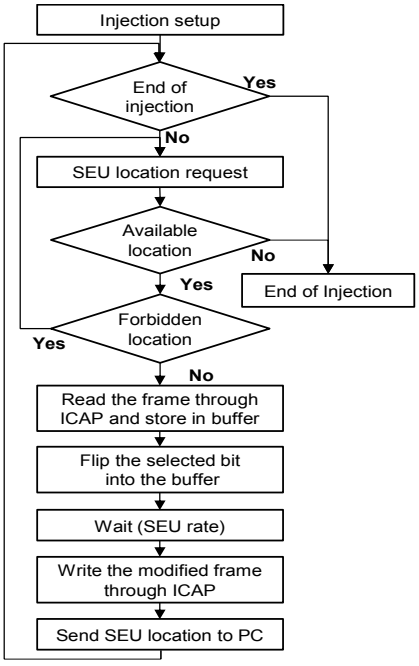


Fig. 2. Flow diagram of proposed fault injector.

TABLE I. Resource Utilization of the Proposed Fault Injector

	LUTs	Registers	BRAMs
PicoBlaze	147	76	1
Memory Controller	86	68	0
LFSR, ICAP controller, frame buffer	705	417	1
Total	938	561	2

III. Fault Location Sources

In this section we explain and compare in detail the two possible sources of fault injection locations.

A. Linear feedback shift register (LFSR)

A pseudo-random generator circuit can be used aiming at supplying random addresses to the injection control and then tuned to simulate the behavior of a real neutron test. Several LFSR structures and seeds were implemented and tested to generate a good random dispersion and to obtain the effects nearest to the produced ones for the radiation experiments. The selected one is a 25-bits LSFR and is based on the frame address structure [1]. Each frame address is divided into 5 main parts: type (4 bits, in our case always "0001"), top/bottom (1 bit), row (5 bits), major address (8 bits) and minor address (7 bits). Then, the LFSR is composed by 4 sub-groups of smaller LFSRs as shown in Fig. 3. The selection of the bit position inside the frame is selected by the 11 firsts bits of the LFSR. The forbidden frame addresses (frames of injector platform or defined by the user) and nonexistent frame addresses are filtered by the injector control.

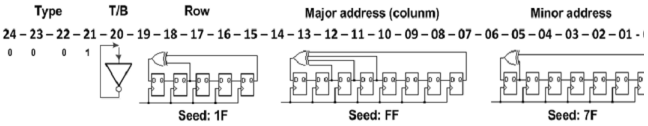


Fig. 3. LFSR structure used to produce random multiple fault injection in the FPGA bitstream.

B. SEU Location Database

A database composed of multiple and accumulated faults in Virtex-5 FPGA were built from previous radiation experiments. The database has the radiation data of two Virtex-5 devices: XC5VLX50T and XC5VLX110T irradiated with a neutron spectrum that resemble the atmospheric one in the ISIS facilities of Rutherford Appleton Laboratory (Didcot, United Kingdom). The flux was about 4.3×10^4 neutrons/s/cm^2. During the experiments, SEUs in the bitstream were detected using a readback procedure. The soft error rate (SER) information was stored as text files that include the time and the bit position in the bitstream of each SEU for both devices.

Based on our knowledge of the FPGA bitstream, we can precisely determine the frame address and bit position of each SEU registered during the experiment as shown in Fig. 4. The readback file (readback.bin) obtained during the radiation experiment contains the configuration bit values at the moment of the readback. When compared with a "golden" readback (before radiation experiment) and considering the mask file (which indicates the position of dynamic configuration bits) we can locate the flipped bits in the static configuration memory. Dynamic configuration bits are composed by RAM blocks distributed RAM and shift register elements, then, it is not possible to detect upset bits in such resources using this procedure. Since the readback is related to the floorplan, we also can locate the position of the bit-flip on the FPGA floorplan. Finally, using the frame address structure available in [1] we extract the frame address and bit position of the bit-flip. These informations are necessary to write into the configuration memory through the ICAP.

In our experiments, more than 1,000 SEUs were identified. This information is stored in the platform in an external flash memory. In the case of the Genesys board, a flash memory of 256Mbit (organized as 16-bit by 16Mbytes) is used to store the SEO locations. We used three memory addresses to store the information of each SEU. The first two positions store the frame address and the last position store the bit position inside the frame. So, up to 5 million SEUs can be stored in this memory.

C. LFSR and SEU location database comparison

Since we are interested in emulating the effects of radiation effects in SRAM-based FPGAs, we analyze the SEUs distribution and its effects in some circuit under test exposed to neutron irradiation. The used FPGAs XC5VLX110T on ML505 board, and XC5VLX50T on Genesys board. Fig. 5 plots the SEUs distribution obtained from radiation experiment with the fault injection using the LFSR and SEU database, which are the same obtained from neutron radiation experiments

Each bar in the plots represents the number of accumulated SEUs per frame in configuration bits (no BRAMs data are considered). The total number of accumulated SEUs for each plot is also shown. Neutrons experiments commonly show one bit-flip by frame, when the injector using LSFR show values between 4 and 8 per frame. Although the fault distribution obtained from the LFSR does not follow the same of radiation experiments, the LFSR could be used when the circuit under test occupies a small area of FPGA.

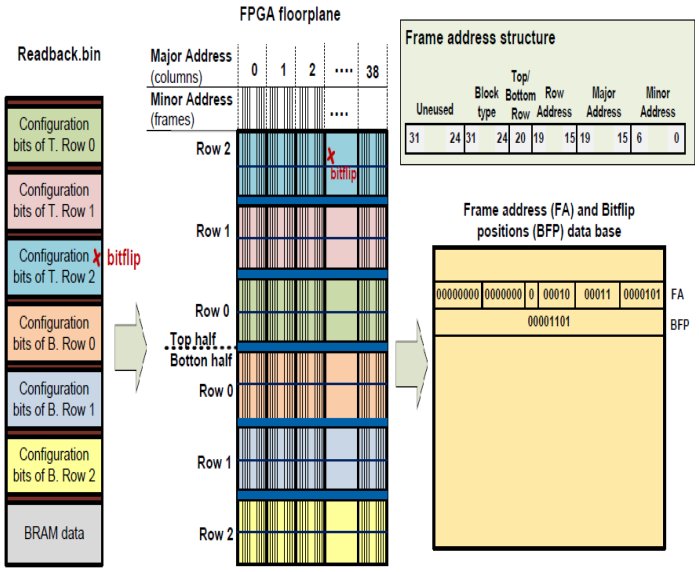
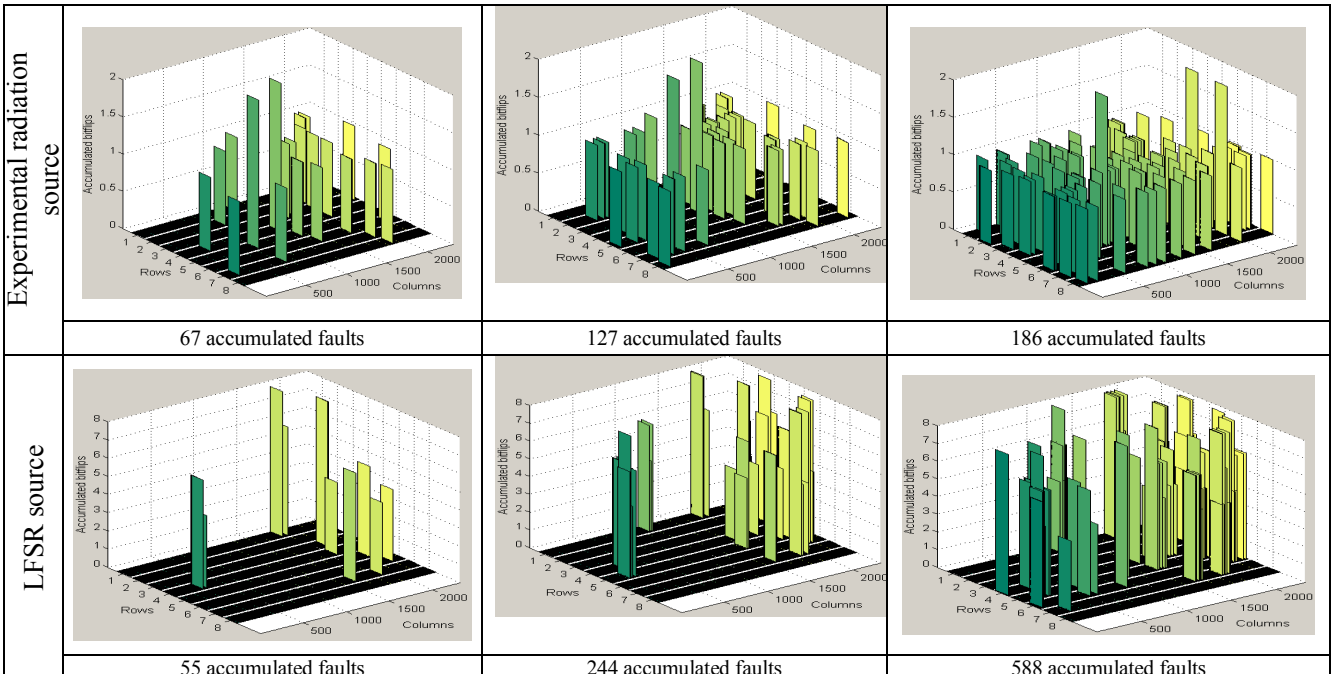


Fig. 4. Getting the bit-flip locations in Virtex-5 FPGAs



IV. Experimental Results

In order to validate the proposed fault injector, we tested by radiation and by fault injection three designs protected by fault tolerant techniques. Radiation experiments were performed using the neutron spectrum available at ISIS laboratory (neutron flux of 3.7×10^4 neutrons/s/cm^2), which resemble the atmospheric one [6]. The first case-study design is a diverse triple modular redundant matrix multiplication, the second one is a multiple redundant adder's chain, and the last case-study design is a multiple redundant soft-processor miniMIPS. The fault injections were performed using the LFSR (first case-study) and the SEU location database bank.

A. Case-study design I: a Diverse triple modular redundant modules (DTMR) matrix multiplication core

DTMR technique was presented in [8] as a variation of the traditional triple modular redundancy (TMR), in which the original circuit is triplicated by adding two extra copies of the original circuit. The three copies of the TMR approach operate in parallel and the output of these copies are delivered to a majority voter. If an error happens in one of the copies, two of them will continue to operate properly and the majority voter can correctly mask the output of the faulty module [9]. In case of DTMR, each redundant copy is implemented in a distinct way using different replicas and algorithms.

In order to evaluate the capacity to replicate the radiation effects of the proposed fault injector platform using as sources the LFSR circuit, fault injection campaigns and neutron radiation experiments were performed in an 8x8 matrix multiplication operation protected by TMR and DTMR without using scrubbing technique. The objective is to observe how the SEU accumulation impacts the effects in the DTMR scheme versus the standard TMR composed of minMIPS processors. Three different designs for the matrix multiplication algorithm were implemented: one using a finite state machine (FSM), one using control and datapath blocks, and one using a miniMIPS processor running the algorithm. The whole DTMR architecture was prototyped in XC5VLX110T. Fig. 6 shows the scheme of the DUT and Table II shows the area utilization of the DTMR and the injector controller into Virtex-5.

Fault injection results are compared and shown in Fig. 7. These results were obtained after 26 (TMR) and 23 (DTMR) runs during radiation experiments, and after 500 runs for both techniques during fault injection campaigns. In the case of neutron experiments, DTMR needs 190 accumulated SEUs in the configuration memory bits to have an error at the design output, against 86 accumulated upsets in the TMR case. This shows that it is necessary 2.21 times more accumulated SEUs to provoke an error in the DTMR design compared to the standard TMR. In the case of injecting faults by LSFR selection, DTMR needs 391 accumulated faults in the configuration memory bits to have an error at the design output, against 158 accumulated upsets in the TMR scheme.

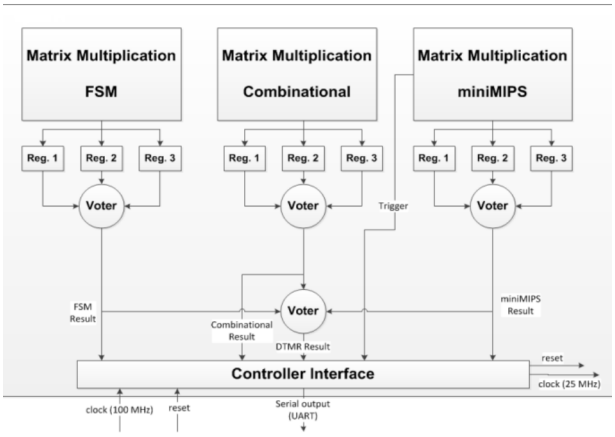


Figure 6. Scheme of CUT: diverse triple modular redundant modules [6].

TABLE II. RESOURCE UTILIZATION OF DTMR CUT FOR VIRTEX-5 LX110T

	LUTs		Registers		BRAMs		DSP48E
	(#)	(%)	(#)	(%)	(#)	(%)	(#)
miniMIPS	3395	4.91	1501	2.17	15	0.1	4
FSM	1129	1.63	702	1.02	0	0	0
Combinational	5113	7.4	642	0.93	0	0	0
Controller Interface	311	0.45	207	0.3	0	0	0
Total	9948	14.39	3052	4.42	15	0.1	4

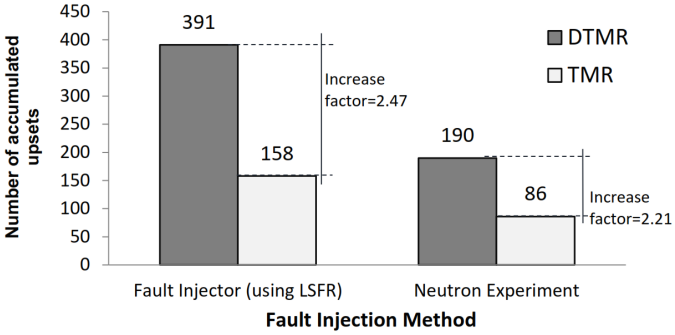


Figure 7. Comparison between neutron radiation experiment and fault injection using LFSR in DTMR and TMR fault tolerant schemes.

Note that both fault injection campaigns show similar proportions from DTMR and TMR: 2.21 and 2.47 times more accumulated upsets are needed in the DTMR design compared to TMR. The difference is less than 12 %.

B. *Case-study design II and III: a Multiple modular redundancy (nMR) adder and processor core*

In [10], authors proposed the use of multiple modular redundancies ($n \geq 3$, where n represents the number of redundancy modules) to improve the increase the number of accumulated faults in the system, and the use of a Self-adaptive voter (SAv) to select the correct output. As the SEUs occur and build up, one by one the modules begin to fail until two modules without failure remains. Fig. 8 represents a diagram of nMR for 7 redundant modules.

Two different study cases, an adders chain and a miniMIPS softcore processor, were tested by radiation experiments and by the proposed fault injection platform using the SEU location database bank. In both cases, a test control based in PicoBlaze processor was used to control the experiment. All the systems were implemented into a Virtex-5 V5LX50T FPGA.

1) *Design II: 7-MR Adders chain*

The DUT is composed by 7 redundancy modules, where each one is composed by 190 16-bit adders connected in cascade (only standard configuration logic blocks were used), a self-adapted voter (SAv) similar presented in [10] and a test control circuit. All adder chains are sources by the same pattern generator block implemented by a 32-bit counter, which is initialized in a specific value. The 16-bit most significant bits compose the first adder operator, and the rest 16-bit less significant bits compose the second one. In this way, the adder chain block result is deterministic and we can expect the correct result ('golden' result) in a specific time. Each adder chain block set a flag 'DONE' when the output is the golden result, and it is used by the test control to detect a no masked error. The fault-free output (FFO), error status flags (ESF), and the non-masked fault flag (NMF) are sent to the test control and to an external PC to know the state of the experiment. When only 2 free-fault modules remain, the FPGA is reconfigured and a new run is performed. Fig. 9 presents a block diagram of the design and Table III shows the resources utilization by the design when synthetized.

Fig. 10 shows the comparison of the number of faults accumulated in the configuration memory bits required to provoke an erroneous output in a module (faulty module). For radiation results 6, 5, 5, 4, 3 and 2 runs were performed to obtain 1, 2, 3, 4, 5 and 6 faulty modules respectively. In contrast, 25 runs were performed in all cases when the proposed fault injector was used. The lower number of runs in radiation experiments is due to the complexity and the SEU rate in this kind of experiments. As shown, more than 100 accumulated faults induced by radiation are needed to get 5 faulty modules (7MR system), which represent more around 500 minutes per run. According to Fig. 10, fault injection results are closed to radiation ones.

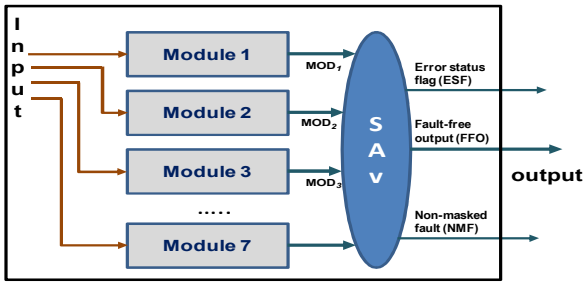


Figure 8. Scheme of CUT: 7MR of adders chain [7].

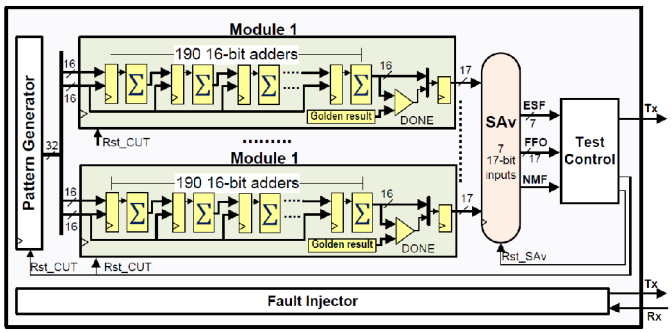


Figure 9. Block diagram of 7MR of adders chain circuit.

TABLE III. RESOURCE UTILIZATION OF THE DESIGN FOR TESTING A 7MR ADDERS CHAIN SYSTEM

	LUTs		Registers		BRAMs	
	(#)	(%)	(#)	(%)	(#)	(%)
Fault injector	938	3.26	561	1.95	2	0
Each module (7 times)	3044	10.57	3072	10.67	0	0
SAv voter	247	0.86	144	0.50	0	0
Pattern generator	15	0.05	16	0.06	0	0
Test control	152	0.53	121	0.42	0	0
Total	**22660**	**78.68**	**22346**	**77.59**	**2**	**3.33**

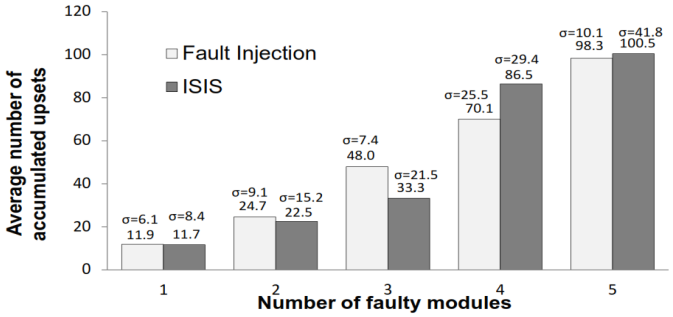


Figure 10. Comparison between fault injection and neutron radiation experiments results of adder chain: average number of faults needed to provoke up 5 faulty modules and its standard deviation (σ).

2) *Design III: 6-MR miniMIPS*

Similar to the previous case, a 32-bits miniMIPS soft processor was replicated by 6 times (6MR), all of them running in tandem a 6x6 matrix multiplication, and implemented in the same FPGA. All processors were synchronized with the same clock and reset signals, then, no pattern generator was used. In radiation test the matrix multiplication run 5 times, while run 17 times in fault injection experiments. Table IV shows the used resources. In this case, the DUT uses 20 BRAMS, which are around 5.9 times (± 18%) more susceptible to SEUs that memory configuration bits in Virtex-5 FPGAs [11]. As explained in Section III, our procedure does not detect faults in BRAM blocks, and consequently we expect to detect by readback around 5.9 times less accumulated faults.

Fig. 11 shows the comparison of the number of faults accumulated in the configuration memory bits required to provoke an erroneous output in a module (faulty module) when tested by radiation (same conditions of previous case) and by the proposed fault injection platform. Notice it was required a slightly higher number of injected faults compared to radiation in order to upset the same number of modules. This behavior was predicted since the RAM blocks (BRAM) of each processor was not protected and radiation may induce faults in such elements, while the fault injector does not inject faults in the BRAM. However, the discrepancy between radiation and fault injection in this case is explained by the BRAM and susceptibility [11], as the current version of the fault injector does not inject faults in the BRAMs.

V. CONCLUSIONS

This work presented a fault injector platform to evaluate accumulated MBU effects in Virtex-5 FPGA. The platform allows using the bit-flip positions generated by an LFSR circuit or taken from a database composed by pre-collected real bit-flips location detected from previous neutron accelerated experiments in ISIS facilities. The flipped bits distribution of real radiation test and fault injector were shown and analyzed. Also, the effects of accumulation SEUs in two different designs using real radiation test and fault injection were tested. Results show the real capability of the platform proposed to prevent the effects of radiation in FPGA designs in shorter time than radiation experiments.

ACKNOWLEDGEMENTS

This work was supported by Brazilian National Council of Scientific and Technologic Development, CNPq.

TABLE IV. RESOURCE UTILIZATION OF THE DESIGN FOR TESTING A 6MR MINIMIPS PROCESSOR SYSTEM

	LUTs		Registers		BRAMs	
	(#)	(%)	(#)	(%)	(#)	(%)
Fault injector	938	3.26	561	1.95	2	0
Each module (6 times)	3514	12.20	1500	5.21	3	0
SAv voter	193	0.67	98	0.34	0	0
Test control	174	0.60	117	0.41	0	0
Total	**22389**	**77.74**	**9776**	**33.94**	**20**	**33.3**

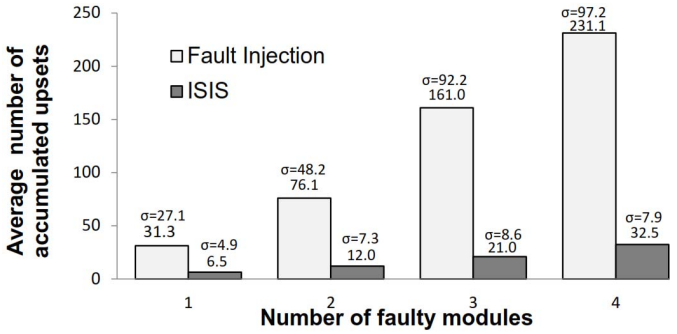


Figure 11. Comparison between fault injection and neutron radiation experiments results (σ: standard deviation) of miniMIPS processors. Discrepancies are due fault injector does not inject faults in RAM blocks. According to reliability report [11] it is expected to obtain around 5.9 times more accumulated upsets than obtained from masked readback of radiation

REFERENCES

[1] Xilinx, UG191. "Virtex-5 FPGA Configuration User Guide", 2010.

[2] Chapman, K. "SEU strategies for Virtex-5 Devices, XAPP864 ", 2010.

[3] Alderighi, M.; Casini, F.; Citterio, M.; D'Angelo, S.; Mancini, M.; Pastore, S.; Sechi, G.R.; Sorrenti, G., "Using FLIPPER to predict irradiation results for VIRTEX 2 devices", Radiation and Its Effects on Components and Systems (RADECS), pp: 300 – 305, 2008.

[4] Guzman-Miranda, H. ; Tombs, J.N. ; Aguirre, M.A., "FT-UNSHADES-uP: A platform for the analysis and optimal hardening of embedded systems in radiation environments", Industrial Electronics, pp.: 2276 – 2281, 2008.

[5] Sterpone, L. ; Violante, M. ; Rezgui, S., "An Analysis Based on Fault Injection of Hardening Techniques for SRAM-Based FPGAs", IEEE Trans. Nucl. Sci,. v53 , n4, pp.: 2054 – 2059, 2006.

[6] M. Violante, L. Sterpone, a. Manuzzato, S. Gerardin, P. Rech, M. Bagatin, a. Paccagnella, C. Andreani, G. Gorini, a. Pietropaolo, G. Cardarilli, S. Pontarelli, and C. Frost, "A New Hardware/Software Platform and a New 1/E Neutron Source for Soft Error Studies: Testing FPGAs at the ISIS Facility," IEEE Trans. Nucl. Sci., vol. 54, no. 4, pp. 1184–1189, Aug. 2007.

[7] Xilinx, "PicoBlaze 8-Bit Embedded Microcontroller User Guide, UG129 ". 2005.

[8] Tambara, L.; Rech, P.; Kastensmidt, F. Evaluating the Effectiveness of a Diversity TMR Scheme under Neutrons. In: Radiation and its Effects on components and systems (RADECS), 2013

[9] A. Anghel, D. Alexandrescu and M. Nicolaidis, "Evaluation of a Soft Error Tolerance Technique Based on Time and or Hardware Redundancy," in Proc. IEEE Integrated Circuits and Systems Design, Manaus, Sept. 2000, pp. 237-242.

[10] J. Tarrillo, F. L. Kastensmidt, P. Rech, C. Frost, C. Valderrama, "Neutron Cross-section of N-Modular Redundancy Technique in SRAM-based FPGAs", IEEE Trans. Nucl. Sci, v61, n.4, p.1558-1566, 2014.

[11] Xilinx, UG116. "Device Reliability Report", February 14, 2014.