Rev. Roum. Sci. Techn.– Électrotechn. et Énerg.
Vol. **69**, *3*, pp. 299–304, Bucarest, 2024

Électronique et transmission de l'information
Electronics and Information Technology

BUFIT: FINE-GRAINED DYNAMIC BURST FAULT INJECTION TOOL FOR EMBEDDED FIELD PROGRAMMABLE GATE ARRAY TESTING

SINDHU THAZHATHETHIL VELAYAUDHAN[1, *], KALPANA DEVI[1]

Keywords: Field programmable gate arrays (FPGAs); Adaptive fault injection (AFI) tool; Linear feedback shift register; Multiple bit upsets.

Fault injection (FI) is a well-known method to attack embedded systems, particularly advanced FPGAs and microcontrollers physically. The FPGA-based embedded system constitutes SRAM for configuration data storage. Multiple-bit upset is a main threat for FPGAs due to technology scaling and complex application bit files. Space environments additionally incur radiation threats to these devices. This paper proposes burst error modeling and a burst fault injection tool (BUFIT) to address these issues. BUFIT has been proposed with fine-grained and coarse-grained circuits. Built-in instrumented FPGA-based FI is proposed for effectively injecting MBUs in configuration memory with space adaptive rate for accurately estimating soft errors. Evaluation of proposed BUFIT on Kintex-7 target FPGA to various OR 1200-based workloads is given to analyze the speed up of the proposed technique. Results on the OR 1200 processor show that BUFIT is three and two orders of magnitude faster than existing DPR and SCFIT techniques. It uses only 0.4 % CLB overhead and has negligible impact on FFs of target SFPGAs.

1. INTRODUCTION

Injection of faults is a common technique for assessing a system's resilience to physical faults [1]. The field programmable gate array (FPGA) is widely employed in practical field applications because of its low power consumption and flexible programming. Unlike Von Neumann-type devices like microcontrollers and digital signal processing (DSP) processors, the FPGA comprises reconfigurable logic, I/O, and connectivity blocks [2]. When referring to environments that could be detrimental to the field-programmable gate arrays' (FPGAs') dependable operation, the term "hostile environment" is usually used. System resilience is tested frequently when systems are implemented in hostile contexts where errors are likely to occur [3]. Since then, all potential uses of radiation-tolerant circuits, such as space missions, satellites, and high-energy physics experiments, have raised interest in exploring fault-tolerant approaches to keep integrated circuits (ICs) functioning in hostile environments [4].

FPGAs mimic defects in electronic systems; this technique is known as an FPGA-based fault injector. An extensive range of digital logic operations can be executed using FPGAs, integrated circuits that can be programmed after manufacturing [5]. FPGA-based fault injectors are employed when examining how errors like bit flips affect the dependability and efficiency of digital systems. Applying this is essential for safety systems, such as medical devices, aircraft, and automobiles. Experiments simulating the impact of MBUs can be conducted with FPGA-based fault injectors by purposefully flipping several bits in the data memory or configuration of the FPGA. Researchers can examine the behavior of the FPGA under practical fault scenarios and validate MBU mitigation strategies by injecting faults based on MBU models.

Multiple-bit upset (MBU) modeling studies and simulates scenarios where multiple bits in a memory unit are corrupted simultaneously. MBU occurs due to high-energy particles, radiation, or other environmental factors that can flip multiple bits in a memory cell, leading to data corruption [6]. MBU modeling helps develop error detection and correction mechanisms to mitigate the impact of such faults.

Over the past few decades, applications for FPGAs in high-energy physics and aerospace have grown in popularity. FPGAs are popular for these applications because of their many advantages, including great adaptability, cheap cost, and fast turnaround time. This is especially true compared to more expensive, specialized alternatives like application-specific integrated circuits [7].

Since commercial SRAM-based FPGAs are more cost-effective and perform better than radiation-hardened FPGA systems, they are now often utilized in radiation settings. Commercial SRAM-based FPGAs are undoubtedly less expensive than radiation-hardened FPGA solutions, but whether or not they "work better" will rely on the particular needs and limitations of the application and the radiation environment [8]. Because of their wider market availability and higher production numbers, commercial SRAM-based FPGAs are more economical than radiation-hardened FPGAs [9].

Advanced techniques in silicon manufacture are employed to attain elevated frequencies and superior performance. Additionally, FPGAs are machines that can be programmed. It can alter their behavior during development to satisfy different mission objectives [10]. One unresolved matter is the relationship between hardware- and software-based fault injection vulnerability detection. Here, fault injection vulnerabilities are found by utilizing both hardware and software. Hardware-based detection is achieved using an EMP generator [11].

The effects of SETs happening during configuration memory re-writing should be thoroughly examined, as reconfiguration tasks are critical to the availability, flexibility, and dependability of FPGA applications. A methodology has been proposed to assess SET pulses' effects when reconfiguring configuration memory in SRAM-based FPGAs [12]. Additionally, SRAM-based FPGAs have more memory elements than their ASIC counterparts, so they are more susceptible to Single Event Upset (SEU). Due to their higher operating voltages, early SRAMs were more resilient to soft errors. On the other hand, the node capacitance and operating voltage decrease with each successive SRAM generation [21,22]. Burst error modeling and burst fault injection tool (BUFIT) have been proposed to overcome these challenges. The major contribution of BUFIT is,

[1] Veltech Rangarajan Dr. Sagunthala R & D Institute of Science and Technology, Chennai, India
Emails: vtd845@veltech.edu.in (corresponding), drkalpanadevip@veltech.edu.in

DOI: 10.59277/RRST-EE.2024.69.3.8

- Built-in Instrumented FPGA-based FI for effectively injecting MBUs into the configuration memory of SFPGAs.
- FI is based on an adaptive rate for accurate estimation of soft errors.
- Evaluation of BUFIT on Kintex-7 target FPGA to various OR 1200-based workloads to analyze the speedup of the proposed technique.

The rest of the research will be shown below. Section 2 reviews past research on AD detection using the ADNI database and other databases. Section 3 explains the proposed approach in extensive detail. Section 4 discusses the outcome and discusses the discussion. Section 5 provides future research directions and a summary.

Table 1

List of abbreviations

Abbreviation	Description
FPGA	Field programmable gate array
FI	Fault injection
(DSP)	Digital signal processing
(EMP)	Electromagnetic pulse
SRAM	Static random access memory
MBU	Multiple bit upsets
(SEU)	Single event upset
QEMU	Quick EMUlator
ARM	Advanced RISC machines
TRAITOR	TRAnsportable glItch aTtack platfORm
(EMFI)	Electromagnetic fault injection
SIFA	Statistical ineffective fault analysis
(DUT)	Design under test
(PDR)	Partial dynamic reconfiguration
(ICAP)	Internal configuration access port
(LFSR)	Linear feedback shift register
ICAP	Internal configuration access port
FG-LFSR	Fine grained LFSR circuit
(FFs)	Flip flops

2. LITERATURE SURVEY

This section discusses the various fault injection techniques for FPGA-based embedded systems.

In 2022, Metawie H. *et a*l. [13] suggested a framework for fault injection using the Quick EMUlator (QEMU). It can simulate faults in the control and execution channels of an ARM processor and extend the fault model for memory coupling problems. They illustrate the usefulness of the approach by evaluating a memory exam.

In 2023, Lanzieri L. *et al*. [14] proposed age detection and monitoring in embedded systems. Hardware aging is an increasing issue for embedded devices that play critical roles in reliable or safety-critical applications. The primary goal of this work is to facilitate future research efforts in this area by organizing all major approaches.

In 2021, Claudepierre, L. *et al.* [15] proposed a low-cost TRAITOR platform that can inject numerous, accurate bursts of faults using clock glitches. The errors are caused by clock glitch injection, which has high repeatability and reliability. This platform is inexpensive, simple to use, and capable of injecting many spurts of faults. Future development will extract an exact fault model for TRAITOR using the STM32F100RB board. Furthermore, the investigation of software or hardware countermeasures is being explored.

In 2022, Richter-Brockmann, J. *et al.* [16] proposed revisiting adversary model hardware faults. Additionally, using custom models makes comparing various designs and evaluation results more difficult. Furthermore, it demonstrates that the suggested adversary model can be incorporated into VerFI, a cutting-edge fault-proof tool.

In 2019, according to a recent study, Liao H. *et al.* [17] suggested that the security of embedded devices is significantly impacted by electromagnetic fault injection (EMFI) techniques. This paper proposes a novel EMFI backside technique based on overclocking and an enlarged fault model based on the concept of critical charge. The security and fault injection resistance of embedded processors and their instruction set designs depend heavily on this research. Part of the study's funding comes from contributions from XtremeEDA.ds and NSERC.

In 2020, Breier J. *et al.* [18] developed a novel method to protect implementations against SIFA based on error-correcting codes. They created an electronic logic analysis tool that checks the output for errors, cycles through all potential inputs, and injects a stuck-at-fault at each gate in the circuit.

In 2018, Cerveira F. *et al.* [19] suggested analyzing the exploratory data of fault injection campaigns. This essay adopts a contemporary perspective on these problems by organizing and executing information extraction using exploratory (big) data analysis techniques, tools, and approaches. As a result, a previously undiscovered possibility for a sharp acceleration of the FI process has been discovered.

Several techniques were used to inject the faults in single-bit and multiple-bit scenarios. However, it faces challenges like immediate practical needs of mitigating hardware aging in current systems and more complex fault models beyond clock glitches. In this work, a novel BUFIT has been proposed to address these issues

3. PROPOSED BUFIT FOR MBU INJECTION

A simulation framework for MBU injection and its associated design methodology is described in this section. The early estimation of sensitivity to run-time MBUs of SFPGA is much required to reduce the further accumulation of MBUs. It allows for the exploration of MBU modeling and the anticipation of its design before the implementation of an efficient MBU injector. The MBU injection framework shown in Fig. 1 is based on modeling an event-driven simulator by including the functional models for the FPGA. The designer can redefine the readback rate, MBU injection rate, frame address, and fault list. This scalable, configurable, and versatile framework allows for virtualizing real-time fault emulation experiments under dynamic radiation environments.

Radiation-induced MBUs are emulated by synthetically changing the contents of the FPGA configuration memory through in-built FI. The FPGA's output is then monitored to decide the impact of a given configuration memory upset on the originally implemented design behavior.

An efficient FI technique requires knowledge of possible fault models; this knowledge will vary for different FPGA resources. For example, Stuck-at-1 or Stuck-at-0 model is used to induce the fault in the routing resource of FPGAs, and the bit flip fault model is used to induce the fault in the memory resources of FPGAs. The recent radiation experiment shows that more than 48 % [20] of the faults are MBUs; in particular, 2-bit upset, 3-bit upset, and 4-bit upset play an important role, and a maximum of 8-bit upset is possible to occur in the same word of the memory units. Based on the real-time radiation experiments on recent technological devices, the modeling of different SBU and MBU fault patterns is done in this work. The 4-bit Linear

Feedback Shift Register (LFSR) produces a 4-bit random fault with any seed value. The number of possibilities for generating the random faults for any 4-bit seed value is shown in Table 2.

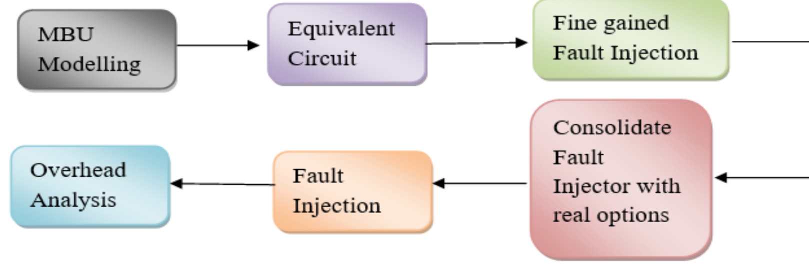


Fig. 1 – Proposed MBU injection framework.

The polynomial equation for 4-bit LFSR is given by

$$x^4 \oplus x^3 \oplus 1. \tag{1}$$

The general representation of MBU models is given in eq. (2) to (5), where n represents the bit width of the LFSR circuit and m represents the number of errors in the MBU set.

$$f_{m=1} = n, \tag{2}$$

$$f_{m=2} = n + m, \tag{3}$$

$$f_{m=3} = n, \tag{4}$$

$$f_{m=4} = 2^{n-m}. \tag{5}$$

Table 2

Random fault modeling using 4-bit LFSR

Bit Flip Fault Models	f1	f2	f3	f4
For Different Seeds	4	6	4	1

The eight 4-bit LFSRs are connected in parallel to generate 32-bit random data. The polynomial equation for 32-bit LFSR is given by

$$x^{32} \oplus x^{22} \oplus x^2 \oplus x^1 \oplus 1. \tag{6}$$

The seed of six LFSRs is considered constant, and the seed of another two LFSRs is varied. If the number of ones in the 4-bit variable seed changes, it can generate a maximum of 4-bit upset. Table 3 shows the modeling of MBUs based on a fine-grained LFSR circuit (FG-LFSR).

Table 3

Fine-grained LFSR-based MBU model

FG-LFSR	Mechanism	Proposed fault model	
SBU	1-bit		25 %
MBU	2-bit	75%	37 %
	3-bit		25 %
	4-bit		7 %
	5-bit		2 %
	6-bit		4 %

3.1. PROPOSED BUFIT

The proposed FPGA-based fault injector is performed in two steps: i) fine-grained and ii) coarse-grained.

3.1.1. *Fine-grained fault injector*

The issue is caused by the XOR operation of the data from the FPGA memory and the random vector produced by the linear feedback shift register (LFSR).

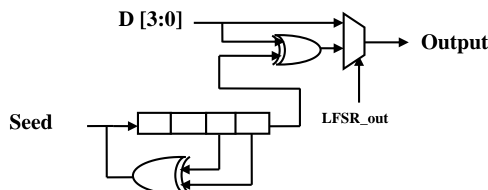


Fig. 2 – Proposed fine-grained 4-bit fault injector circuit.

The existing and proposed fault models can derive from LFSR's seed and feedback polynomial vectors. The proposed fine-grained is a 4-bit fault injector, and the circuit is shown in Fig. 2. The 4-bit data D [3:0] from memory is XOR with the seed to generate a random 4-bit vector from the LFSR.

3.1.2. COARSE-GRAINED FAULT INJECTOR

The reconfiguration signal is used to select the error data for FI and error-less data for normal operation. Figure 3 shows the coarse-grained 32-bit fault injector circuit, in which eight 4-bit fault injector circuits are concatenated to inject a 32-bit fault at a time.

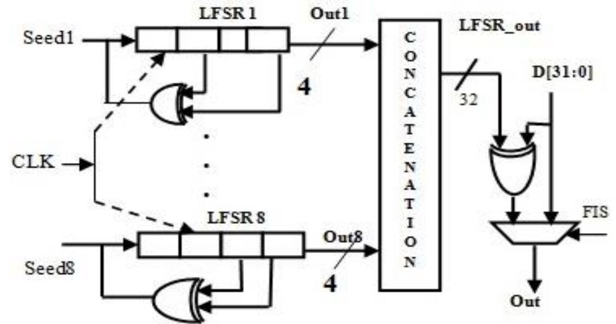


Fig. 3 – Proposed coarse-grained 32-bit MBU fault injection circuit.

The fine-grained fault injector may be single, double, triple, or four-bit upsets. Table 1 matches the possibility of a fault occurrence. The influence from Table 2 and Table 3 indicates that the fault occurrences are similar.

3.2. BUFIT FLOW

The fault injector flow of the proposed BUFIT is shown in Fig. 4 and consists of 3 phases:

i) Initialization phase
ii) Emulation phase
iii) Classification phase.

In a harsh radiation environment, the initialization phase converts the given FPGA clock rate into an adaptive fault injection (AFI) rate. The emulation phase consists of a read-back manager and adaptive fault injector to induce the adaptive rate MBUs in the configuration memory. Finally, the fault classification phase analyzes the faults to improve the controllability and observability of FI. Equation (7) gives the relationship between fault rate, read-back rate, and AFI rate

$$\text{Fault rate} \times \text{Read back rate} \times \text{AFI rate}. \tag{7}$$

The real-time space environment is largely ionized, which leads to a high fault rate and a highly respected read-back rate, which makes for fast FI. The small ionization present in

the space environment leads to a slow fault rate and a slow, respected read-back rate, which makes for slow FI.

3.2.1. *Initialization phase*

The main function of the initialization phase is setting the host FPGA clock and defining the fault list. The FI rate can be fixed or varied. The target FPGA is based on a particular frequency, and the rate converter is required to realize efficient FI. The function of the rate converter is to either increase or decrease the initial FPGA clock frequency based on the system requirement.

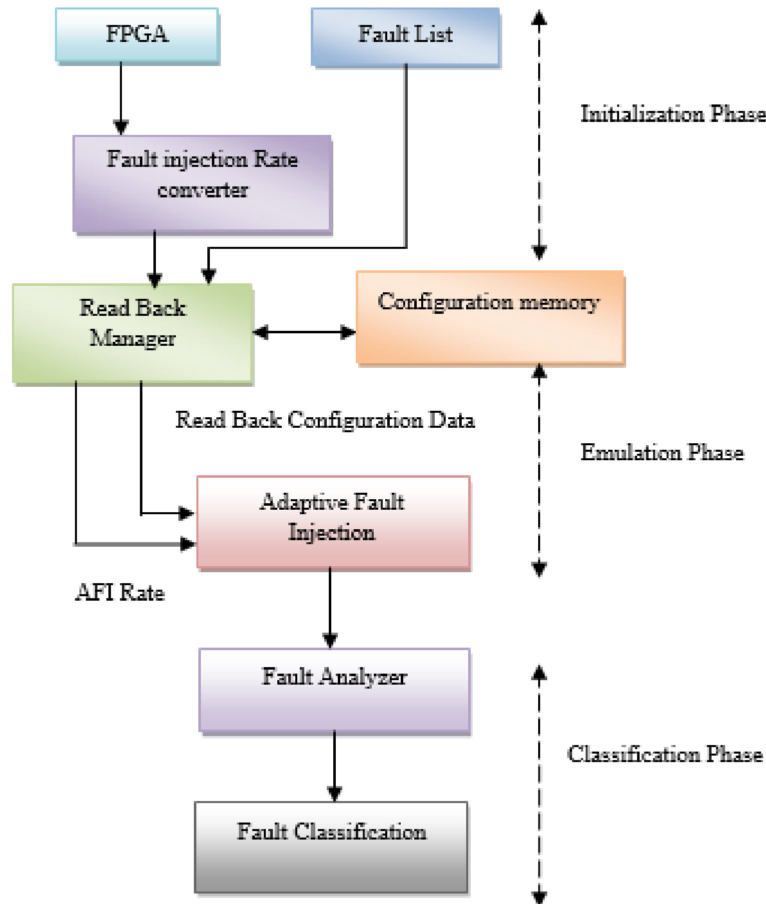


Fig. 4 – BUFIT flow.

3.2.2. *Emulation phase*

The heart of the proposed BUFIT is the emulation phase, which consists of configuration ma read-back manager, and an adaptive fault injector module. The read-back manager gives an adaptive fault injection rate and specifies configuration data for input tonput of the adaptive fault injector module, generating dynamic faulty data.

3.2.3. *Classification phase*

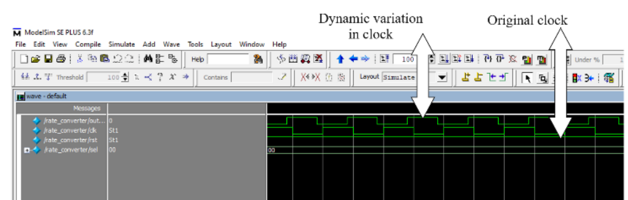
The fault classification is performed in a computer connected with host FPGA. Serial communication interface is used to connect the laptop or computer with the host FPGA. A designer can evaluate the expected failure rate of the circuit and the effectiveness of the implemented fault mitigation mechanisms by using fault classification.

4. **RESULTS AND DISCUSSION**

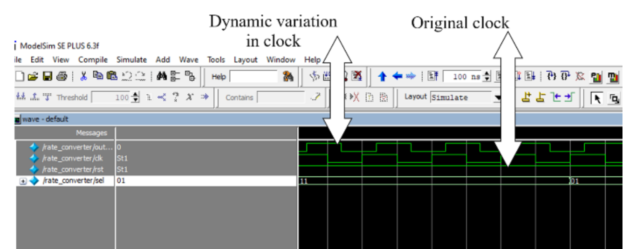
The proposed BUFIT is simulated and implemented on FPGA to generate faults suitable for dynamic system

4.1. SIMULATION ANALYSIS

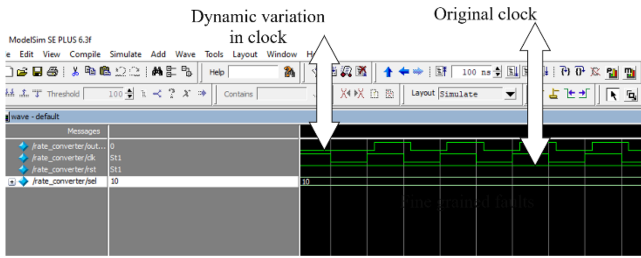
The proposed MBU Injection framework consists of different sub-modules like rate converter to realize the real-time radiation environment, fine-grained fault injection circuit, and integration of all sub-modules to realize the complete MBU injection task. The rate converter module converts the original FPGA clock to the expected radiation environment. This could be achieved by 'sel' 00, 11, 10 inputs. For the different values of 'sel', the simulation plots are given in Fig. 5.



(a)



(b)



(c)

Fig. 5 – Rate converter: a) – sel: ‘00’; b) – sel: ‘11’; c) – sel: ‘10’.

4.2. FINE-GRAINED FAULT INJECTOR

The simulation plot of the proposed fine-grain LFSR-based 4-bit FI is shown in Fig. 6, which gives the random 4-bit outputs. This fault injector circuit gets the clock signal from the previously said rate converter circuit for sel: ‘00’, sel: ‘11’, sel: ‘10’.

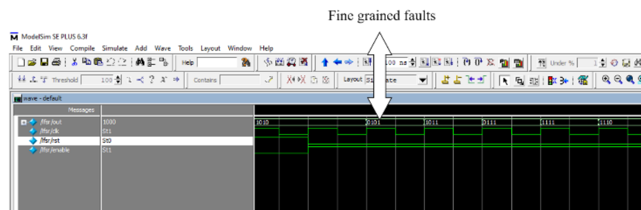


Fig. 6 – Fine grain fault generation.

4.3 COARSE-GRAINED FAULT INJECTOR

Similarly, the simulation plot of the proposed LFSR-based 32-bit fault injector gives the random 32-bit outputs, as the circuit shown in Fig. 7.

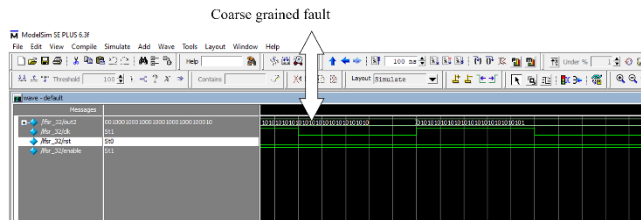
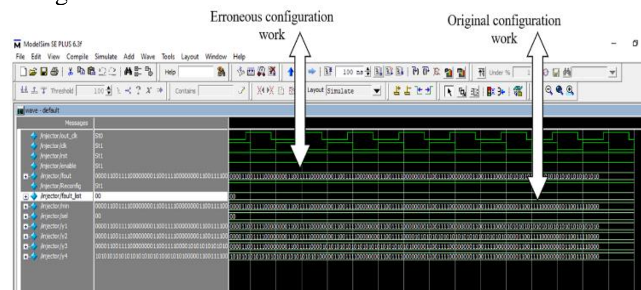


Fig. 7 – Coarse grain fault generation.

The MBU injection framework, BUFIT simulation, is performed with different radiation environments and memory locations. Different memory locations are represented by the signal ‘fault list’. Figure 8(a) shows the simulation plot of BUFIT with the memory location of fault list ‘00’. The 128-bit original configuration word is taken as input, and the 2-bit fault list is taken as another input for inducing the 32-bit MBU in the LSB part of the original configuration word. Similarly, a fault list for the ‘10’simulation plot of BUFIT with the memory location was also generated.



(a)

Fig. 8 – Simulation plot of AFITO (a) impact of fault list ‘00’.

4.4. COMPARATIVE ANALYSIS

The MBU vulnerability of the OR 1200 processor under various real application workloads is evaluated in Nintex-7 FPGA xc7k70t-2-fbg676. Table 4 provides a comparison of different fault injection methods in terms of their efficiency and performance. The efficiency of the proposed BUFIT method compared to DPR and SCFIT. BUFIT demonstrates a significantly lower total injection time and a higher injection frequency, suggesting that it can inject faults more rapidly and efficiently. BUFIT has the shortest injection time (18.7 ms) compared to DPR (54 ms) and SCFIT (36 ms). As shown in Table 4, the proposed BUFIT is three and two times faster than the existing DPR and SCFIT, respectively.

Table 4

Fault injection time

Injection Method	Instru-ment delay (ms)	Write delay (ms)	Injection	
			Time (ms)	Frequency (Hz)
DPR	–	54	54	18.5
SCFIT	18	18	36	27
Proposed BUFIT	7	18	18.7	53.4

Table 5 shows the fault injection time and speed up comparison for different workloads. Counter and Bubble sort circuits occupies 36 and 144 frames respectively in the chosen target FPGA. 4-bit adder and 4-bit multiplier circuits also implemented with the proposed BUFIT and the existing DPR.

Table 5

Speedup comparison of fault injection techniques

Work load	DPR (ms)	SCFIT (ms)	Proposed BUFIT (ms)	Speed up	
				DPR	SCFIT
Counter	1944	1296	673	~3	~2
Bubble sort	7779	5184	2693	~3	~2
4-bit adder	1466	983	512	~3	~2
4-bit multiplier	3122	2042	1064	~3	~2

The speed-up of the proposed technique over existing DPR and SCFIT are 3× and 2×, respectively. The proposed BUFIT method has the lowest instrument delay at 7 ms and a write delay of 18 ms, resulting in a total injection time of 18.7 ms. This method achieves the highest injection frequency at 53.4 Hz, making it the most efficient in injecting faults quickly and frequently. Overall, the table highlights the efficiency of the proposed BUFIT method in terms of lower injection time and higher frequency compared to DPR and SCFIT, suggesting that BUFIT is superior for applications requiring rapid and frequent fault injections.

Table 5 compares the fault injection times for different workloads using three techniques: DPR, SCFIT, and the proposed BUFIT. Table 6 shows FPGA resource overhead analysis due to FI instrument added with the target FPGA. The SCFIT technique consumes 4.3 % CLBs and 5.8 % flip flops (FFs) overhead in addition to the target FPGA. The required FFs are much higher than the maximum available FFs in the Kintex-7 FPGA. This result shows the practical limitation of using the SCFIT technique. However, the proposed BUFIT does not have such a limitation, and it needs only 0.4 % additional CLBs and a negligible amount of FFs.

Table 6

FPGA resource overhead due to built-in AFITO

TARGET	TOTAL CLBs	TOTAL FFs
Kintex-7 FPGA	69120	69120
OR1200	34228	66847
OR1200+SCFIT	35873 (4.8 %)	70737 (5.8 %)
OR1200+ proposed AFITO	34357 (0.4 %)	66853 (~0 %)

5. **CONCLUSION**

The proposed BUFIT injects MBUs into the memory elements of SFPGA. These MBU's sizes varied similarly to real-time radiation environments. Both single bits upset and MBU were realized by using a coarse-grained circuit. Results on the OR 1200 processor show that BUFIT is three and two orders of magnitude faster than existing DPR and SCFIT techniques, and it uses only 0.4 % CLB overhead and has negligible impact on FFs of target SFPGAs. The future scope of this work is to implement additional features with the BUFIT proposed for improving fault classification performance on SFPGA. Moreover, the deterministic nature and finite cycle length of LFSRs should be addressed by exploring the integration of true random number generators (TRNGs), which could be more appropriate in the future.

ACKNOWLEDGEMENTS

The author would like to express his heartfelt gratitude to the supervisor for his guidance and unwavering support during this research for his guidance and support.

Received on 25 May 2023

REFERENCES

1. A. Gangolli, Q.H. Mahmoud, A. Azim, *A systematic review of fault injection attacks on IOT systems.* Electron., **11**, *13*, p. 2023 (2022).
2. Z. Gao, X. Liu, *An overview on fault diagnosis, prognosis and resilient control for wind turbine systems,* Processes, **9**, *2*, p. 300 (2021).
3. M. Carminati, G. Scandurra, *Impact and trends in embedding field programmable gate arrays and microcontrollers in scientific instrumentation,* Rev. Sci. Instrum., **92**, *9*, p. 091501 (2021).
4. A. Appathurai, P. Deepa, *Design for reliability: A novel counter matrix code for FPGA based quality applications,* 6th Asia Symposium on Quality Electronic Design (ASQED), pp. 56–61 (2015).
5. L.A. Aranda, A. Sánchez-Macián, J.A. Maestro, *ACME: A tool to improve configuration memory fault injection in SRAM-based FPGAs,* IEEE Access, **7**, pp.128153–128161 (2019).
6. S. Medjmadj, D. Diallo, A. Arias, *Mechanical sensor fault-tolerant controller in PMSM drive: experimental evaluation of observers and signal injection for position estimation.* Rev. Roum. Sci. Techn. – Électrotechn. Et Énerg., **66**, *2*, pp.77–83 (2021).
7. A. Ahilan, P. Deepa, *Modified decimal matrix codes in FPGA configuration memory for multiple bit upsets,* International Conference on Computer Communication and Informatics, **10** (2015).
8. T. Given-Wilson, N. Jafri, A. Legay, *Combined software and hardware fault injection vulnerability detection.* Innovations Syst. Software Eng., **16**, pp. 101–120 (2020).
9. S. Mandal, S. Sarkar, W.M. Ming, A. Chakrabarti, *Criticality aware soft error mitigation in the configuration memory of SRAM based FPGA*, 32nd International Conference on VLSI Design and 18th International Conference on Embedded Systems (VLSID), pp. 257–262 (2019).
10. A. Ramos, R.G. Toral, P. Reviriego, J.A. Maestro, *An ALU protection methodology for soft processors on SRAM-based FPGAs*, IEEE Trans. Comput., **68**, *9*, pp.1404–1410 (2019).
11. R.V.W. Putra, M.A. Hanif, M. Shafique, *Respawn: energy-efficient fault-tolerance for spiking neural networks considering unreliable memories*, 2021 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 1–9.
12. I. Bentchikou, K. Halbaoui, F. Boudjema, D. Boukhetala, T. Abdelhalim, *Alternative hybrid control of switched systems. An application to machine DC fed by multicellular converter*, Rev. Roum. Sci. Techn. – Électrotechn. Et Énerg., **67**, *3*, pp. 247–252 (2022).
13. H. Metawie, M. Safar, M.W. El-Kharashi, *An evaluation method for embedded software dependability using QEMU-based fault injection framework*, 6th International Conference on System Reliability and Safety (ICSRS), pp. 548–555 (2022).
14. L. Lanzieri, G. Martino, G. Fey, H. Schlarb, T.C. Schmidt, M. Wählisch, *A Review of Techniques for Ageing Detection and Monitoring on Embedded Systems.* arXiv preprint arXiv:2301.06804 (2023).
15. [illegible] Claudepierre, P.Y. Péneau, D. Hardy, E. Rohou, *TRAITOR: a low-cost evaluation platform for multifault injection*, Proceedings of the 2021 International Symposium on Advanced Security on Software and Systems, pp. 51–56.
16. [illegible] Richter-Brockmann, P. [illegible]rich, T. Guneysu, *Revisiting fault adversary models-hardware faults in theory and practice*, IEEE Trans. Comput, 2022.
17. H. Liao, C. Gebotys, *Methodology for em fault injection: Charge-based fault model*, Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 256–259 (2019).
18. J. Breier, M. Khairallah, X. Hou, Y. Liu, *A countermeasure against statistical ineffective fault analysis*, IEEE Transactions on Circuits and Systems II: Express Briefs, **67**, *12*, pp. 3322–3326 (2020).
19. F. Cerveira, I. Kocsis, R. Barbosa, H. Madeira, A. Pataricza, *Exploratory data analysis of fault injection campaigns*, IEEE International Conference on Software Quality, Reliability and Security (QRS), pp. 191–202 (2018).
20. A. Chatzidimitriou, G. Papadimitriou, C. Gavanas, G. Katsoridas, D. Gizopoulos, *Multi-bit upsets vulnerability analysis of modern microprocessors*, IEEE International Symposium on Workload Characterization (IISWC), pp. 119–130 (2019).
21. B. M. Kumar, J. Ragaventhiran, V. Neela, *Hybrid optimization integrated intrusion detection system in WSN using ELMAN network,* International Journal of Data Science and Artificial Intelligence, **02**, *02*, pp. 55–62 (2024).
22. M. Anisha, V.A. Beenu, *Double secure cloud medical data using Euclidean distance-based Okamoto Uchiyama homomorphic encryption,* International Journal of System Design and Computing, **02**, *01*, pp. 1–7 (2024).