Original scientific paper

Informacije MIDEM
Journal of Microelectronics,
Electronic Components and Materials
Vol. 43, No. 4 (2013), 259 – 266

On-line Testing and Recovery of Systems with Dynamic Partial Reconfiguration

Anton Biasizzo

Jožef Stefan Institute, Ljubljana, Slovenia

Abstract: The FPGA devices are increasingly being used in mission critical systems like security systems, banking systems, and avionics. The errors induced by high-energy radiation, also known as Single Event Upsets (SEUs), corrupt the configuration memory of the FPGA device and are a major concern for the system reliability and dependability. For this, error mitigation techniques like triple module redundancy and ECC codes are commonly employed techniques. However error mitigation techniques do not recover the system. The fault-free system can be recovered using reconfiguration of the FPGA device. Previous recovery methods employ processor cores as a reconfiguration controller consuming notable amount of device resources and introducing additional error detection and recovery latency. In this paper a low area overhead error recovery mechanism for SRAM based FPGA systems is presented. The error recovery mechanism is implemented as a state machine. The reliability of the developed solution was experimentally evaluated by fault emulation environment.

Keywords: FPGA, dynamic partial reconfiguration, self-recovery, fault emulation, single event upset

Sprotno preiskušanje in popravljanje sistemov z dinamično delno rekonfiguracijo

Izvleček: Programirljiva vezje FPGA se vedno bolj uporabljajo tudi v visoko zanesljivih sistemih, ki se uporabljajo v nadzornih sistemih, bančništvu in v letalski industriji. Napake, nastale zaradi visoko-energijskega sevanja, imenovane SEU, lahko spremenijo vsebino konfiguracijskega spomina vezja FPGA in predstavljajo eno večjih težav pri razvoju zanesljivih sistemov. Najpogosteje zanesljivost sistemov izboljšamo s pomočjo metod za zmanjševanje vpliva napak kot sta potrojitev modulov in uporaba kod za odpravljanje napak. Pomanjkljivost takih metod je v tem, da ne odpravijo same napake v sistemu. Napake v sistemu osnovanemu na vezjih FPGA lahko odpravimo z ponovnim konfiguriranjem vezja FPGA. Obstoječe metode popravljanje sistemov osnovanih na vezjih FPGA kot krmilnik rekonfiguracije uporabljajo procesorko jedro kar znatno poveča porabo logičnih blokov, poveča pa tudi latentnost pri odkrivanju ter odpravljanju napak. V članku je predstavljen mehanizem odpravljanja napak za sisteme osnovne na vezjih FPGA, ki uporabi relativno malo logičnih blokov. Mehanizem odpravljanja napak je izveden z avtomatom prehajanja stanj. Zanesljivost razvite metode je bila ovrednotena s pomočjo okolja za emulacijo napak v vezjih FPGA.

Ključne besede: dinamična delna rekonfiguracija, samopopravljivost, emulacija napak

* *Corresponding Author's e-mail: anton.biasizzo@ijs.si*

1 Introduction

SRAM-based FPGA devices are increasingly being used for implementing embedded applications used as a part of a mission-critical and reliable system. The main advantage of FPGAs is their high reconfigurability, which enables fast prototyping, flexible functionality through partial reconfiguration, on-site hardware upgrades, and on-site configuration recovery. Due to the increasing integration density FPGA devices are getting more susceptible to faulty behavior, caused by cosmic or artificial radiation [1-3]. Such faults are modeled as Single Event Upsets (SEUs). While radiation is a major concern in space [4], systems in avionics and on ground level are less exposed to it because of the planetary atmospheric and magnetic radiation shield. However, experiments [1-2,5] showed that with increased density of integrated circuits the neutron particles present in the atmosphere are also capable of producing SEU.

The Single Event Upset is a change of logic state caused by the radiation. It is a result of the free charge created by ionization in the node proximity. While such errors are typically transient they can cause a bit flip in the device memory.

SRAM FPGA devices are programmable logic devices using the SRAM memory for storing the configuration. While SRAM memory enables the effortless and rapid prototyping, they are especially vulnerable to SEUs. Thus it is imperative that FPGA based applications, where high reliability is required, include mechanisms that can easily and quickly detect and correct SEUs.

Many techniques have been developed to protect critical systems on SRAM FPGAs against SEU [6]. At the design level of the FPGA these techniques are classified as SEU mitigation techniques which prevent SEU to disturb the normal operation of the target design, and SEU recovery techniques that recover the original programmed information in the FPGA configuration memory after an upset.

Some SEU mitigation techniques use time redundancy but they are effective only against transient faults. The most common SEU mitigation techniques employ hardware redundancy like Triple Modular Redundancy (TMR) and Error Correcting Codes (ECC). In the case of TMR, design logic is triplicated and a voter is used to identify the correct value [7-10]. Since the voter is also vulnerable to upsets an improved TMR strategy for FPGA was developed [7]. The voters are triplicated and implemented using dedicated FPGA logic resources. TMR can be distributed over reconfigurable modules and when the voter detects a fault the faulty module can be recovered by a partial reconfiguration. A controller for managing the reconfiguration was proposed by [11].

Error Correction Codes (ECC) [12] are also used to mitigate the SEU in integrated circuits. Different ECC are used to protect systems against single and multiple SEUs. The most common ECC are Hamming codes and Reed-Solomon codes. ECC are mostly used to protect memories of the systems.

The SEU recovery techniques in SRAM FPGAs are also known as configuration scrubbing. The basic principle of this method is to use partial reconfiguration to recover SEUs within the FPGA configuration memory. Depending on which FPGA configuration interface is used to reconfigure the device, the scrubbing techniques are classified as external and internal. The external scrubbing techniques use external configuration ports (i.e. JTAG, SelectMap). They require an external radiation hardened scrubbing controller (processor [13], or FPGA [14]) and external radiation hardened memory to store the so called "Golden copy" of the FPGA configuration bits. The internal SEU recovery techniques use internal configuration interface to access the configuration memory of the FPGA. Scrubbing is also controlled internally and the controller usually consists of an embedded microprocessor [15-18].

The SEU detection-and-recovery mechanism should be fast in order to reduce the system-error latency. Besides, since it is implemented with similar FPGA resources as the target application, the mechanism itself is subject to SEUs. It is therefore imperative that its hardware overhead is as small as possible.

This paper summarizes the results of the development of a novel internal SEU detection-and-recovery mechanism [19]. This approach outperforms existing recovery mechanisms in terms of speed as well as in minimizing the hardware overhead. It can be implemented using different error mitigation techniques to achieve required degree of the system reliability. The error recovery mechanism with accompanying error mitigation techniques are evaluated using a specially designed fault-emulation environment that allows the injection of faults into specified FPGA resources. The obtained failure-in-time (FIT) estimations can be used to select appropriate solution.

2 Single event upsets in SRAM FPGA

FPGA device is an integrated circuit that consists of an array of logic blocks, interconnection networks, and configuration memory. The configuration memory controls the behavior of each logic block as well as the connections in the interconnection networks. Changing the content of the configuration memory different logic circuits can be implemented on such array.

In the Xilinx FPGA the configuration memory is organized as a matrix of configuration frames. More detailed structure of the FPGA configuration matrix is depicted in Figure 1. A configuration frame is the smallest reconfigurable part of a FPGA device. The size of the configuration frame in Xilinx Virtex 4 and Virtex 5 FPGA family is 41 words or 1312 bits. The whole column of configuration frames in the configuration matrix corresponds to a single type of the FPGA resources: Configuration Logic Blocks (CLBs), Block RAMs (BRAMs), and DSPs. The number of configuration columns of each type of the FPGA resources varies with the family and size of the FPGA device.

The frame is identified by the frame address. The frame address is composed of top/bottom bit, major address identifying the column, the row, and the minor address identifying the frame within the row.

The configuration memory of the SRAM based FPGA is susceptible to the SEUs. The SEU can cause the change in a bit of the configuration frame that corresponds to a particular FPGA resource. It may correspond to internal memory of the FPGA device (BRAM), or to the

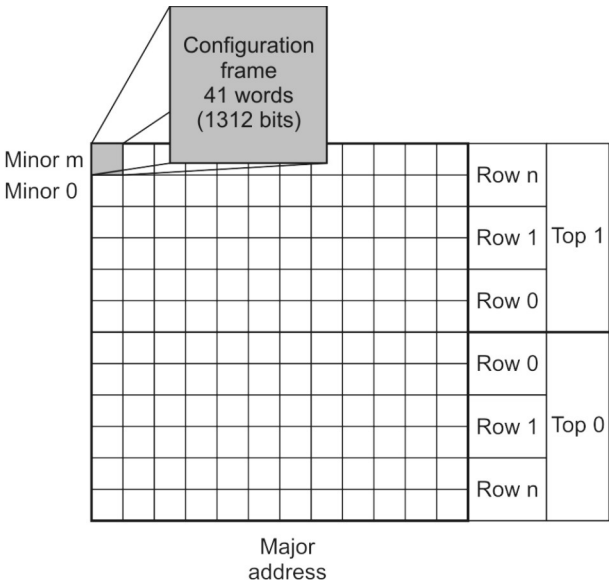

Figure 1: FPGA configuration matrix structure

DSP block (e.g. multiplier), or to the Configurable Logic Block (CLB), comprising Look-Up Tables (LUT) and flip-flops, or to the internal routing.

The changed configuration bit may manifest as an altered device. However the user design does not occupy whole FPGA and even when the affected bit corresponds to the used resource its effect might be masked. The configuration bits that correspond to the used FPGA resources are considered potentially critical.

A SEU can also affect other non-configurable parts of the FPGA device like Power-On-Reset circuit (POR), SelectMap or Internal Configuration Port (ICAP), Digital Clock Managers (DCMs), or global signals (Global Wire Enable, Global 3-State, etc.). Such faults can cause regional or device wide failure. These SEUs are referred as Single Event Functional Interrupts (SEFI) [18].

3 Error-recovery mechanism

The errors in the FPGA configuration memory can be recovered by the reconfiguration. The straightforward approach is to reconfigure the whole device however such approach requires long configuration time and is inefficient. The alternative is to reconfigure only the affected portion of the configuration using partial reconfiguration. The reconfiguration can be implemented using external reconfiguration port (either SelectMap or JTAG) and controller, or using internal configuration port (ICAP) and internal reconfiguration controller. The fault-free configuration can be stored externally in the nonvolatile memory or the reconfiguration content can be determined if the error location is known.

3.1 Error detection and recovery method

In Xilinx Virtex 4 and Virtex 5 FPGA families each configuration frame contains an Error Correction Code (ECC) signature. ECC signature consists of 12 parity bits that can locate single bit fault within the frame and detect double bit faults. The error within the frame can be determined by comparing the stored ECC signature with the newly calculated ECC. The comparison is called the syndrome value. First 11 bits of a syndrome value identify the location of a single erroneous bit within the frame while the last bit indicates the presence of a double error in the frame.

Xilinx Virtex 4 and Virtex 5 FPGA families have an embedded ECC circuit that calculates a frame syndrome value during the read-back of the frame. The Table 1 summarizes possible syndrome values and the corresponding error status.

Table 1: Syndrome value to error status mapping.

Syndrome bit 11	Syndrome bits 10 to 0	Error status
S[11]=0	S[10:0]=0	No error
S[11]=0	S[10:0]>0	Single bit error S[10:0] location
S[11]=1	S[10:0]=0	Single bit error in last parity bit
S[11]=1	S[10:0]>0	Double bit error

The error in the FPGA configuration memory can be detected by continuously reading configuration frames and monitoring the syndrome value. When the syndrome value is non-zero there are two possibilities:

1. Single fault is detected and the original frame content can be restored using syndrome value.
2. Double fault is detected and the original frame cannot be restored. To recover the affected device the frame must be restored using externally stored fault free configuration.

3.2 Error recovery mechanism implementation

Proposed error recovery mechanism consists of an Internal configurations Port (ICAP), a frame ECC device, one dual-port Block RAM (BRAM) and control logic. Its architecture is depicted in Figure 2.

The FPGA device can be reconfigured by applying configuration commands in the configuration registers of the device. The ICAP device has a direct access to these registers and it is used for both reading and writing the content of the configuration frames.

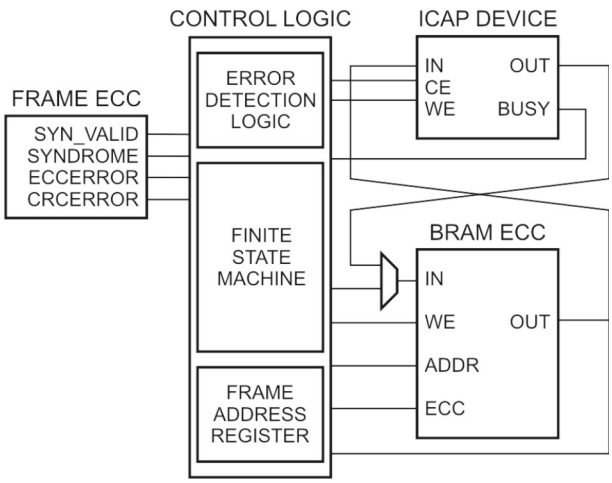

Figure 2: The error recovery mechanism architecture

The command sequences for both reading and writing of configuration frame are predefined and stored in the Block RAM. During the iteration an address of the current frame is determined and inserted in the read command sequence. The sequence is then transferred to the configuration registers by using ICAP device.

The frame ECC device is used to detect and locate configuration errors within the frame content. It works in parallel with the ICAP device during the read operation. While ICAP device reads the frame content the frame ECC device calculates the syndrome value.

The internal Block RAM is used to store the configuration command sequences as well as the content of the current configuration frame, which is used for error recovery if the frame is faulty. Since BRAMs are also susceptible to SEUs, strengthen BRAM configuration with internal ECC option is used.

The control logic monitors error detection process and triggers the frame reconfiguration, if an error is detected. When double error is detected an external alarm is triggered. External alarm can be used for external reconfiguration of the device. The control logic consists of a Finite State Machine (FSM), a frame address register, and error detection logic. The frame address register holds major address, top/bottom bit, row, and minor address and its value is updated when frame address is needed. The error detection logic examines the syndrome value, and corrects the frame content in the BRAM and triggers the reconfiguration if an error occurred.

3.3 Error recovery mechanism operation

The operation of the error recovery mechanism is controlled by a Finite State Machine (FSM) depicted in Figure 3.

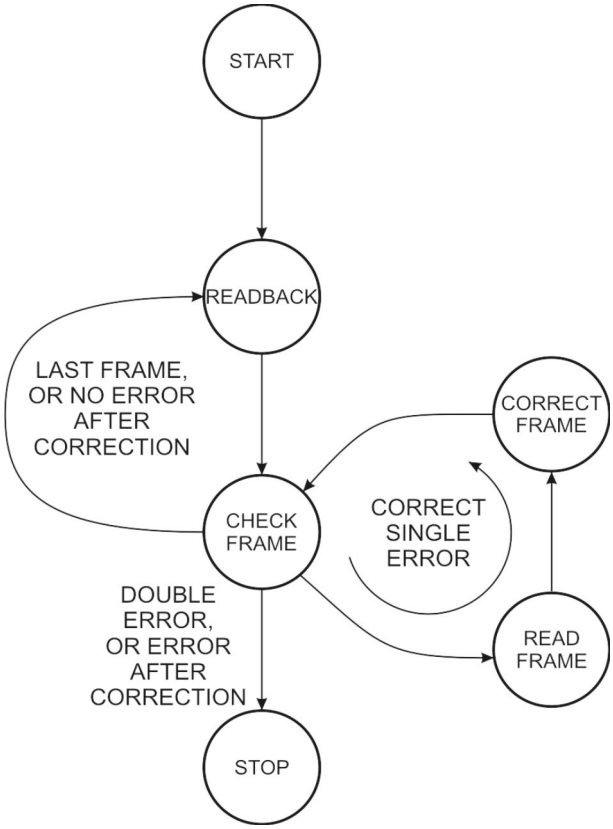

Figure 3: The error recovery state machine diagram

The recovery process starts in the state Start, where internal registers and signals are initialized.

In the state Readback the reading of the current frame register is initiated by issuing the readback commands with current frame address.

In the state Check Frame the syndrome value of current configuration frame, determined by the frame ECC device is inspected by the error detection logic. SYNDROMEVALID signal indicate the completion of the frame check operations. There are several options depending on the syndrome value:

1. The syndrome value is zero. This indicates that current frame is fault free. If the current frame is the last frame of the FPGA device, the register is initiated to the first frame, otherwise the frame address is incremented to the next frame address. The next frame read is started in Readback state.
2. If a single error is detected the erroneous frame is read in the state Read and its content stored in the BRAM. According to the syndrome value the frame is reconfigured with the corrected content in BRAM in the Correct frame state. Additional Check Frame is initiated. In the case of two consecutive failures of the same state, the mechanism switches to Stop state where an alarm is triggered.

3. In the case of double faults the error is detected but cannot be corrected. The mechanism switches to the state Stop, where an alarm is triggered.

In the case of multiple errors the operation of the mechanism is unreliable. However the probability of multiple errors in the same configuration frame is negligible.

4 Hardware implementation comparison

The internal error recovery mechanism was implemented using Virtex 4 and Virtex 5 FPGA families. The hardware overhead as well as the error recovery time of our error recovering mechanism was compared with other reported mechanisms.

The implementation in Virtex 4 FPGA family was compared with the error recovering mechanism described by Heiner *et al.* [16]. The comparison is presented in Table 2.

Table 2: Virtex 4 implementation comparison

Virtex 4 XC4VLX15	Our mechanism	Heiner et al. [16]
Slice	176	736
Flip-flops	118	680
BRAM	2	2
Worst-case Error detection time	1.477ms	~1.5ms
Worst case Error correction time	2us	~24.0ms
Worst-case Error recovery time	1.479ms	~25.5ms

The controller of their error correction mechanism is 8 bit PicoBlaze microprocessor. While PicoBlaze microprocessor requires relatively little FPGA resources, it is still significantly bigger than our FSM controller. On the other hand a microprocessor offers additional features, like easier monitoring and debugging through communication devices. While such features ease the development of the system, they require additional FPGA resources which in turn reduce the reliability of the recovery mechanism.

The error detection is performed using a device readback and achieve comparable error detection time, however they do not have the address of the erroneous frame. Therefore, when an error occurs, they have to perform an additional frame by frame check to determine the location of the error. This leads to a large error correction time.

The Virtex 5 implementation was compared to the Xilinx error recovery mechanism described by Chapman [17]. The results are shown in Table 3.

Table 2: Virtex 4 implementation comparison

Virtex 5 XC5VLX30	Our mechanism	Chapman [17]
Slice	72	172
Flip-flops	115	321
BRAM	1	1
Worst-case Error detection time	2.261ms	2.261ms
Worst case Error correction time	2µs	125µs
Worst-case Error recovery time	2.263ms	2.386ms

The controller of the Xilinx error correction mechanism is 8 bit PicoBlaze microprocessor and occupies about three times more FPGA resources than our mechanism.

Virtex 5 devices have an additional dedicated device readback CRC which can perform a continuous readback of the FPGA device. The recovery mechanism [17] employs readback CRC device to detect errors and the error-detection time is the same as error-detection time of our recovery mechanism. However the error correction time is four times longer since it is controlled by 8-bit PicoBlaze processor in contrast to our 32-bit architecture.

5 Fault-emulation experiment

In order to assess the reliability of the proposed recovery mechanism an error emulation environment was developed. Bit-flip errors are injected into the configuration memory of the targeted bit of the FPGA configuration. This is achieved by dynamic partial reconfiguration with a circuit similar to our error-recovery mechanism. It is additionally connected to the external computer, which controls the placement of the bit-flip error and gathers information of the error recovery. This error injection approach enabled us to inject configuration faults at a precise location within the FPGA configuration memory.

The fault emulation experiment was performed by placing both fault injection circuit and error recovery mechanism in the target FPGA device. Additional measures have been taken in order to prevent simultaneous access to the FPGA reconfiguration registers. The hardware structure of the fault emulation and error recovery mechanism assessment is given in Figure 4. The computer behaves as a fault emulation controller,

external memory, and external reconfiguration device. It also acts as an external watchdog timer for the error recovery mechanism.

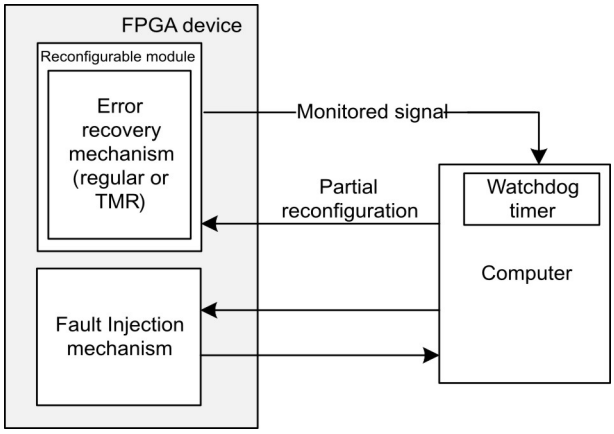

Figure 4: The error recovery mechanism architecture

Faults are injected only into the configuration memory of the FPGA device which corresponds to the error recovery mechanisms. Errors in other parts of the configuration memory do not affect the operation of the error recovery mechanism and are corrected during configuration memory recovery cycle. Furthermore only single recovery cycle is performed since the error should be detected and repaired in first cycle.

The fault emulation process starts by configuring the FPGA device with fault injection mechanism and fault-free error-recovery mechanism, which is being stopped. Then following steps are performed for each fault from the fault list:

- Fault is injected into error-recovery mechanism configuration memory via fault injection circuit.
- The error-recovery mechanism is started and it stops after its repair cycle finishes. Depending on how injected fault affects the performance of the error-recovery mechanism, there are following situations:
  - The error-recovery mechanism performs correctly and the error is corrected. This is detected by external watchdog timer, which is reset. The fault injection mechanism verifies that the fault was corrected. The computer logs the result and the fault emulation proceeds with the next fault from the fault list.
  - The error-recovery mechanism fails, but the failure is detected by the watchdog timer. Error-recovery mechanism did not complete the recovery cycle and the watchdog timer was not reset. Such failures are still manageable by the error-recovery mechanism if an external watchdog timer is added. The computer logs the result and reconfigures the error-recovery mechanism using stored partial configuration image. The fault emulation proceeds with the next fault from the fault list.
  - The error-recovery mechanism fails, however the failure is not detected by the watchdog timer. This indicated that error-recovery mechanism completed the recovery cycle and reset the watchdog timer, but it didn't correct the failure. The situation is detected by fault-injection mechanism and logged by the computer. The computer reconfigures the error-recovery mechanism from stored partial configuration image since the error-recovery mechanism could introduce additional fault into its configuration. The fault emulation proceeds with the next fault from the fault list.

5.1 Fault-emulation results

Fault emulation experiment was performed on Virtex 5 XC5VLX110T device. In the fault-emulation experiment 181056 faults were injected into 138 configuration frames occupied by error-recovery mechanism. Only 9177 faults affected the operation of the error-recovery mechanism. 719 of undetected faults can be further detected and corrected by an external watchdog timer and external reconfiguration device. These faults were further classified by their corresponding type of the FPGA resource. The results of the fault-emulation are presented in Table 4.

Table 4: Fault-emulation results of an error-recovery mechanism implemented on Virtex 5 FPGA device

181056 injected faults	Unrecoverable Errors	Detected by watchdog
Routing	7180	554
LUT content	1429	122
CLB configuration	509	37
BRAM configuration	59	6
Whole mechanism	9177	719

5.2 Reliability estimation

The reliability of Xilinx devices in the atmospheric conditions is being investigated by an ongoing Rosetta experiment [2]. Current reliability estimation of Xilinx FPGA devices can be found in Xilinx device reliability report [20]. The SEU induced error rate is given in terms of Failure In Time (FIT) or Mean Time Between Two Failures (MTBF). The FIT is the number of failures that can be expected in 10^9 hours of operation.

Our experimental results were determined for Virtex 5 FPGA device. The nominal failure rate for Virtex 5 devices is 162 FIT/Mb according to [20]. According to experi-

ments shown in [21] the failure rate of the application can be estimated by the multiplication of the percentage of the critical bits acquired with fault emulation with the failure rate of the whole device.

Table 5: SEU reliability estimation for applications on Virtex 5 XC5VLX110T device

XC5VLX110T device	Number of critical bits	FIT (SEU/109 h)
Whole device	31.1 Mb	5039.8
Average design	3.42 Mb	554.4
Error-recovery mechanism	9177 b	1.49
Error-recovery With watchdog	719 b	0.12

Table 5 summarizes the reliability estimation of different design scenarios. The whole FPGA configuration of Virtex 5 XC5VLX110T device has over 5000 FIT. However an average design uses only portion of FPGA resources and only part of their corresponding configuration bits are critical to its operation. In the estimation 11% configuration bits are considered to be critical for the average application [22].

The fault-emulation experiment determined that 9177 configuration bits are critical which in turn gives a reliability estimation of 1.49 FIT. Applying an external watchdog timer with accompanying external reconfiguration device the reliability estimation is decreased to 0.12 FIT which is equivalent to MTBF of approximately 1 million years.

6 Conclusions

An error recovery mechanism for SRAM based FPGAs was developed. It is controlled by a finite state machine architecture providing the smallest reported hardware overhead and minimal recovery latency. It can be further hardened by the use of external watch dog timer.

The efficiency and performance of the proposed mechanism was evaluated by the fault evaluation environment.

Acknowledgments

This work has been performed in the Computer Systems Department at Jožef Stefan Institute and I would like to acknowledge the contributions of other colleagues working in this area.

It was supported by the Slovenian Research Agency under grant P2-0098.

References

1. E. Normand, "Correlation of inflight neutron dosimeter and SEU measurements with atmospheric neutron model," IEEE Trans. On Nuclear Science, vol. 48, no. 6, pp. 1996 - 2003, 2001..
2. A. Lesea, S. Drimer, J. Fabula, C. Carmichael, P. Alfke, "The Rossetta Experiment: Atmospheric Soft Error Rate Testing in Differing Technology FPGAs," IEEE Trans. on Device and Materials Reliability, vol. 5, no. 3, pp. 317 - 328, 2005.
3. J. B. Ferron, L. Anghel, R. Leveugle, "Analysis of configuration bit criticality in designs implemented with SRAM-based FPGAs," in Proc. of IEEE Symposium on Industrial Electronics and Applications (ISIEA), 2011, pp. 83-88.
4. E. Fuller, M. Caffrey, P. Blain, C. Carmichael, N. Khalsa, and A. Salazar, "Radiation test results of the virtex FPGA and ZBT SRAM for space based reconfigurable computing," in Proc. Military & Aerospace Applications of Programmable Logic Devices (MAPLD), 1999, pp. 1-8.
5. A. Lesea, "Continuing Experiments of Atmospheric Neutron Effects on Deep Submicron Integrated Circuits," Xilinx Corporation, 2009, WP286 (v1.0.1).
6. F. Lima Kastensmidt, L. Carro, R. Reis, Fault-Tolerance Techniques for SRAM-Based FPGAs, first ed., Dordrecht:Springer, 2006.
7. C. Carmichael, "Triple module redundancy design techniques for Virtex FPGAs," Xilinx Corporation, 2001, XAPP197 (v1.0).
8. F. Lima, C. Carmichael, J. Fabula, R. Padovani, and R. Reis, "A fault injection analysis of Virtex FPGA TMR design methodology," in: Proc. of the European Conference on Radiation and its Effects on Components and Systems, 2001, p. p. 275 – 282.
9. S. Rezgui, G.M. Swift, K. Somervill, J. George, C. Carmichael, and G. Allen, "Complex Upset Mitigation Applied to a Re-Configurable Embedded Processor", IEEE Trans. on Nucl. Sci., vol. 52, n. 6, pp 2468-2474, 2005.
10. C.C. Yui, G.M. Swift, C. Carmichael, R. Koga and J.S. George, "SEU Mitigation Testing of Xilinx Virtex II FPGAs", in: Radiation Effects Data Workshop Record, 2003.
11. C. Bolchini, A. Miele, and M. D. Santambrogio, "TMR and Partial Dynamic Reconfiguration to mitigate SEU faults in FPGAs," in: Proc. Defect and Fault Tolerance in VLSI Systems, 2007, pp. 87-95.
12. W. Peterson, "Error-correcting codes," secondnd ed., Cambridge, The Mit Press, 1980.

13. M. Ceschia, A. Paccagnella, S.C. Lee, C. Wan, M. Bellato, M. Menichelli, A. Papi, A. Kaminski, J. Wyss, "Ion beam testing of ALTERA APEX FPGA's," in Proc. Radiation Effects Data Workshop Record (NSREC), 2002, pp. 45-50.
14. H. Asadi, M. B. Tahoori, B. Mullins, D. Kaeli, K. Granlund, "Soft Error Susceptibility Analysis of SRAM-Based FPGAs in High-Performance Information Systems," IEEE Trans. on Nuclear Science, vol. 54, pp. 2714-2726, 2007.
15. C. Carmichael, M. Caffrey, A. Salazar, "Correcting single-event upsets through virtex partial configuration," Xilinx Application Notes XAPP216 (vl.0), 2000.
16. J. Heiner, N. Collins, M. Wirthlin,"Fault Tolerant ICAP Controller for High-Reliable Internal Scrubbing," In Proc. IEEE Aerospace Conference, 2008, pp. 1-8.
17. K. Chapman, "SEU strategies for Virtex-5 Devices," Xilinx Appplication Notes XAPP864 (v2.0), 2010.
18. C. Carmichael, C. Wei Tseng, "Correcting Single-Event Upsets in Virtex-4 FPGA Configuration Memory," Xilinx Application Notes XAPP1088 (vl.0), 2009.
19. U. Legat, A. Biasizzo, F. Novak, "SEU recovery mechanism for SRAM-based FPGAs," IEEE Trans. on Nuclear Science, vol. 59, pp. 2562-2571, 2012.
20. Xilinx, "Device Reliability Report", Xilinx User Guides UG116 (v5.12) 2011.
21. R. Velazco, G. Foucard, P. Peronnard, "Combining Results of Accelerated Radiation Tests and Fault Injections to Predict the Error Rate of an Application Implemented in SRAM-Based FPGAs," IEEE Trans. On Nuclear Science, vol. 57, no. 6, pp. 3500 – 3505, 2010.
22. K. Chapman, "Virtex 5 SEU Critical Bit Information," Xilinx Documentation (v1.0), 2010. Avalable: www.xilinx.com

Arrived: 18. 11. 2013
Accepted: 26 .11 2013