Self-reference Scrubber for TMR Systems Based on Xilinx Virtex FPGAs

Ignacio Herrera-Alzu and Marisa López-Vallejo*

Department of Electronics Engineering
E.T.S.I. Telecomunicación
Universidad Politécnica de Madrid, Madrid, Spain
{iherrera,marisa}@die.upm.es

Abstract. SRAM-based FPGAs are sensitive to radiation effects. *Soft errors* can appear and accumulate, potentially defeating mitigation strategies deployed at the Application Layer. Therefore, Configuration Memory scrubbing is required to improve radiation tolerance of such FPGAs in space applications. Virtex FPGAs allow runtime scrubbing by means of dynamic partial reconfiguration. Even with scrubbing, intra-FPGA TMR systems are subjected to common-mode errors affecting more than one design domain. This is solved in inter-FPGA TMR systems at the expense of a higher cost, power and mass. In this context, a self-reference *scrubber* for device-level TMR system based on Xilinx Virtex FPGAs is presented. This *scrubber* allows for a fast SEU/MBU detection and correction by peer frame comparison without needing to access a *golden* configuration memory.

Keywords: SRAM-based, FPGA, EDAC, SEU, MBU, SEFI, TMR, Scrubber, Scrubbing, Readback, Reconfiguration, Configuration Memory, Reliability.

1 Introduction

When used in space applications, electronic devices are exposed to an ionizing radiation that is orders of magnitude higher than the one received at sea level. This affects the behaviour of semiconductors, inducing both a long term cumulative degradation as well as instantaneous damage or Single Event Effects (SEE). SEE can be reversible (*soft errors*) or destructive (*hard errors*).

Xilinx Virtex FPGAs are also sensitive to these effects. Tolerance to both long term degradation and destructive effects has been improved in Virtex-4QV and Virtex-5QV series by hardening the technology. For example in 4QV series, Total Ionization Dose (TID) of 300 krad(Si) and Single Event Latchup (SEL) immunity has been reported [1]. However, their SRAM-based Configuration Layer is still susceptible to several types of *soft errors*, namely Single Event Upsets (SEU),

* This work has been funded by CICYT Project TEC2009-08589.

Multiple Bit Upsets (MBU) and Single Event Functional Interrupts (SEFI). 4QV series characterization for SEU and SEFI has also been reported in [1], but an equivalent report for 5QV series has not been published yet.

When high-energy particles hit the Virtex FPGAs, SEUs can appear either in Application Layer (memory elements driven by users application) or in the underlying Configuration Layer (logic and routing resources set by configuration SRAM). When in Application Layer, SEU effects can be persistent or transient depending on whether the affected logic has memory or is memoryless. When in Configuration Layer, SEUs can affect logic or routing resources, and their effects are persistent until a reconfiguration restores their initial state. A particular case are SEFIs affecting the configuration logic, which cannot be repaired without re-initializing the FPGA.

A SEU in Configuration Layer may propagate as a functional failure in the Application Layer, independently of the user functionality implemented. Indeed a SEU in a routing resource may induce single or multiple errors in the Application Layer. This situation is dramatic considering that about 90% of the configuration bits are linked to routing resources [2], and that configuration memory density keeps on growing in every new FPGA generation, thus increasing the MBU cross-section [3]. Complementary SEU tolerance techniques have been incorporated to enable the use of SRAM-based FPGAs in harsh radiation environments: Error Detection And Correction (EDAC, such as TMR and majority voting) at Application Layer, and memory scrubbing at Configuration Layer. The reliability improvement of TMR systems when scrubbing is applied has been deeply analyzed in [4].

In this paper we propose a simple *scrubber* architecture for an inter-FPGA TMR system. As opposed to techniques based on embedded Error Correction Codes (ECC) [5], multiple errors per *frame* can be detected and corrected. The faulty *frame* is fully restored by taking advange of the redundant configuration data. And, as opposed to previously reported systems, it does this without nominally needing to access a *golden* configuration memory or a reference EDAC code memory [6]. The proposed *scrubber* also allows a fast SEFI detection by periodically monitoring the already known SEFI symptoms. Because radiation environment is variable, it allows dynamic scrub rate adjustment, thus reducing power consumption to the minimum necessary. Finally, the proposed *scrubber* concept is universal for Xilinx FPGAs from Virtex-4 on, only some control logic specificities need to be tailored for each FPGA series.

The paper is organized as follows. Section 2 discusses the scrubbing techniques for Xilinx Virtex FPGAs. Section 3 introduces the proposed self-reference *scrubber* for a TMR system. Section 4 presents the simulation results for the proposed *scrubber*. Finally, Section 5 presents the conclusions of this paper and outlines the future work.

2 Scrubbing Xilinx Virtex FPGAs

Xilinx Virtex FPGAs can be dynamically reconfigured after the initial power-on configuration practically an unlimited number of times. Dynamic global

reconfiguration affects the whole programmable fabric, and application execution is interrupted. Dynamic partial reconfiguration only affects part of the fabric while the rest operates without interruption. The dynamic partial reconfigurability is a unique feature of Xilinx Virtex FPGAs and can be exploited in different ways. In space applications it is particularly useful as it allows, on one hand, Configuration Memory scrubbing for SEU mitigation and, on the other, Application Layer in-flight reconfigurability [7].

2.1 Internal versus External Scrubbing

Configuration Memory scrubbing is driven by a Configuration Manager (informally known as *scrubber*), typically implemented in a radiation-hardened device external to the FPGA, either HW or SW driven. *Scrubbers* internal to the FPGA may also be implemented by using the Virtex primitives ICAP and FRAME_ECC [5],[8]. These primitives are theirselves implemented in the programmable fabric, so they are also susceptible to radiation. Although fault-tolerant implementations of internal *scrubbers* have been proposed [9], external scrubbers are typically more reliable [10]. In this work, only external *scrubbers* will be considered.

All Virtex FPGA configuration ports support external scrubbing. Among them, the 8-bit parallel SelectMAP (SMAP) interface is typically used as a good trade-off between speed and pinout penalty. For example, in Virtex-4QV series this interface can be clocked up to 80MHz. At this rate it is possible to fully configure the largest 4QV (XQR4VLX200) in about 70ms, excluding block RAM contents.

2.2 Configuration Memory Structure

For Virtex-4 and subsequent FPGA families, the minimum configurable element in the Configuration Memory is a *frame*, which is a set of 41 words of 32 bit each. The *frames* do not cover the whole programmable fabric height, but only the height of a memory *row*. Each FPGA has an specific number of *rows* that cover the whole fabric area. The *rows* are grouped in two halves, so called *top* and *bottom*. In addition, *frames* in each *row* are grouped into *columns* with variable sizes which depend on the *column type*. With such memory structure it is possible to define multiple bidimensional reconfigurable regions. The number of *frames*, *rows*, *columns* and *column types* are specific for each FPGA of the Virtex series. Fig. 1 depicts the generic Configuration Memory structure described above.

In addition to the programmable fabric, the Virtex Configuration Layer also includes configuration control logic and associated registers for monitoring and control. Among these 32-bit registers, the Frame Address Register (FAR) allows addressing any individual *frame*. The FAR is composed of several fields, with different organization depending on the Virtex FPGA generation. The FAR resets to the left-most *frame* of the *top-row 0*, and auto-increments right-up first. After the top right-most *frame*, the FAR goes back to the left-most *frame* of the *bottom-row 0* and auto-increments right-down.

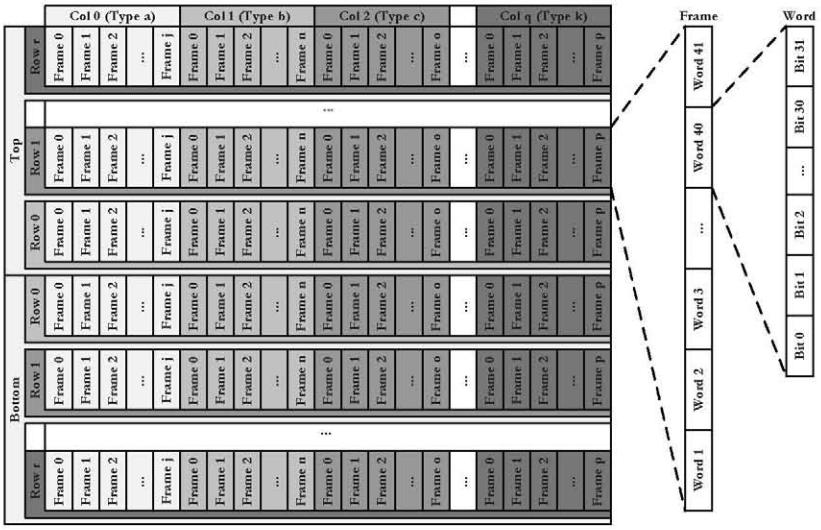

Fig. 1. Configuration Memory Sctructure for Xilinx FPGAs from Virtex-4 on

2.3 Readback and Scrubbing Configuration Memory

Two main strategies exist for SEU/MBU mitigation: open-loop and closed-loop scrubbing. Open-loop scrubbing (or *blind scrubbing*) consists on a preventive and cyclic SEU/MBU correction (i.e. without prior detection) through dynamic partial reconfiguration of the full Configuration Memory. The configuration data is retrieved from a *golden* data source, typically a radiation-hardened non-volatile memory, and written through the configuration port. As long as scrub rate is faster than the expected SEU rate, SEUs will not accumulate. An example of an open-loop *scrubber* for Virtex-4 is [11].

Closed-loop scrubbing instead is a two step process. First the configuration *frames* are cyclically read back for SEU/MBU detection. Only if an error is detected, the correction process is initiated. There are different possibilities for SEU/MBU detection and correction. The simplest one is the comparison of readback *frames* against their *golden* counterparts. In case they differ, the *golden frame* is used to scrub the faulty *frame* through the configuration port. Similarly to the open-loop scrubbing, this solution implies the permanent availability of the *golden* configuration memory, with the corresponding power consumption. Another possibility, which is available in Virtex-5 FPGAs, is to check the CRC computed during the readback process by means of an embedded circuitry. In case error is detected, the *frame* Error-Correcting Code (ECC) bits are used to locate the error. Once located, the faulty *frame* is scrubbed with a corrected content through the configuration port. However in this last case, the process is relatively complex and only single errors per *frame* can be corrected.

In Virtex FPGAs, Configuration Memory readback is done by reading *frame* 32-bit words on a byte basis (in case of using 8-bit wide SMAP) from the Frame

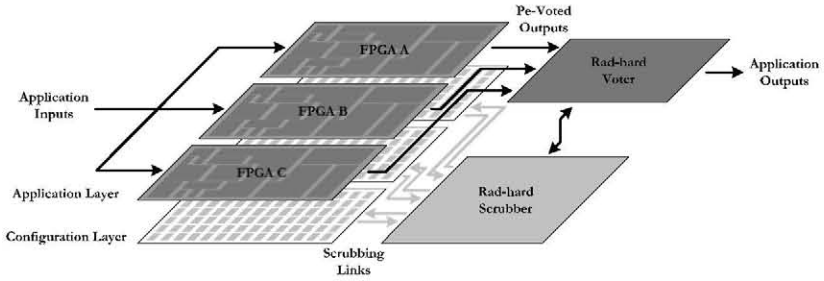

Fig. 2. Inter-FPGA TMR System

Data Register Output (FDRO). Likewise, scrubbing is done by writing *frame* 32-bit words on a byte basis to Frame Data Register Input (FDRI). After a complete *frame* is read back or scrubbed, the FAR auto-increments as described in Section 2.2. This exempts the *scrubber* from continuously updating FAR before each access, thus reducing the processing overhead.

3 Self-Reference *Scrubber* for Inter-FPGA TMR Systems

3.1 Inter-FPGA TMR Systems

Inter-FPGA TMR, or device-level TMR, implies the use of independent FPGAs for each of the three design domains. A diagram of such system is depicted in Fig. 2. A three-way external *scrubber* is represented, interfacing the three SMAP ports at the Configuration Layer. A three-way external voter is also represented, voting the three application data streams at the Application Layer. These two elements are critical and must be implemented in radiation-hardened devices, either jointly or separated. Strategies for inter-operation between both elements will be covered in a future work, but are out of the scope of this paper.

This architecture is used for high reliability and availability systems where common-mode errors affecting more than one design domain must be avoided. In intra-FPGA TMR systems, and as highlighted in Section 1, this is more likely to happen as configuration memory densities increase [3]. In addition, inter-FPGA TMR is required for those applications that cannot afford a potential outage due to SEFI.

3.2 *Scrubber* General Description

In this work we propose a closed-loop *scrubber* for TMR systems, whose diagram is shown in Fig. 3. The *scrubber* is governed by the FSM, which in turn manages the Readback and Scrubbing Controllers. The SMAP interfaces handle the bidirectional links with the FPGAs. The *Frame* Buffers are 41 words each, i.e. 1,312 bits each. The SEFI Detection block is monitoring some specific FPGA pins and registers and asynchronously alerts the FSM in case SEFI is found. Finally, the *Golden* Configuration Memory is mainly used for power-on configuration and SEFI recovery.

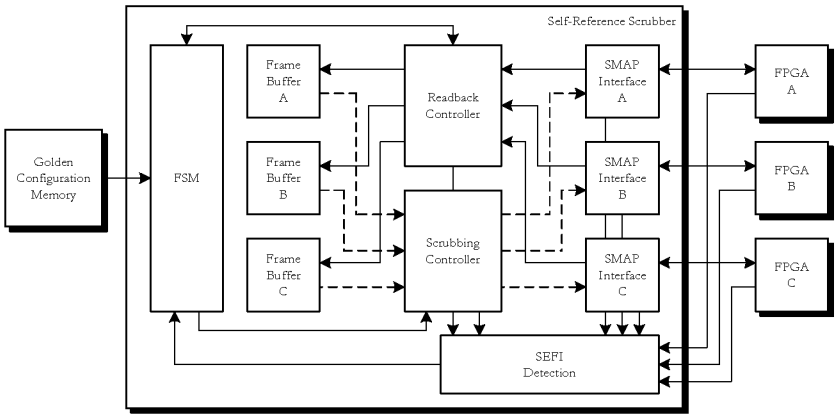

Fig. 3. Self-Reference *Scrubber* Diagram

FSM and *Scrubber* Operation. A simplified state diagram for the *scrubber* FSM is shown in Fig. 4. In a nominal operation, three identical configuration *frames* are read back from the FPGAs and stored in their corresponding *Frame* Buffers. At the same time, *frame* triplets are compared on a byte basis as they are read through SelectMAP interface in order to detect SEU/MBUs. At the end of each *frame* readback, a flag indicates if any mismatch was found in each particular domain. This implies that the faulty element at system level is the Frame In Error (FIE). If *frames* are found identical, readback resumes and new *frames* are read back. If one FIE flag is raised, the *dirty frame* is scrubbed with a correct *frame* from one of the other two *Frame* Buffers. After that, the FIE flag is cleared and the readback for the three FPGAs is resumed.

There is one scenario in which SEU/MBU will go undetected due to a common-mode error: when three particles hit each FPGA in exactly the same *frame*, the same bits within the *frame*, and within the same scrub cycle. If instead of three particles is two, the two *dirty frame* will be taken as good and the error will be propagated to the third one. However, the probability of such remote events is extremely low. The also remote possibility of having simultaneously three different SEU/MBU location in the three *frames* under comparison can be recovered by scrubbing the three *frames* with data from the *golden* memory. Apart from this case and for power-on configuration and SEFI recovery, the *golden* memory can be almost permanently unpowered.

In case a persistent FIE is found (i.e. the same *frame* address is found *dirty* in consecutive readbacks), the *dirty* FPGA is passivated and the sytem turns to DMR mode. In this mode, FIE detection is performed by comparing bytes from two domains. Again, if the *frames* are identical the readback resumes and new *frames* are read back. If they are not, the two *frames* are considered *dirty* and are scrubbed with data from the *golden* memory at the expense of higher power and slower operation. After that, readback for the two FPGAs is resumed.

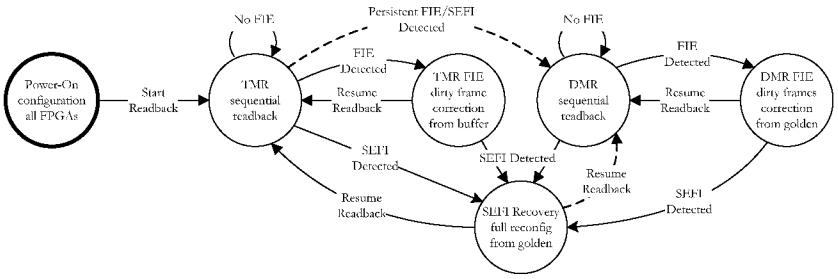

Fig. 4. Self-Reference *Scrubber* FSM States

As highlighted in Section 1, one of the main advantages with respect to other embedded SEU/MBU correction techniques proposed, is that multiple errors in a single *frame* can be detected and corrected. This is because the full *dirty frame* is scrubbed regardless of how many bits in error it may have.

Readback and Scrubbing Controllers. These blocks translate high level commands from the FSM into atomic commands for the SMAP Interfaces. In addition, the Readback Controller compares each byte triplet for the incoming readback *frames*. Scrub rate can be dynamically adjusted by FSM in order to adapt to the changing radiation conditions and to limit *scrubber* power consumption. Previous publications state that scrub rate should be at least ten times the expected SEU rate [13],[14]. In any case, no more than one FIE should be detected across all three FPGAs within a single scrub cycle. Note that this does not guarantee that TMR protection at Application Layer is not defeated, as SEUs in routing resources may induce multiple errors. If more than one FIE is detected within a scrub cycle, or FIEs are detected in consecutive scrub cycles at any given time, one should consider increasing the scrub rate. A shorter scrub cycle improves the Mean Time To Failure (MTTF) figure [4]. Therefore, the trade-off between reliability and power consumption must be managed.

SEFI Detection and Recovery. Concurrent SEFI detection is achieved by several means in accordance with [1] and [12]. Configuration pins are continuously monitored for Power-On-Reset (POR) and SMAP SEFIs. Configuration control registers are cyclically polled to detect SEFI symptoms, such as abnormal values or flags indicating that the control logic has reached an invalid state.

In case SEFI is detected while in TMR mode, the failing FPGA is fully reconfigured from *golden* memory while the other two are still operating. No power-cycle is needed, according to [12]. During this time, the readback controller seeks for mismatches between the two available *frames* (DMR mode). Once the failed FPGA is available again, its FAR is synchronized with the other two to resume readback in lockstep in TMR mode. In case of persistent FIE/SEFI on the same FPGA (i.e. present after consecutive reconfiguration cycles), the failing FPGA is passivated and the sytem turns to DMR mode. Finally, in case SEFI is detected

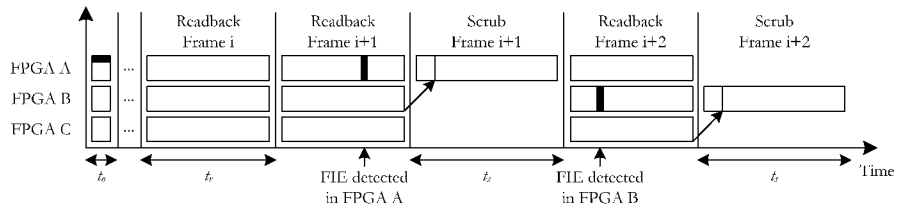

Fig. 5. Self-Reference *Scrubber* Simulation Timeline

while in DMR mode, the failing FPGA is in turn fully reconfigured. During this time, FIE detection is suspended as it works in purely simplex mode. Once the failed FPGA is available again, the system returns to DMR mode.

4 Simulation Results and Discussion

A VHDL description for the system shown in Fig. 3 was simulated for the purpose of validating the self-reference *scrubber* concept. For the FPGAs a cycle-accurate simulation model from previous work was used [15]. The model represents functionally the Configuration Layer of a XQR4VLX200 FPGA, implementing both the 39,120 configuration *frames* and the associated control logic. Such model allows for fault injection at random configuration bits of any of the three Configuration Memories and at random instants within the scrub cycle. Some of the known SEFI effects have also been modelled.

The simulation timeline for the basic readback-scrubbing operation in TMR mode is shown in Fig. 5. The *frame* readback time is t_r and the *frame* scrub time is t_s. The overhead time required to poll some of the configuration registers to detect SEFIs and to initalize some other registers before each scrub cycle is t_o. An overhead time is also included in t_s, as a single *frame* scrubbing also requires initialization of some control registers. The simulated SelectMAP clock frequency is 20MHz as an initial trade-off between reliability and power.

The main *scrubber* simulated performance is summarized in Table 1. Besides Simulated Time, CPU Time for the full readback and configuration cycle simulations, running on an processor at 3.2GHz, is shown for reference.

CPU Time is more than reasonable taking into account that three Configuration Memories with more than 51Mbit each are simulated, and that SelectMAP interface is simulated with clock cycle accuracy. The table shows that t_r and t_s are in the same order of magnitude, as they are mainly driven by the SelectMAP clock cycle. The main difference is due to the overhead time included in t_s. If no FIE is found within a complete scan, the full readback cycle takes $t_{r,f} \approx t_o + (33{,}720 \times t_r)$, where 33,720 is the number of *frames* to be read back (after excluding the BRAM areas). If SEFI is found, the full reconfiguration cycle takes, for one or more FPGAs running concurrently, $t_{s,f} \approx 33{,}720 \times t_s$ provided that the interface with the *golden* memory is at least as fast as the SelectMAP

Table 1. Self-Reference *Scrubber* Simulation Results

Scrubber Performance	t_o	t_r	t_s	$t_{r,f}$	$t_{s,f}$
Simulated Time	8.0 μs	8.2 μs	12.1 μs	276.5 ms	276.5 ms
CPU Time	-	-	-	49 s	52 s

Note: SelectMAP Clock Cycle = 50ns

interface. The simulation results for $t_{r,f}$ and $t_{s,f}$ confirm the validity of the simulation model for the purpose of evaluating the *scrubber* concept, and provides a better insight of the *scrubber* performance.

5 Conclusions and Future Work

A self-reference *scrubber* for inter-FPGA TMR systems based on Xilinx Virtex FPGAs has been presented in this paper. This *scrubber* allows for a fast SEU/MBU detection by peer *frame* comparison, and correction by scrubbing the *dirty frame* with one of the other two. Scrubbing relies on dynamic partial reconfiguration. SEFI can be detected by polling internal registers or FPGA pins, and recovered by means of full reconfiguration. A *golden* configuration memory is not needed for SEU/MBU mitigation, but it is for SEFI recovery.

With the proposed *scrubber* the system redundancy (and reliability by extend) may vary over time. From initial TMR mode, persistent errors will lead the system to a DMR mode with reduced reliability. In DMR mode, *golden* memory is required for SEU/MBU mitigation. In addition, the scrub rate can also adapt to dynamic radiation environment. The *scrubber* concept has been validated with a simulation model for the FPGA Configuration Layer from a previous work, and some simulation results have been presented. The *scrubber* performance during readback and scrub cycles are in line with expectations.

As future work it is foreseen to prototype the *scrubber* in hardware. This will allow to characterize *scrubber* performance and power consumption in different redundancy and scrub rate configurations. Hardware-based fault injection via SelectMAP is planned for these experiments, and convergence with simulation-based results will also be analyzed. Future work is also planned to develop strategies for inter-operation of *scrubber* and voter modules, as well as for user data recovery after a full FPGA reconfiguration.

References

1. Allen, G., Swift, G., Carmichael, C.: Virtex-4 VQ static SEU characterization summary. Jet Propulsion Laboratory, Pasadena, CA. National Aeronautics and Space Administration (2008)
2. Sonza Reorda, M., Sterpone, L., Violante, M.: Multiple errors produced by single upsets in FPGA configuration memory: a possible solution. In: European Test Symposium, pp. 136–141. IEEE Computer Society, Los Alamitos (2005)

3. Quinn, H., Morgan, K., Graham, P., Krone, J., Caffrey, M., Lundgreen, K.: Domain crossing errors: Limitations on single device triple-modular redundancy circuits in Xilinx FPGAs. IEEE Transactions on Nuclear Science 54(6), 2037–2043 (2007)
4. Ostler, P.S., Caffrey, M.P., Gibelyou, D.S., Graham, P.S., Morgan, K.S., Pratt, B.H., Quinn, H.M., Wirthlin, M.J.: SRAM FPGA reliability analysis for harsh radiation environments. IEEE Transactions on Nuclear Science 56(6), 3519–3526 (2009)
5. Xilinx: SEU : Strategies for Virtex-5 Device, `http://www.xilinx.com/support/documentation/application_notes/xapp864.pdf`
6. Lanuzza, M., Zicari, P., Frustaci, F., Perri, S., Corsonello, P.: A self-hosting configuration management system to mitigate the impact of Radiation-Induced Multi-Bit Upsets in SRAM-based FPGAs. In: IEEE International Symposium on Industrial Electronics, pp. 1989–1994. IEEE, Los Alamitos (2010)
7. Osterloh, B., Michalik, H., Habinc, S.A., Fiethe, B.: Dynamic partial reconfiguration in space applications. In: NASA/ESA Conference on Adaptive Hardware and Systems, pp. 336–343. IEEE, Los Alamitos (2009)
8. Xilinx: Correcting Single-Event Upsets with a Self-Hosting Configuration Management Core, `http://www.xilinx.com/support/documentation/application_notes/xapp989.pdf`
9. Heiner, J., Collins, N., Wirthlin, M.: Fault tolerant ICAP controller for high-reliable internal scrubbing. In: IEEE Aerospace Conference, pp. 1–10. IEEE, Los Alamitos (2008)
10. Berg, M., Poivey, C., Petrick, D., Espinosa, D., Lesea, A., LaBel, K.A., Friendlich, M., Kim, H., Phan, A.: Effectiveness of internal versus external SEU scrubbing mitigation strategies in a Xilinx FPGA: Design, test, and analysis. IEEE Transactions on Nuclear Science 55(4), 2259–2266 (2008)
11. Berg, M.: The NASA Goddard space flight center radiation effects and analysis group Virtex 4 scrubber. In: Xilinx Radiation Test Consortium (XRTC) Meeting (2007)
12. Xilinx: Correcting Single-Event Upsets in Virtex-4 FPGA Configuration Memory, `http://www.xilinx.com/support/documentation/application_notes/xapp1088.pdf`
13. Quinn, H., Morgan, K., Graham, P., Krone, J., Caffrey, M.: A review of Xilinx FPGA architectural reliability concerns from Virtex to Virtex-5. In: 9th European Conference on Radiation and Its Effects on Components and Systems, pp. 1–8. IEEE, Los Alamitos (2007)
14. Adell, P., Allen, G.: Assessing and mitigating radiation effects in Xilinx FPGAs. Jet Propulsion Laboratory, Pasadena, CA. California Institute of Technology (2008)
15. Herrera-Alzu, I., López-Vallejo, M.: Cycle-Accurate Configuration Layer Model for Xilinx Virtex FPGAs. In: 13th European Conference on Radiation and Its Effects on Components and Systems. IEEE, Los Alamitos (2011)