LA-UR-

*Approved for public release;
distribution is unlimited.*

Title:

Author(s):

Intended for:

An Introduction to Radiation-Induced Failure Modes and Related Mitigation Methods For Xilinx SRAM FPGAs

Heather Quinn[1], Paul Graham[1], Keith Morgan[1], Jim Krone[1], Michael Caffrey[1], and Michael Wirthlin[2]
[1]ISR-3 Space Data Systems, Los Alamos National Laboratory, Los Alamos, NM, 87545 USA
[2]Brigham Young University, Provo, UT, 84602 USA

Abstract - *Using reconfigurable, static random-access memory (SRAM) based field-programmable gate arrays (FPGAs) for space-based computation has been an exciting area of research for the past decade. In comparison with traditional radiation-hardened electronics, these devices would allow spacecrafts to be more adaptive and responsive to changing mission needs. Unfortunately, all commercially available SRAM-based FPGAs have problems with the harsh radiation environment in space. This paper will provide an introduction to the potential radiation-induced faults and possible mitigation methods*

Keywords: FPGAs, failure modes, redundancy, scrubbing

1. Introduction

Over the past decade, several organizations have started using static random-access memory (SRAM) based field-programmable gate arrays (FPGAs) in space-based computational platforms [1], [2]. SRAM-based FPGAs can provide orders of magnitude speedup over traditional radiation-hardened microprocessors without the cost of manufacturing application-specific integrated circuits (ASICs). The reprogrammable nature of these devices allows them to be updated while on orbit, either with improved designs for current missions or to support new missions. Thus, as a result of this reconfigurability, the hardware's usable lifetime can be significantly extended.

The space environment has a very rich radiation environment of electrons, protons and heavy ions. Each orbit is characterized by an ion spectrum, where each ion has a corresponding energy spectrum. Unfortunately, all commercial SRAM devices are affected by these radiation environments, and SRAM-based FPGAs are no exception. Radiation-induced faults are particularly difficult with FPGAs since both the circuit and the circuit's state are stored in radiation-tolerant SRAM.

In this paper, we will present a taxonomy of possible radiation effects (Section 2) and a taxonomy of failure modes for Xilinx Virtex family SRAM-based FPGAs[1] in harsh radiation environments (Section 3). This paper will also provide an overview of fault mitigation methods (Section 4) that can be used to mitigate their effects. This paper is an update of our 2003 paper [3] and a shortened version of our Design Guide available through the FPGA Mission Assurance Center [4].

2. Basic Radiation Effects on Semiconductor Devices

Space-based electronics must be able to withstand the radiation environment and still be able to process reliably over the usable lifetime of the mission. While there are number of radiation-induced faults that could beset space-based hardware, most designers are concerned about total ionizing dose (TID) and single-event effects (SEEs). SEE can take many forms, such as single-event latchup (SEL), single-event transients (SETs), single-event upsets (SEUs), and single-event functional interrupts (SEFIs). These effects are discussed in greater detail below.[2]

2.1 Total Ionizing Dose

While deployed, the voltage and switching characteristics of transistors can change gradually with long-term exposure to protons and electrons [6]. Space-bound electronics are tested for the maximum amount of radiation the device can accumulate before it cannot be used reliably. The amount of radiation a deployed system will endure is dependent on the orbit and the mission duration. For a low earth orbit (LEO), 100 kRads of total ionizing dose should be sufficient for several years of reliable operation, which is a requirement that Virtex family devices meet.

2.2 Single-Event Effects

There are four primary forms of SEE that SRAM-based devices are concerned about: SEL, SEU, SEFI, and SET.

Document release number: LA-UR-08-0843. This work was funded by the Department of Energy through the Deployable Adaptive Processing Systems and Sensor-Oriented Processing and Networking projects, the Cibola Flight Experiment project at Los Alamos National Laboratory, NASA through the Reconfigurable Hardware in Orbit project under AIST contract #NAG5-13516, and the Air Force Research Laboratory under the FPGA Mission Assurance Center.

[1] In this paper, Xilinx Virtex family devices means Virtex-I, Virtex-II and Virtex-4 devices, unless specified. These three devices currently have space-qualified parts ("Q") versions that are manufactured on an epitaxial layer to prevent SEL and have military specification packaging. The Virtex-II Pro and the Virtex-5 device will not be touched on this paper, since there are not space-qualified part for these lines.

[2] For a more in-depth discussion of radiation effects, the authors suggest "The Radiation Effects Handbook" [5].

While there are a handful of SEE types that can damage a device, SEL is the predominant concern. The remaining three SEE mechanisms discussed in this paper are not destructive, but can make fault-tolerant computation challenging. These phenomena are discussed below.

2.2.1 Single-Event Latchup

Latchup is an issue that semiconductor manufacturers are already concerned about for terrestrial electronics reliability since it can destroy semiconductor devices through excessive current draw. Complementary metal oxide semiconductor (CMOS) technology is prone to latchup due to the parasitic transistors that result from integrating PMOS and NMOS transistors. SEL is a radiation-induced version of this destructive mechanism, where the charge implanted from the ionizing particle causes current to flow in the parasitic transistors. Designers generally avoid devices that are prone to SEL since many systems cannot tolerate the risk of having a damaged device while on orbit. For this reason, Altera's product lines have been avoided [7]. Xilinx products are free from latchup in the presence of protons and heavy ions.

2.2.2 Single-Event Upsets

The most common radiation-induced faults in SRAM-based FPGAs are SEUs (or *upsets*). SEUs affect memory devices by changing the stored values in memory bits, which could change the implemented circuit or the circuit's state in SRAM-based FPGAs. Specific failure states caused by SEUs are discussed in Section 3.

2.2.3 Single-Event Functional Interrupts

SEFIs are SEUs that cause more global functional effects and may require a device reset for device functionality to return. In SRAM-based FPGAs, SEFIs are often caused by SEUs in the control logic of the device, such as the internal control registers or the configuration interfaces (JTAG or SelectMap). Detecting and mitigating SEFIs is a challenge since the affected state can not be easily observed or fixed from the user standpoint. Specific SEFIs are discussed in Section 3.

2.2.4 Single-Event Transients

Single-event transients (or *transients*) are common in many semiconductor circuits. With this phenomena the ionizing particle causes a transient current state. If this transient state can propagate to a register during the setup and hold time (called the *window of vulnerability*), the transient will be latched (called a *latched SET*) as the intermediate data value and the circuit's state could be corrupted. For modern CMOS devices with fast clock speeds, latched SETs have become increasingly more common and distinguishing transients from legitimate signals is challenging. Unlike SEUs, latched SETs have a radiation-induced error rate that is dependent on the circuit's operating speed as faster clock speeds are more likely to latch SETs than slower clock speeds. For SRAM-based FPGAs, where the user flip-flops are outnumbered by several orders of magnitude by the configuration memory, the current understanding is that SETs are possible, but observability of SETs is hindered by the sheer number of SEUs in the configuration memory.

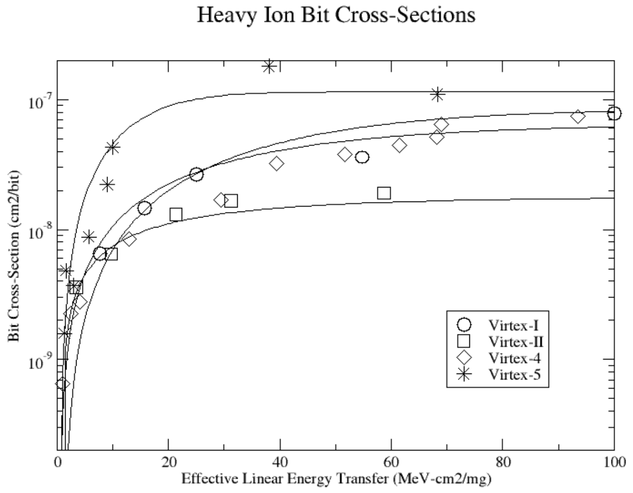


Fig. 1. Heavy Ion Bit Cross Sections for Virtex Family Devices [8].

Table 1. Bit Cross-Section for SEUs and Device Saturation Cross-section for SEFIs for Protons for Several Xilinx FPGAs [9]

Device	Energy (MeV)	σ_{bit} (cm^2/bit)	σ_{SEFI} (cm^2/device)
XCV1000	63.3	1.32×10^{-14}	$\approx 7.1 \times 10^{-13}$ (config SEFI)
XC2V1000	63.3	2.10×10^{-14}	9.46×10^{-13}
XC4VLX25	63.3	1.08×10^{-14}	6.43×10^{-12}
XC5VLX50	65.0	7.56×10^{-14}	Unknown

2.2.5 SEE Data

To determine on orbit error rates, the SEU and SEFI sensitivities have been measured experimentally for the Xilinx Virtex family devices using heavy ion and proton particle accelerators. SEUs are measured as either a per-bit sensitivity (*bit cross-section*) with units of cm^2/bit or a per-device sensitivity (*device cross-section*) with units of cm^2. SEFIs are measured as a per-device sensitivity (*device cross-section*) with units of cm^2. Cross-sections have two interesting characteristics: an *onset threshold* and a *saturation cross-section*. The onset threshold indicates the lowest energy or energy equivalent needed to cause an SEU or a SEFI, which can be less than 1 MeV-cm^2/mg for heavy ions. The saturation cross-section indicates the maximum sensitivity to the radiation source and often does not saturate in modern devices due to the presence of multiple-bit upsets [9].

Table 1 has a list of SEU bit cross-sections and SEFI device cross-sections for 63.3 or 65 MeV protons and Figure 1 shows the SEU bit cross-sections for heavy ions for Virtex family devices. Note that in proton the SEFI device cross-sections from Table 1 appear to be on the same scale as the SEU bit cross-sections, which is consistent with our understanding that the control logic is controlled by tens to hundreds of configuration bits. It should also be noted that the sensitivity to heavy ions is five to seven orders of magnitude larger than protons. While all of the SEU bit cross-sections are very small, each device has millions of bits. These cross-section values are used with orbit prediction tools, such as CREME96 [10], to determine the expected on-orbit error rate for a given device.

3. Failure modes from SEUs and SEFIs

FPGAs have many SEU-induced failure modes that conventional ASICs circuits do not have. For example, by changing one configuration bit, a LUT resource may no longer operate as a simple LUT, a wire might not connect the same two endpoints, or an input may suddenly be coming from somewhere else. For this paper, we will classify errors in this manner: failure modes that affect the circuit functionality, failure modes that affect the circuit's state, and failure modes that affect the device's functionality.

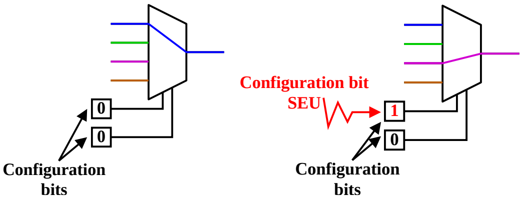
3.1 Failures that Affect Circuit Functionality

For SRAM-based FPGAs the circuit functionality is vulnerable to three types of changes: routing, LUTs, and tie offs. While maintain the LUT functionality is of the utmost priority for fault tolerant computing, the routing network and the tie offs are equally as important, if not more so, to maintaining circuit functionality. SEUs in the routing network can sever wires, making the transmission of the intermediate data values or the clock impossible. Furthermore, SEUs in tie offs can cause incorrect values to be injected into adders and multipliers. The vulnerabilities of these components is discussed in detail below.

3.1.1 Routing Vulnerabilities

In Virtex family FPGAs, the routing network largely consists of multiplexers, programmable interconnect points (PIPs), and buffers. In the older devices wires were connected using pass transistors (PIPs), whereas the newer devices use multiplexers. Finally, buffers are used where wires need to be actively and selectively driven by a few sources. These three resources are discussed below.

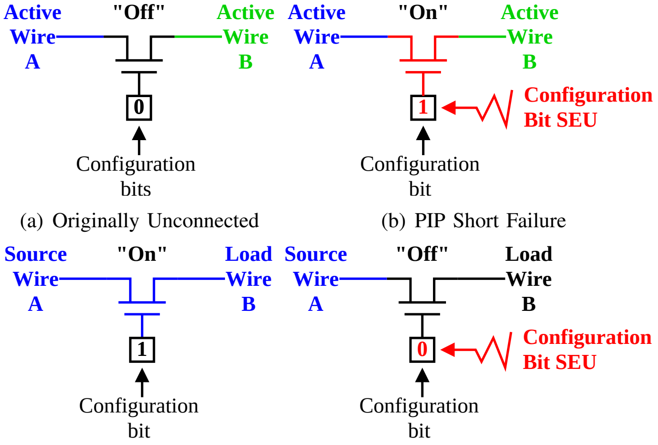
The select lines for routing multiplexers control which route is configured. These select line values are stored in configuration memory. An SEU in the select line configuration bits will cause a different routing configuration to be used. An example multiplexer select failure is shown in Figure 2. In practice, this could cause an input to float if the new input is not driven.



(a) Original (b) After Upset

Fig. 2. Multiple xor Select Failure Example

PIPs have two kinds of SEU-induced failures: shorts and opens. Figure 3(b) depicts a PIP short failure, where two wires with different functions in the design are shorted together. A PIP short can produce contention, causing output errors and increased power consumption. Figure 3(d) shows a PIP open failure. This failure effectively breaks a wire into two pieces and interrupts the flow of information from one part of the design to another.



(a) Originally Unconnected (b) PIP Short Failure

(c) Originally Connected (d) PIP Open Failure

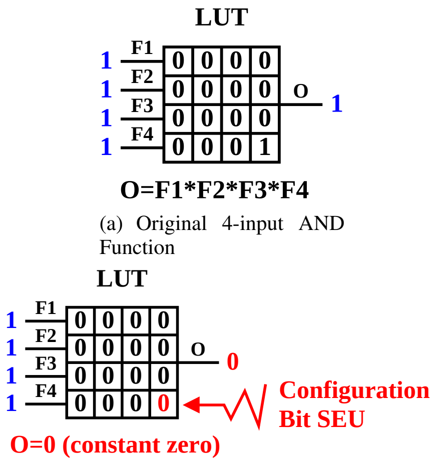
Fig. 3. PIP Failure Mode Examples

Buffers, like PIPs, either short or open when they fail. The main difference between buffers and PIPs is that a buffer failure is caused by an active driver and, therefore, is unidirectional, in a sense. With a PIP failure, it is quite possible that errors can be caused on both sides of the PIP, but with a buffer failure only the output is affected. As buffers usually are placed on the outputs of some multiplexers and on bi-directional wires, an SEU could cause a wire to be undriven.

3.1.2 Logic Vulnerabilities

There are two types of logic vulnerabilities: LUT value changes and control bit changes. The Virtex family FPGAs use lookup tables to generate most logic functions, so a change in the values stored in a LUT would impact the logic function implemented therein. This failure mode could cause constant or intermittent output errors depending on the inputs to the circuit and which part of the logic function is impacted. An example of a LUT value change is shown in Figure 4. Here the LUT implements a 4-input AND function. If the one bit that defines the "true" condition is upset, the result is a constant-zero function. For most inputs, the output of the function would still be correct, however, one case would cause problems.

In contrast to LUT value changes, control bit changes generally cause errors for all, or almost all, possible circuit inputs. The CLBs and IOBs use quite a few control bits to determine miscellaneous functionality. Figure 5 shows a partial schematic of a CLB. Bits V, E, F, and G are called programmable inversion bits. An upset to one of these will cause the value carried on that particular wire to be inverted, likely resulting in a circuit error when the value is used. The T bits, on the other hand, determine whether the LUT in this



(a) Original 4-input AND Function

(b) Upset LUT Function (Constant "0")

Fig. 4. LUT Upset Example

CLB performs as a LUT, a 16x1 dual-ported RAM, a part of a 32x1 RAM, or as a programmable shift register. If a LUT suddenly turns into a shift register, output errors will likely result. Other control bits determine such things as the electrical standard used in off-chip I/O and whether a storage element is a flip-flop or a latch.

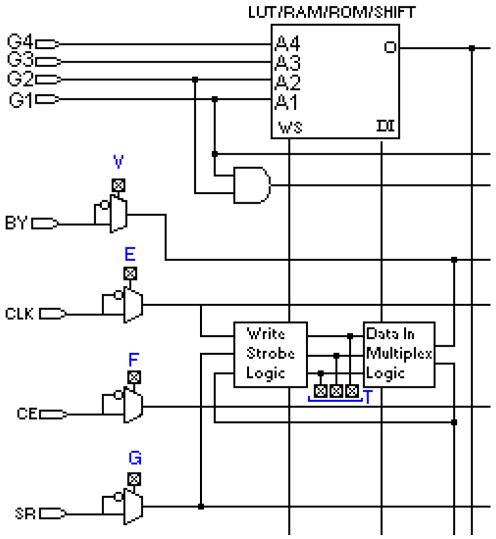


Fig. 5. Control Bit Examples [11]

3.1.3 *Tie Offs*

Tie offs are needed to generate constant zero and one logic values used internally by FPGA designs [12]. "Implicit" logical constants are widely used in designs to drive inputs to I/O, logic, RAM, clocking, and other resources. "Explicit" logical constants are needed for the zeroth bit of the carry chain for adders and unused multiplier/DSP inputs. Each device in the Virtex line has designed different approaches to tie offs. In all of the devices, the implicit logical constants were implemented with half-latches. In the Virtex-I, explicit logical constants were less noticeable, since these devices do not have embedded multipliers. When explicit logical constants are needed in the Virtex-I a combination of half-latches and constant LUTs are used, depending on where the value is needed. In the Virtex-II, the explicit logical constants are implemented with constant LUTs and, in the Virtex-4, each CLB has a VCC post. Unfortunately, half-latches and constant LUTs both have known failure modes.[1]

The vulnerability of half-latches is two fold: the weak keeper circuit is susceptible to SEUs and the half-latches are hard to observe. As half-latches are not directly initialized or controlled with programming data, their state is hard to observe and modify. For the Virtex-I, these two problems in concert made half-latches challenging, as the upset half-latches could hold the erroneous state for many seconds or longer and only a full reconfiguration would re-initialize half-latches. In Virtex-II and Virtex-4 devices, the half-latches generally return to their intended state within a few seconds, due to what is assumed to be a weak keeper with leakier transistors. While this means that transient-like behavior could be caused by the Virtex-II or Virtex-4 half-latches, at least they do not appear to require intervention to fix the circuit functionality.

For the Virtex-II, all of the explicit tie offs used in the carry chain of adders and unused inputs of the multipliers are supplied by constant LUTs, called the global logic 0/1 networks or the power network. In the Virtex-I some of the explicit tie offs are implemented in half-latches and others are implemented through constant LUTs. These constant LUT tie offs are susceptible to the usual LUT and routing SEU vulnerabilities. Unlike half-latches, which usually only provide localized, slice-level logical constants, constant LUTs are load balanced by the Xilinx tools so that several resources share the same constant LUT. Unfortunately, in designs that are masking SEUs using redundancy-based methods, sharing resources in this manner can cause single points of failure in the design.

3.2 Failures that Affect Circuit State

Maintaining a circuit's state can be difficult on orbit, as the state is vulnerable to SEUs that affect circuit functionality or SEUs in the user memory. In particular, when a circuit's functionality is affected by an SEU, incorrect intermediate data values could be generated. After the circuit's functionality is repaired through on-line reconfiguration, the bad state data generated during the error state will remain until it either naturally flows out of the system in feed forward circuits or the circuit is reset under more pathological conditions.

Vulnerabilities in the routing network are especially problematic to global signals, such as clocks and resets. Since errors are likely in circuits, space-ready designs frequently have global reset signals to force designs into a known state to ease initialization and recovery. Since FPGA architectures generally do not provide dedicated routing resources for resets, global resets utilize the general routing network, using a significant amount of resources and providing a large "target" for SEU-induced errors. Finally, one of the most common approaches to eliminating problems with logical constants used for tie offs involves driving the constants from input

[1] We are currently investigating SEU-related reliability issues with the Virtex-4 tie off posts.

pins, elevating logical constants to global signals. In all of these cases, the global nature of these signals means that SEUs that affect these signals can have significant impacts on circuit function.

Finally, SEUs can also directly affect user memory, such as user flip-flops or user SRAM, which could directly affect the circuit's state. While it is possible to discern upsets to any user-specified ROMs in the bitstream, the state of most user memory tends to be very dynamic, changing on a clock-cycle basis in some cases, and it is, therefore, hard to distinguish an error state from normal operation. Furthermore, it is not generally possible to read the contents of the memory while it is actively being used in a circuit without the possibility of affecting its content.

3.3 Failures that Affect the Device's Functionality

The device has a handful of configuration and control registers that result in SEFIs. Examples of SEFIs that have affected multiple Virtex family devices include JTAG TAP controller upsets, SelectMAP controller upsets, and configuration control logic upsets (Power-On-Reset or POR). The Virtex-4 is also susceptible to Frame Address Register (FAR), Global Signal, Readback and Scrub SEFIs. All of these SEFIs are discussed below.

A SEFI in the JTAG controller, whether it is being actively used or not, will move the device into an undesirable state. Compounding the problem, the Virtex family devices do not make the reset pin (TRST) available to the user, denying designers the ability to hold the JTAG controller in reset while it is deployed.

SEFIs within the programmable SelectMAP interface configuration pins will cause bad values to be returned on a read and corrupt the configuration when writing. Since the interface can no longer be accessed reliably, reading and writing to the device through this interface should be suspended until fixed. The Virtex-4 has two additional SEFIs (FAR and Readback) that mimic the SelectMAP SEFI. With the FAR SEFI the FAR will increment "continuously" and "uncontrollably" [13]. While the SelectMAP port will remain accessible during a FAR SEFI, the FAR will not be under user control. Virtex-4 designs that do not use the GLUTMASK configuration option are also susceptible to a Readback SEFI. This option ensures that the state of LUTs being used as RAM or shift registers are not affected by on-line reading or writing of the FPGA's configuration memory. Without GLUTMASK enabled, a false-positive SelectMap SEFI can be detected because on-line reconfiguration schemes might appear to not be working.

The configuration control state machine of Virtex family devices are vulnerable to the power-on reset (POR) SEFI. In this case, the device behaves as if $\overline{PROGRAM}$ has been asserted — its configuration is cleared and the $DONE$ pin is driven low.

The Global Signal SEFI is an umbrella SEFI that covers functional interrupts to the the Virtex-4's global signals, such as Global Set/Reset, Global Write Enable, and Global Drive High. These signals are observed through upsets in the configuration status (STAT) and control (CTL) registers.

The final SEFI is the Scrub SEFI observed in the Virtex-4. In this case, the control logic can be upset while scrubbing, causing a corruption of the data being scrubbed in. Not only does this affect the design, but causes a high current state [14].

4. Mitigation and Repair Methods

Without proper mitigation, there is no guarantee that output data will not be corrupted by SEUs. Accumulating SEUs increases the likelihood that output data corruption occurs. Accumulating upsets will also cause the device to draw more current, which could critically affect the device or the battery resources on the spacecraft. Therefore, mitigation and repair of SEUs is essential for reliable computation. To date, the best option found for mitigating SEUs [15] is to mask SEUs through the use of redundancy-based methods, such as triple-modular redundancy (TMR). On-line reconfiguration, called *scrubbing*, can be used to remove SEUs from devices. Previous research has shown that TMR with scrubbing is an effective method of masking the effects of SEUs [16]. As stated before, detecting SEFIs internally on the device is impossible and SEFI detection/mitigation is usually handled by an external scrubbing circuit. These two concepts will be described in better detail in the following two sections.

4.1 Triple Modular Redundancy (TMR)

The concept of a redundancy-based masking scheme was first introduced by von Neumann in 1956 [17]. While redundancy-based masking schemes can have any number of redundant copies of the circuit, the minimum number for masking is three copies. Voters compare the modules' outputs and output the majority value. Internal voters on SRAM-based FPGAs are susceptible to SEUs and are often triplicated, as shown in Figure 6, to remove the possibility that the data can be corrupted in the voters. For those applications that require maximum reliability for the FPGA hardware, we strongly recommend that TMR be applied to all aspects of the design, including I/O, global signals, logic and voters. Our recent work has shown that not triplicating global signals can make TMR-protected circuits less reliable than unprotected circuits [18].

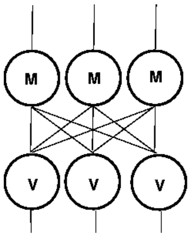


Fig. 6. The recommended implementation of TMR functional with triplicated modules (M) and voters (V).

The robustness of TMR masking schemes is dependent on failures existing in only one module at any give time. Previous research shows that fully TMR-protected FPGAs circuits have had the single-bit SEU cross-section completely removed from a design, leaving the design vulnerable to only SEFIs and non-single-bit SEUs. When faults exist in different redundant modules, masking cannot be guaranteed. Therefore, scrubbing is necessary to maintain the assumption that at most one modules is broken at a time. As long as the repaired module can resynchronize it's state before the next SEU occurs, there should only be one fault in the system at any given time. In larger, newer systems, the chance that either multiple-independent upsets (MIUs) or a multiple-bit upset (MBU) [9], where a single-ionized particle causes multiple upsets, occurs has become increasingly more likely. We have shown in previous work that designs that are able to mask all single-bit upsets might not be able to mask all MBUs [8]. Currently, there is no method to mask all MBUs. MIUs can be mitigated more effectively when the design is broken into several partitions and the smaller partitions are voted on. With this method, the TMR-protected design should be able to withstand more upsets, as long the upsets do not cause two or more modules in the same partition fail.

Because design tools perform many steps of translation and optimization on SRAM-based FPGA circuit designs, even the most careful descriptions of TMR-protected circuits in hardware description languages are often undermined by the synthesis tools removing all or some of the redundant logic. Furthermore, tying off all of the tristate buffers and unused inputs is difficult at the VHDL level. To circumvent these problems, we strongly recommend using one of the two tools that automatically apply TMR to post-synthesis circuit representations. One tool, called the BL-TMR tool, was developed by BYU and LANL and specializes in automatically applying partial TMR to designs when the fully triplicated design will not fit on the device [19]. The second tool is from Xilinx and is called the TMR tool [20]. Both of these tools will not only automatically TMR the design properly, but can handle the extraction of logical constants by driving half-latches and the power network from input pins.

Due to pin and area limitations, designers are sometimes unable to fully apply TMR to a design. In other cases, the reliability requirements for a mission may not warrant complete triplication, freeing FPGA resources for computation rather than reliability. In these situations, designers frequently decide not to triplicate global signals and/or I/O blocks, and SEUs affecting these resources will affect the hardware dependent on these resources. Under these circumstances, internally triplicating these signals can minimize the unprotected cross-section.

Alternatively, both the BL-TMR and TMR tools can apply partial TMR to a design. While the portions of the design not protected by TMR are user-specified in the TMR Tool, the BL-TMR tool was designed to automatically determine the most "critical" portions of the circuit that may induce persistent error states due to feedback. These are more critical because, unlike feed-forward circuits, these circuits may require external intervention to return to a functioning state. Of course, circuits that have only been partially protected with TMR are susceptible to some number of single-bit SEU-induced failures.

We have also found several scenarios where the implemented design is not as fully protected as expected. Furthermore, new error modes continue to be discovered. Therefore, assuming that a design is completely protected from SEUs is unwise until the design has been tested. Determining the unprotected cross-section from the design tools is an arduous, error-prone task and is not recommended. The current "gold standard" for pre-launch testing is radiation experiments at a particle accelerator. On one hand, given enough time and money, the experiments will be able to exercise all of the possible radiation-induced failure modes and find all of the problems with a user design. On the other hand, given the statistical nature of radiation-induced faults, it may be too expensive to get good test coverage and difficult to understand how the errors correlate to faults in the user design. In our work, we use fault injection and fault modeling tools until the design is mature enough to be taken to an accelerator. In this manner, accelerator testing is used for validation.

LANL and BYU, among a few others, have created fault injection tools for the Xilinx Virtex family of devices that can inject single- and multiple-bit errors into an FPGA's programming data so that output errors can be accurately attributed to individual bits or sets of bits in the configuration bitstream [21]. The advantage of well-designed fault injection tools is that the design can be uniformly upset across the device, effectively removing the statistical nature of accelerator testing. In addition, LANL has created a modeling tool—the Scalable Tool for the Analysis of Reliable Circuits (STARC)—that analyzes EDIF circuits representations for SEU vulnerabilities [18]. STARC is helpful for reliability analysis early in the design process (even before the target hardware is available) and in cases where fault injection is not feasible due to the design or available hardware and software. With these tools, unprotected cross-section can be quantified so that designers can decide whether the amount of unprotected cross-section is reasonable.

4.2 Scrubbing

Scrubbing uses on-line reconfiguration — a feature unique to Xilinx SRAM FPGAs — to reload the FPGA's configuration bitstream during circuit operation, removing any SEUs that may have accumulated in the bitstream between scrubs. While Xilinx provides some guidance [22], Xilinx should be engaged to guarantee scrubbing is done properly. In the past, "blind scrubbing", where the programming data is continually rewritten without reading the data back to ensure data integrity, was used frequently. Over the years, though, the control logic and registers necessary for scrubbing have grown larger and SEUs in these areas during scrubbing have been observed to cause the Scrub SEFI that causes a high current state on the device [14].

One recommended scrubbing algorithm is as follows:

1) Readback the configuration data.
2) Complete a CRC check for each configuration data frame.
3) If the CRC value does not match, scrub the frame.

Since the device is not scrubbed end-to-end, the Scrub SEFI should only affect one frame instead of multiple frames. Often times, when the device has suffered a SEFI, the number of frames that need to be scrubbed in a single scrub cycle will increase dramatically. Therefore, an effective method of detecting SEFIs is to keep track of how many frames are scrubbed in a scrub cycle. If this number exceeds some threshold, then a complete reconfiguration of the device is done and the circuit state is reset to fix the SEFI-affected logic.

For the Virtex-I and Virtex-II bitstreams, the portion scrubbed includes all of the configuration data for the global clock (GCLK), BRAM interconnect, IOB, and CLB configuration data. In these devices, LUT RAM, SRL16s and BRAM could not be scrubbed effectively, since either reading back their stored data would interrupt circuit operation or predicting the correct value to scrub into memories with dynamic values was too difficult. In the Virtex-4, there is a mechanism on the device that can be used to skip LUT RAM resources when the programming data is being read or written, making it possible to use the scrubbing approach mentioned above in the presence of user LUT RAM and SRL16s. In cases where the BRAM is being used as a ROM, scrubbing is necessary, and a BRAM scrubber is available from Xilinx [23].

5. Conclusion

In summary, this paper presented a number of possible radiation-induced faults in SRAM-based FPGAs. These faults can affect circuit functionality, circuit state, or device functionality. Further, we presented how TMR could be used to mask SEUs in the user design and how on-line reconfiguration could be used to remove SEUs from the device.

References

[1] 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 *Proceeding of the Military and Aerospace Programmable Logic Devices International Conference(MAPLD)*, Laurel, MD, September 1999.

[2] M. Caffrey, M. Echave, C. Fite, T. Nelson, A. Salazar, and S. Storms, "A space-based reconfigurable radio," in *Proceedings of the 5th Annual International Conference on Military and Aerospace Programmable Logic Devices (MAPLD)*, September 2002, p. A2.

[3] P. Graham, M. Caffrey, M. Wirthlin, E. Johnson, and N. Rollins, "Consequences and categories of SRAM FPGA configuration SEUs," in *Proceeding of the Military and Aerospace Programmable Logic Devices International Conference(MAPLD)*, Washington, DC, September 2003.

[4] P. Graham, H. Quinn, and J. Moore, *Xilinx Virtex FPGA Design Guide for Space*. on web at www.fpgamac.com, 2008.

[5] A. Holmes-Siedle and L. Adams, *Handbook of Radiation Effects*. Oxford University Press, 2002.

[6] H. Barnaby, *Evolving Issues for the Application of Microelectronics in Space*, ser. Short Course Notebook for the Nuclear and Radiation Effects Conference. IEEE, 2005, ch. Total Dose Effects in Modern Integrated Circuit Technology.

[7] S. L. Clark, K. Avery, and R. Parker, "TID and SEE testing results of the Altera Cyclone Field Programmable Gate Array," *IEEE Radiation Effects Data Workshop*, pp. 88 – 90, 2004.

[8] H. Quinn, K. Morgan, P. Graham, J. Krone, M. Caffrey, and K. Lundgreen, "Domain crossing errors: Limitations on single device triple-modular redundancy circuits in Xilinx FPGAs," *IEEE Transactions on Nuclear Science*, vol. 54, no. 6, pp. 2037 – 43, 2007.

[9] H. Quinn, P. Graham, J. Krone, M. Caffrey, and S. Rezgui, "Radiation-induced multi-bit upsets in SRAM-based FPGAs," *IEEE Transactions on Nuclear Science*, vol. 52, no. 6, pp. 2455 – 2461, December 2005.

[10] "Cosmic ray effects on micro-electronics (1996 revision)," on web https://creme96.nrl.navy.mil/.

[11] "http://www.xilinx.com/labs/projects/jbits/."

[12] P. Graham, M. Caffrey, M. Wirthlin, D. E. Johnson, and N. Rollins, "SEU mitigation for half-latches in Xilinx Virtex FPGAs," *IEEE Transactions on Nuclear Science*, vol. 50, no. 6, pp. 2139–2146, December 2003.

[13] G. Allen, G. Swift, and C. Carmichael, "Virtex-4VQ static SEU characterization summary," Xilinx Radiation Test Consortium, Tech. Rep. 1, 2008.

[14] C. W. Tseng, C. Carmichael, and G. Swift, "Optimizing configuration management for SEU mitigation in Xilinx Virtex-4 FPGA and self-scrubbing," http://nepp.nasa.gov/mafa/talks/MAFA07_20_Allen.pdf.

[15] K. Morgan, D. McMurtrey, B. Pratt, and M. Wirthlin, "A comparison of TMR with alternative fault-tolerant design techniques for FPGAs," *IEEE Transactions on Nuclear Science*, vol. 54, no. 6, pp. 2065 – 2072, 2007.

[16] N. Rollins, M. Wirthlin, M. Caffrey, and P. Graham, "Evaluating TMR techniques in the presence of single event upsets," in *Proceedings fo the 6th Annual International Conference on Military and Aerospace Programmable Logic Devices (MAPLD)*. Washington, D.C.: NASA Office of Logic Design, AIAA, September 2003, p. P63.

[17] J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," in *Automata Studies*, C. Shannon and J. McCarthy, Eds. Princeton University Press, 1956, pp. 43–98.

[18] H. Quinn, P. Graham, and B. Pratt, "An automated approach to estimating hardness assurance issues in triple-modular redundancy circuits in Xilinx FPGAs," accepted to the IEEE Nuclear and Space Radiation Effects Conference 2008.

[19] K. Morgan, M. Caffrey, P. Graham, E. Johnson, B. Pratt, and M. Wirthlin, "SEU-induced persistent error propagation in FPGAs," *IEEE Transactions on Nuclear Science*, vol. 52, no. 6, pp. 2438 – 45, 2005.

[20] "Xilinx TMRTool user guide," on web: http://www.xilinx.com/products/milaero/ug156.pdf.

[21] E. Johnson, M. Caffrey, P. Graham, N. Rollins, and M. Wirthlin, "Accelerator validation of an FPGA SEU simulator," *IEEE Transactions on Nuclear Science*, vol. 50, no. 6, pp. 2147–2157, December 2003.

[22] C. Carmichael, M. Caffrey, and A. Salazar, "Correcting single-event upsets through Virtex partial configuration: Application Note 216," on web: http://www.xilinx.com, 2000.

[23] G. Miller, C. Carmichael, and Jet Propulsion Labs, "Single-event upset mitigation for Xilinx FPGA block memories: Application Note 962," on web: http://www.xilinx.com, 2004.