Dynamic Reconfiguration for Management of Radiation-Induced Faults in FPGAs

Maya Gokhale and Paul Graham
Los Alamos National Laboratory

Eric Johnson, Nathan Rollins, and Michael Wirthlin
Brigham Young University

Abstract

This paper describes novel methods of exploiting the partial, dynamic reconfiguration capabilities of Xilinx Virtex V1000 FPGAs to manage single-event upset (SEU) faults due to radiation in space environments. The on-orbit fault detection scheme uses radiation-hardened reconfiguration controllers to continuously monitor the configuration bitstreams of 9 Virtex FPGAs and to correct errors by partial, dynamic reconfiguration of the FPGAs while they continue to execute. To study the SEU impact on our signal processing applications, we use a novel fault injection technique to corrupt configuration bits, thereby simulating SEU faults. By using dynamic reconfiguration, we can run the corrupted designs directly on the FPGA hardware, giving many orders of magnitude speed-up over purely software techniques. The fault injection method has been validated against proton beam testing, showing 97.6% agreement. Our work highlights the benefits of dynamic reconfiguration for space-based reconfigurable computing.

1. Introduction

Field Programmable Gate Arrays (FPGAs) offer significant advantage over microprocessors for space missions, which are characterized by demanding schedules, low budgets, and low volume. SRAM-based (as opposed to anti-fuse) FPGAs are especially appealing due to their in-situ reprogrammability and high performance for signal processing tasks. However, the use of commercial SRAM-based FPGAs in satellites and spacecraft presents unique challenges in the presence of the space radiation environment. Heavy ion testing [1] has shown that Xilinx Virtex XQVR300 SRAM-based FPGAs are single-event latchup (SEL) immune to up to a linear energy transfer (LET) of 125 MeV-cm^2/mg, but are sensitive to single-event upsets (SEUs) at an average threshold LET of 1.2 MeV-cm^2/mg with an average saturation cross-section of 8.0×10^{-8} cm^2. This means that in a Low Earth Orbit (LEO), the nine-FPGA system we have built (Figure 1) can be expected to experience SEUs 1.2 times/hour in low radiation zones and 9.6 times/hour when there are solar flares.

While it is highly desirable to exploit dense, dynamically reprogrammable commercial SRAM-based FPGAs rather than radiation-hardened anti-fuse parts with 1/10 the capacity, it is clear that fault detection and mitigation must be addressed before these FPGAs can be deployed on satellites or spacecraft for compute-intensive applications.

In this paper, we describe the crucial role of dynamic reconfiguration in detecting and correcting SEU-induced faults. Our system has been built and will be launched as an experimental payload in 2006 [2]. The system uses dynamic readback and reconfiguration techniques to detect and correct SEU faults in Xilinx Virtex V1000 FPGAs. In addition, we use a novel fault injection method based on dynamic reconfiguration to characterize SEU-induced configuration bitstream faults and to study the effects of these faults on the execution of our application set. This fault simulation technique has demonstrated a 97.6% correlation with the results found during a proton beam radiation study[3].

2. Fault detection and correction in a space-based reconfigurable computer

Our reconfigurable computer will serve as space-based reconfigurable radio (Figure 1). A radiation-hardened microprocessor, a 30-MHz RAD6000, controls the system and coordinates communication between the spacecraft and experimental payload. Other system components include non-volatile memory modules, the FPGA compute modules (labeled "RCC" in the diagram) and an A/D module receiving input from the radio. The signal processing is performed on three compute boards containing a total of nine Virtex V1000s and 288 MB of SDRAM organized in 8 M $\times$ 32-bit modules. The Virtex FPGAs in our system are radiation-tolerant XQVR Virtex FPGAs. While these FPGAs uses the mask sets of existing commercial devices, they are fabricated on epitaxial silicon wafers to provide SEL immunity.

Figure 2 shows one board of the reconfigurable computer. The FPDP channels provide high-speed paths to pass data among the FPGA boards. The Actel FPGA controller is a radiation-hardened anti-fuse FPGA that provides an in-

Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS'04)

0-7695-2132-0/04/$17.00 (C) 2004 IEEE

terface to the microprocessor and, more importantly, serves as fault manager. The Actel scans each Xilinx FPGA for SEU faults by continuously reading the FPGAs' configuration bitstreams and calculating a cyclic redundancy check (CRC) for each frame of each configuration. The frame is the smallest granularity of reconfiguration available on the Xilinx parts [4]. The calculated CRC is then compared with a codebook of stored CRCs. The stored CRCs are loaded from the FLASH module via the microprocessor and are kept in local SRAM.

Using the Virtex SelectMAP interface, each configuration is read every 180 ms while the FPGA is operating. If an error is found, the microprocessor is interrupted and notified of the specific device and frame that was corrupted. This information is stored and later relayed back to the ground station, contributing to the "State-of-Health" record of the subsystem. The microprocessor fetches the original frame bitstream segment from FLASH, partially reconfigures the device to restore that frame (156 bytes for the XQVR 1000), and then resets the system. Figure 3 shows the fault detection and mitigation process.

The system also allows for artificial insertion of SEUs into the Virtex parts using the microprocessor to partially configure with 'corrupt' frames. This stimulates the system to verify that the response to an SEU is correct at the logic and software level. Extensive SEU testing can be done this way.

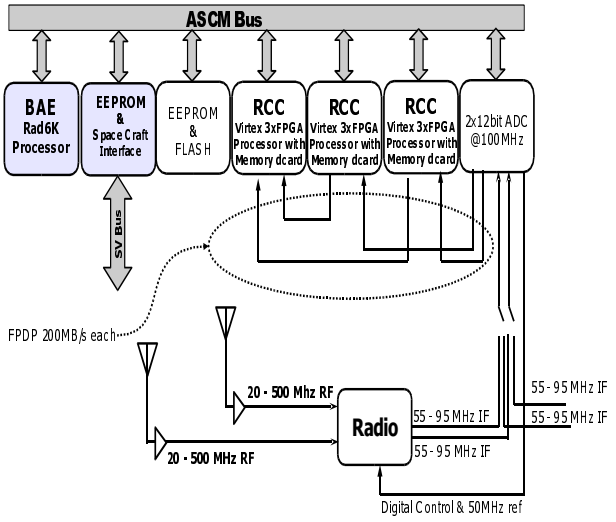

Figure 1. Space-Based Reconfigurable Radio

We note that there are limitations to this technique of configuration readback to detect and correct SEU faults. First, an error-free readback of the configuration bitstream

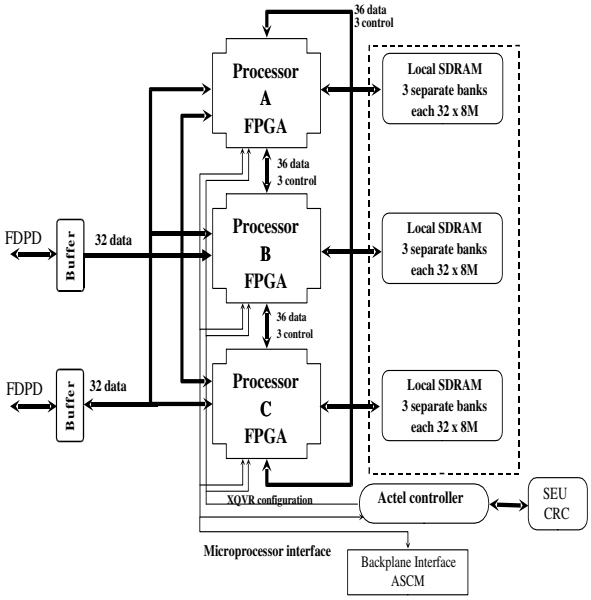

Figure 2. FPGA board with Configuration Manager

does not guarantee that an SEU did not occur. The FPGA contains hidden state that cannot be read back, and upsets to hidden state can conceivably cause errors in the design without any bitstream errors being detected[1]. Further, SEUs in flip-flop states can occur without disturbing the bitstream. Another limitation relates to the use of look-up tables (LUTs) as memory elements within Configurable Logic Blocks (CLBs). If a LUT is used as a memory element and that memory element is written while the readback operation is taking place, the bitstream will be corrupted on readback. Thus we must either disable the fault manager while running a design that uses LUTs as memory elements or stop the clock to do readback. For similar reasons, the on-chip Block SelectRAM (or BlockRAM) memory cannot be reliably read back without stopping the clock. Further, the output registers of the BlockRAMs become corrupted during readback. Thus, for BlockRAM arrays, error detection and correction must be handled via ECC or checksums since readback of BlockRAM cannot be reliably performed while the design is running.

Even with these limitations to dynamic reconfiguration

[1] In fact, this scenario occurred in our proton cyclotron experiments and, through detailed testing and analysis, was attributed to the use of half-latches in our designs. We have developed a tool that automatically removes half-latches and have subsequently found much higher correlation between errors in the configuration bitstream and errors in algorithm results.

0-7695-2132-0/04/$17.00 (C) 2004 IEEE

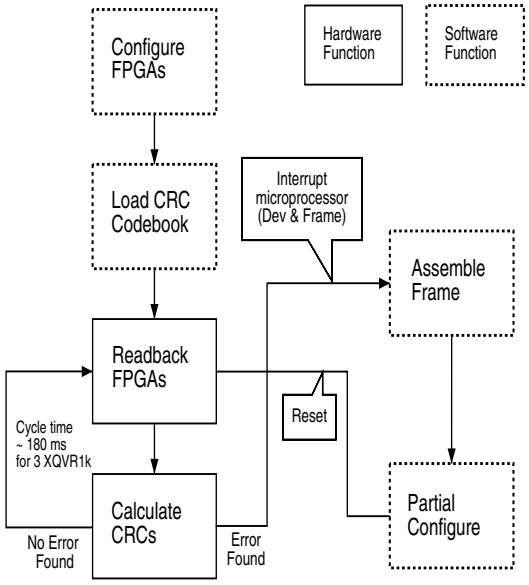

Figure 3. Flowchart of SEU-Induced Fault Detection and Correction

on the Virtex V1000 FPGAs, our reconfigurable radio represents a revolutionary advance in space-based processing. First, our choice of high performance SRAM-based FPGAs allows us to perform complex signal processing algorithms in orbit. In contrast, the current state of practice uses slow, expensive, radiation-hardened electronics that is not capable of deployed processing—data must be captured and then downlinked to the ground, with all processing and analysis occurring on the ground. Second, the use of reconfigurable FPGAs allows us to upload new designs, enabling deployment of new algorithms in the reconfigurable radio. The current state of practice uses anti-fuse FPGAs or other fixed logic so that the algorithm is fixed at launch time. Finally, our fault detection and correction scheme exploits the Virtex FPGA's readback and partial reconfiguration features, allowing us to correct the effects of bitstream SEUs on the algorithms while they are running.

3. Single event upset simulation

While it is important to detect and correct SEU-induced configuration bitstream faults on-orbit, it is equally important to understand the effect of faults on representative signal processing algorithms that will be fielded. A complementary aspect of our SEU-detection and mitigation project is to induce faults artificially into our designs and study the effects as the circuits function.

One method for inducing faults is to use ground-based radiation sources as in [1]. We (and others, for example [5, 6]) have put FPGAs under heavy ion (linear accelerator) and proton (cyclotron) radiation and observed SEUs during both static testing (i.e., the device is configured but the design is not executing) and dynamic testing (i.e., the design is executing). While testing has established that these radiation tolerant FPGAs are latchup-immune and SEU sensitive, extensive and repeated testing in the cyclotron is not feasible due to availability and cost.

To study SEU fault effects on our designs without using the cyclotron, we have developed a bench-testing methodology based on dynamic reconfiguration. We use an SEU simulator [7] that dynamically reconfigures the FPGA under test with corrupted configurations. Our testbed uses the SLAAC-1V PCI board from USC/ISI [8]. This board has three Xilinx XCV1000 FPGAs, ten 256x36 ZBT SRAMs, and a PCI bus interface. The three FPGAs are connected by a three-port crossbar and share a common clock and reset. In addition, the board includes a configuration controller—a Virtex XCV100 dedicated to configuration loading, partial reconfiguration, and readback.

With this platform, we can load identical designs into the X1 and X2 FPGAs, start the clock, and observe results (Figure 4). As the designs run, we selectively modify the bitstream of the device under test (DUT) and monitor the effects of the configuration corruption by comparing results produced from the two X1 and X2 FPGAs. X0 performs this comparison in real-time on a clock-by-clock basis. We have experimented with two design classes that are characteristic of our applications: feed-forward, data-path-dominated designs to assess the impact of SEUs on computation hardware and designs with local feedback (e.g., linear feedback shift registers) to assess the impact of error feedback. In the future we will focus on control structures, such as state machines, where configuration upsets can have global effects.

An important goal of our simulation is complete coverage of the configuration bitstream, which can only be accomplished through fast, partial reconfiguration of the device under test. We achieve rapid fault injection by using SLAAC-1V's high-speed PCI configuration mode along with the Virtex SelectMAP configuration interface. On our testbed, a single bit can be modified and loaded in 100 μs.

The simulator, running on the host with support hardware in X0, operates in a simple loop:

- The simulator corrupts the next bit in the configuration bitstream.
- The simulator partially reconfigures the DUT to load the corrupted frame.
- While the clock is running, the comparator circuitry in X0 checks for output discrepancies between the DUT and "golden" designs.
- The simulator logs discrepancies to a file.
- The simulator corrects the current bit.

This process, shown in Figure 5, takes 214 μs, making it

0-7695-2132-0/04/$17.00 (C) 2004 IEEE

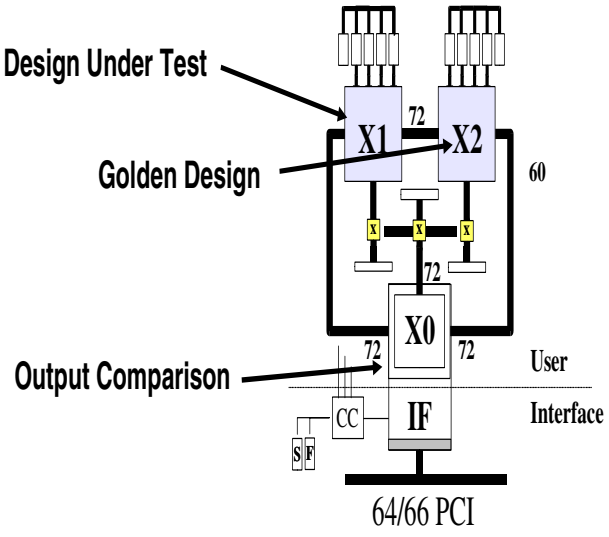

Figure 4. SEU Simulation on the SLAAC1V

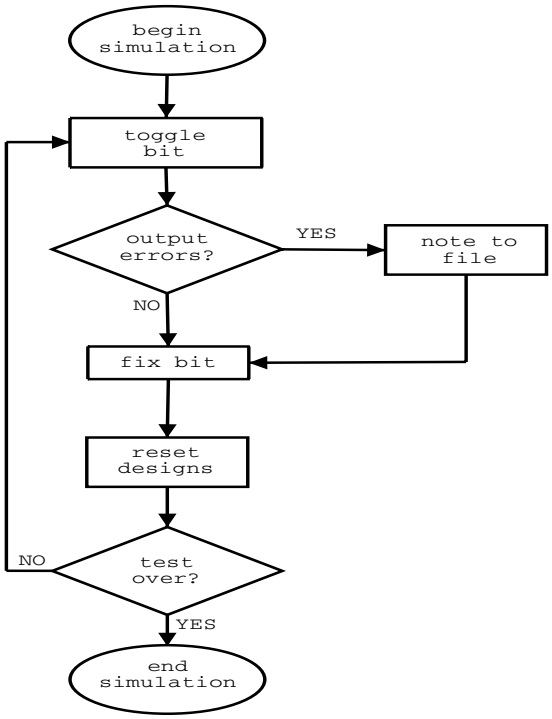

Figure 5. Flowchart of SEU Fault Injection

possible to exhaustively test the entire bitstream of 5.8 million bits in 20 minutes.

By repeated exhaustive tests, it is possible to correlate a single-bit upset in the bitstream with an output error. Such a correlation table was developed for our example designs, giving a probability of output failure for each bit in the bitstream.

To assess the accuracy of our SEU simulator, we have also benchmarked the simulator results against cyclotron testing. In this experiment, we used a SLAAC-1V board on a PCI bus extender with the DUT FPGA in a socket so irradiated FPGAs can be exchanged for new FPGAs when needed. Sheets of .75" aluminum shielded the FPGA board (except for the DUT) and the PC host. The objective of the radiation testing was to operate the designs at speed (up to 20 MHz) in the proton beam while appropriately adjusting the beam's flux so that about one bitstream upset occurred during each .5-second observation interval. Keeping the SEUs to about one per observation more closely mimics the on-orbit occurrence of SEUs since they are generally isolated events.

During the test, the output of the DUT was compared with the "golden" part output, and differences were logged along with a timestamp of the occurrence. At the same time, configuration readback was performed at regular intervals, and, when an upset was found, its position was recorded along with a timestamp.

Analysis of the log data showed a 97.6% correlation between output errors discovered through radiation testing and output errors predicted by the simulator [3]. This validates our bench testing methodology and greatly reduces the number of radiation experiments that must be conducted.

4. Readback and reconfiguration: architectural implications

Our choice of Xilinx FPGAs was driven by the desire to read back the configuration data and partially reconfigure the device. Readback, of course, enables bitstream SEU error detection while partial configuration allows us to repair just the portion of the bitstream that is corrupted as the design runs, avoiding the overhead of stopping the design and then completely reloading the programming data. With the utility of partial configuration and readback for improving design reliability in the space radiation environment come several shortcomings with the current generations of Xilinx SRAM FPGAs.

4.1. Readback

As pointed out in Section 2, it may be difficult to use LUTs as RAMs or shift registers on Xilinx Virtex FPGAs when using configuration readback. As stated in [9], a LUT being used as a RAM or shift register must not be written to as its contents are being read out by the FPGA's configuration circuitry since doing so can corrupt the contents of the LUT. Further, the CRC-based scheme described earlier for detecting errors in bitstream frames must be altered to mask out the contents of LUTs actively used as dynamic storage

0-7695-2132-0/04/$17.00 (C) 2004 IEEE

or a more bit-level comparison must be made, again, masking out regions of LUT state being used as RAM or shift-register storage.

Ideally, the FPGAs themselves could be designed so that this dual access of the LUT memory could be performed without corrupting the data. In a sense, the LUTs could be designed to be something like a true dual-ported memory or a memory that has a second "shadow" memory that can be read out without affecting design operation. Another alternative is to design the readback of LUTs so that their locations in the readback stream are set to zeros when the LUTs are being used in RAM mode. This would allow standard CRC checking to be done to the bitstream without having to mask out some locations and would not require dual-port access to the memories.

Since the volume of customers that actually try to perform readback during device operation is fairly small, an FPGA manufacturer such as Xilinx is not likely to make this sort of change since it would increase the area (and therefore the cost) of the their devices. Thus, we must compensate for the readback limitations at the application design and system levels.

At the design and system levels several approaches can and have been taken to deal with this issue. Many do not want to deal with the complexity of using LUTs as RAMs or shift registers during readback and so they are forced to completely avoid the use of LUT RAMs and they must use flip-flops for creating shift registers. For most of our designs, this is the standard approach since we *do* want to use readback for bitstream error location to help protect the integrity of our designs without having to create design-specific methods for handling readback.

Another approach is to not use readback at all to detect configuration bitstream errors but use built-in self-test techniques to periodically validate that the circuit is still functioning correctly. In this case, if an error is found, the test circuitry signals the configuration control circuitry that a configuration error exists and that a full reconfiguration is needed. This second approach was taken by Ray Andraka [10] when designing the 4096-point FFT used in our space application.

Yet another approach is to carefully design each application so that the columns with the LUT memories can be skipped during readback. This is not very convenient with the original Virtex architecture since using a LUT as a shift register or RAM in a single slice within a CLB column would require that 16 out of the 48 configuration data frames for that CLB column not be read back when trying to detect bitstream SEU errors. Further, if LUTs are being used as RAMs or shift registers in both Slice 0 and Slice 1 of the CLB column, 32 out of the 48 frames cannot be readback without causing the above mentioned problems. For Virtex-II, the situation is better since all of the LUT data for a given CLB column is contained in two configuration data frames, so most of the bitstream data for that column of CLBs can be read back during design execution without disturbing the circuit.

Lastly, it should be possible to schedule design operations and readbacks in such a way that they do not happen simultaneously for LUT resources. In the current approach used in our space payload, the bitstream SEU mitigation process executes asynchronously with regards to the operation of the FPGA designs. If this were changed to be synchronous with design operation and the LUT RAMs were not used continuously, it might be possible to schedule the readback of the LUT RAMs for when they are not being written. Alternatively, the design could be created so that writes to LUTs are inhibited during a readback process, or a portion thereof, to avoid the conflict. This approach might work if portions of the design could be stalled at appropriate times without seriously degrading the design's operation.

Block SelectRAM (BlockRAM) in Virtex has a similar access conflict during readback plus another problem. During readback, the configuration logic takes over the address lines of the BlockRAM so a design cannot read from or write to the RAM during readback [9]. An additional problem with BlockRAM and readback is that readback generally corrupts the internal output register of the BlockRAM so use of the BlockRAM after readback can be complicated. A work-around for the output register corruption is possible by modifying the user's design (see [11]), but again, it would be much easier for the FPGA user if the output register's initial value was restored once readback was complete by internal BlockRAM and/or configuration circuitry. Further, to overcome the RAM access problem during readback, the BlockRAM would effectively need a third memory port so that the operation could be performed without disrupting design operation.

4.2. Partial configuration

The main complication with using partial readback to repair designs is the granularity of access to the configuration data. With Virtex and Virtex II, the smallest portion of the bitstream which must be read or written is a frame—a collection of hundreds to thousands of bits. Forcing configuration data operations to manipulate at least a frame of data at a time was probably a compromise between providing the flexibility of partial configuration and the silicon costs of implementing partial configuration. In other words, Xilinx probably did not want to pay the costs for full, bit-level random access to the configuration bitstream data so they picked a larger addressable block size.

As a result, the main complication of using partial configuration to repair designs in the presence of dynamic storage resources (such as BlockRAMs or LUTs used as either RAMs or shift registers) is that a read-modify-write (RMW) operation really needs to be performed to fix bit-

0-7695-2132-0/04/$17.00 (C) 2004 IEEE

stream SEUs. If a configuration bitstream data frame is repaired with the original bitstream data when RAMs or LUT-based shift registers are contained in the design, the contents of these dynamic resources will be overwritten with their original initialization state, likely disturbing the operation of the design. For partial configuration to be usable with the current generations of FPGAs, the current state of the BlockRAM or LUT must be read and then, either that state must be inserted into the original bitstream frame or the readback bitstream frame must be repaired and written back into the FPGA. The big assumption for this to be successful is that the RMW operation can be done before the contents of the RAM or shift register change—a big issue for many designs.

One solution to this issue (as well as the readback issues mentioned above) is to provide a smaller granularity for read and write accesses to the configuration data. This way only the bits that need to be read or written are touched, making it easier to avoid accesses that might disturb circuit operation.

5. Conclusions

In this work, we show practical methods for exploiting readback and dynamic reconfiguration to enhance reliability of non-radiation-hardened FPGAs. Our methodology enables data and compute intensive signal processing algorithms to fly on satellites and in spacecraft. We have built a complete system that judiciously balances radiation-hardened and radiation-tolerant FPGAs to fulfill its goals—radiation-hardened FPGAs are used to monitor, control, and reconfigure the compute engine, which is built from radiation-tolerant FPGAs. Our simulation methodology frees us from extensive beam testing and gives complete, systematic, repeatable coverage of SEU effects by using an FPGA board in a workstation. Simulation results have been validated using proton radiation. We have suggested several architectural, system, and application methods to overcome the limitations of readback/reconfiguration in the current generation Xilinx Virtex FPGAs.

Acknowledgments: This work was sponsored by the Department of Energy through the Deployable Adaptive Processing Systems (DAPS) and Cibola Flight Experiment projects at Los Alamos National Laboratory.

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 *MAPLD Proceedings*, September 1999.

[2] M. Caffrey, "A space-based reconfigurable radio", in *Proceedings of the International Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA)*, T. P. Plaks and P. M. Athanas, Eds. June 2002, pp. 49–53, CSREA Press.

[3] M. J. Wirthlin, D. E. Johnson, N. H. Rollins, M. P. Caffrey, and P. S. Graham, "The reliability of FPGA circuit designs in the presence of radiation induced configuration upsets", in *Proceedings of the IEEE Symposium on FPGAs for Custom Computing Machines (FCCM '03)*, K. L. Pocek and J. M. Arnold, Eds. April 2003, IEEE Computer Society Press, To Be Published.

[4] C. Carmichael, M. Caffrey, and A. Salazar, "Correcting single-event upsets through Virtex partial configuration", Tech. Rep., Xilinx Corporation, June 1, 2000, XAPP216 (v1.0).

[5] R. Katz, K. LaBel, J. Wang, B. Cronquist, R. Koga, S. Penzin, and G. Swift, "Radiation effects on current field programmable technologies", *IEEE Transactions on Nuclear Science*, vol. 44, no. 6, pp. 1945–1956, December 1997.

[6] R. Katz, J. J. Wang, R. Koga, K. A. LaBel, J. McCollum, R. Brown, R. A. Reed, B. Cronquist, S. Crain, T. Scott, W. Paolini, and B. Sin, "Current radiation issues for programmable elements and devices", *IEEE Transactions on Nuclear Science*, vol. 45, no. 6, pp. 2600–2610, December 1998.

[7] E. Johnson, M. J. Wirthlin, and M. Caffrey, "Single-event upset simulation on an FPGA", in *Proceedings of the International Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA)*, T. P. Plaks and P. M. Athanas, Eds. June 2002, pp. 68–73, CSREA Press.

[8] USC-ISI East, *SLAAC-1V User VHDL Guide*, October 1, 2000, Release 0.3.1.

[9] Xilinx, "Virtex configuration - how do I perform Virtex readback?", Answers Database Record 8181, Xilinx, San Jose, CA, May 2001.

[10] R. Andraka and J. Brady, "Low complexity method for detecting configuration upsets in SRAM-based FPGAs", in *Proceedings of the 5th Annual International Conference on Military and Aerospace Programmable Logic Devices (MAPLD)*, September 2002, B4, To Be Published.

[11] P. S. Graham, *Logical Hardware Debuggers for FPGA-Based Systems*, PhD thesis, Brigham Young University, Provo, UT, December 2001.

0-7695-2132-0/04/$17.00 (C) 2004 IEEE