

LA-UR-

*Approved for public release;
distribution is unlimited.*

Title:

Author(s):

Submitted to:



Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the University of California for the U.S. Department of Energy under contract W-7405-ENG-36. By acceptance of this article, the publisher recognizes that the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

Reconfigurable Computing in Space: From Current Technology to Reconfigurable Systems-On-a-Chip

Paul Graham and Michael Caffrey
NIS-3, Space Data Systems
Los Alamos National Laboratory
Mailstop D440
Los Alamos, NM 87545, USA
{grahamp,mpc}@lanl.gov

Michael Wirthlin, D. Eric Johnson, and Nathan Rollins
Department of Electrical and Computer Engineering
Brigham Young University
459 Clyde Building
Provo, UT 84602, USA
{wirthlin,dej23,nhr2}@ee.byu.edu

Abstract—With the improved performance, in-system reprogrammability, flexibility, and reduced costs of using SRAM-based FPGAs in low-volume systems, FPGAs have some distinct advantages over both microprocessor and ASIC solutions in space-based computing systems. Despite these advantages, radiation-tolerant FPGAs are still susceptible to single-event upsets (SEUs) that must be managed to make these devices useful in space. This paper describes the various SEU categories for the Virtex FPGA and their observability characteristics as well as several techniques for managing the effects of these SEUs. Additionally, this paper provides several practical design considerations for systems employing Virtex FPGAs for space systems as well as a brief look at newer FPGA families and their potential for space-based reconfigurable computing. From this and previous work, we believe that some space-based systems can effectively use radiation-tolerant SRAM-based FPGAs.

TABLE OF CONTENTS

- 1 INTRODUCTION
- 2 SENSITIVE CROSS-SECTION
- 3 SYSTEM DESIGN
- 4 MITIGATION TECHNIQUE DEVELOPMENT
- 5 FUTURE ARCHITECTURES
- 6 CONCLUSION
- 7 ACKNOWLEDGMENTS

1. INTRODUCTION

The performance, in-system reprogrammability, flexibility, and reduced costs of SRAM-based field-programmable gate arrays (FPGAs) make them very interesting for high-speed, on-orbit data processing, but, because the current generation of radiation-tolerant SRAM-based FPGAs are derived directly from COTS versions of the chips, several issues must be dealt with for space, including SEU sensitivities, power consumption, thermal problems, and support logic. This paper will discuss Los Alamos National Laboratory's approach to using the Xilinx XQVR1000 FPGAs for an on-orbit pro-

cessing payload as well as the possibilities and challenges of using newer, system-on-a-reprogrammable-chip FPGAs, such as Virtex-II Pro, in space-based reconfigurable computing.

Reconfigurable Computing in Space

FPGA-based reconfigurable computing refers to the use of SRAM-based FPGAs for creating application specific hardware for computation. Since SRAM-based FPGAs can be reprogrammed over and over again, the same FPGA-based system can be reused for various tasks by simply changing the designs held in the FPGAs. Systems such as this can provide the computation speed of application-specific hardware while providing the flexibility and reprogrammability of software. For applications in a wide range of areas—including image and signal processing, cryptography, genetic database searches, and neural networks, to mention a few—reconfigurable computing has demonstrated the ability to outperform equivalent software implementations by factors of 10–1000 ([1][2][3][4][5][6]), depending on the application.

This processing advantage with respect to microprocessors can provide a space-based system with improved performance and can reduce the amount of communication bandwidth required with the ground since processed, rather than raw, data can be transmitted. Systems based on FPGAs can sustain processing rates of 100 million samples/second and greater, something which can be challenging for hardened microprocessor devices. In addition, the poor availability of hardened microprocessors devices makes processing with alternative technologies attractive.

Similar performance advantages can be achieved with application specific integrated circuits (ASICs), however, FPGAs have several important advantages over ASICs for space applications. First, many space systems are produced at low volumes, meaning that ASICs for these systems may be quite costly due to non-recurring engineering (NRE) costs and the economics of using expensive, low-volume, radiation-hardened fabrication facilities. FPGAs, on the other hand, provide a higher volume solution which can be customized to the application after manufacture, allowing the NRE and other costs to be amortized more effectively across multiple applications and customers. Radiation-tolerant FPGAs can,

therefore, be more cost effective for low-volume systems than radiation-tolerant or hardened ASICs—a situation similar to that of traditional SRAM-based FPGAs and ASICs.

Second, FPGA-based systems usually add relatively little time to the length of the design process when compared with ASIC-based systems. The time required to design, fabricate, and test ASICs can add a year or more to the system design process while the integration of COTS-based FPGAs ideally involves the integration of pre-tested, immediately available devices from the FPGA manufacturer. Additionally, design changes are much less costly and time consuming when designs are implemented with FPGAs rather than as ASICs.

Third, SRAM-based FPGAs can be reprogrammed within a system numerous times while ASICs have a fixed function. This reprogrammability has several impacts for space:

- *Reusability:* The same FPGA processing hardware can be reused for multiple tasks or applications while in flight.
- *Mission obsolescence and design flaws:* Design fixes and improvements or even mission changes can be made after launch through reprogramming.
- *Hardware survivability:* On-orbit faults in the FPGAs, if they occur, can be assessed and then mitigated through modifying designs and reprogramming the system with the modified designs.

Radiation-Tolerant SRAM FPGAs in the Space Environment

Currently, Xilinx is the main (and possibly only) manufacturer providing radiation-tolerant SRAM-based FPGAs. As described in [7], the approach taken to develop these radiation-tolerant FPGAs has been to take the mask sets for existing commercial FPGA devices and fabricate the designs on epitaxial silicon wafers to provide single-event latchup (SEL) performance useful for many space applications. Total ionizing dose performance for these parts, which is a serendipitously provided by modern CMOS processes, is guaranteed through radiation lot acceptance tests (RLATs). Since the designs for these devices are the same as those used in the commercial devices, the radiation tolerant devices have no specific mechanisms for mitigating single-event upsets in device state, hence, they are still susceptible to single-event upsets (SEUs).

In addition to the challenges of managing SEUs with these FPGAs, these high-density SRAM-based FPGA devices can consume substantial amounts of power. Thus, using these devices requires careful thought for the thermal design aspects of systems and, possibly, power-conscious FPGA design techniques.

A Space-Based Reconfigurable Radio

Los Alamos National Laboratory has an on-going project to understand the challenges of using a reconfigurable computing platform in space[8]. For this project, Los Alamos is building a reconfigurable processor payload intended for a

low-earth orbit (LEO) system which will survey the radio spectrum from 20–500 MHz. Networks of reprogrammable FPGAs are used to process the intermediate frequency (IF) for ionospheric and lightning studies. The object is to detect and measure impulsive events that occur in a complex background. Using the reconfigurable capability, we can change processing to search for different signatures or enhance the sensitivity while the system is on orbit by improving the detection algorithms. The processing demands for these measurements are immense and cannot be accomplished with multiprocessing for our system.

The payload consists of a chassis containing power converters, tuners, and digital processing hardware including: analog to digital converters (ADC), reconfigurable computers (RCC) using radiation-hardened Xilinx Virtex FPGAs, memory, spacecraft communications, and a microprocessor (R6000). The payload also has 2 antennas. Two 40-MHz RF channels are gang tuned to an intermediate frequency (IF) on the range of 55–95 MHz. The two IF channels are then sampled at 100 MHz with 12-bit resolution and the digital IF is transmitted to a network of 9 Virtex XQVR1000 reconfigurable FPGAs for processing. Figure 1 provides a block diagram of the system.

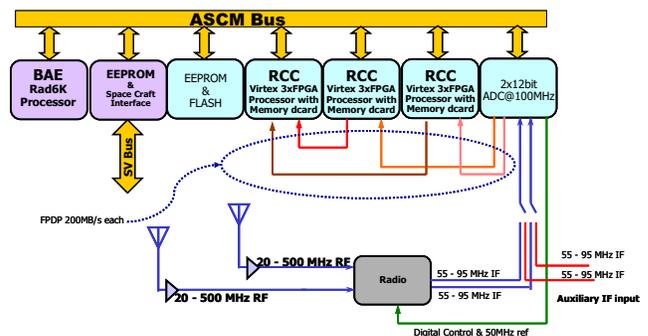


Figure 1. Functional Diagram of the Reconfigurable Radio

Unlike other approaches to mitigating SEUs in SRAM-based FPGAs ([9][10]), the techniques at Los Alamos are *not* attempting to make FPGA circuits 100% tolerant of SEUs, but instead are interested in developing FPGA designs which can continue to operate without a system reset for extended periods of time in the presence of SEUs. The approach allows SEUs to affect feed-forward circuits, where the results of the upsets are temporary and appear as noise in the system, while applying redundancy to portions of circuits with feedback, where SEU-induced errors may persist for extended periods of time. This more resource- and power-efficient approach to SEU mitigation should insure that more resources are available for providing additional computation power rather than using these additional resources for implementing mitigation through redundancy.

The remainder of this paper will describe the nature of the various SEU issues as well as the various approaches which

Los Alamos has taken to handle the SEU mitigation problem in the context of this reconfigurable radio system. Also, the paper will discuss how these approaches should be applicable to newer architectures, such as newer FPGA architectures such as Virtex-II and Virtex-II Pro.

2. SENSITIVE CROSS-SECTION

The set of all upsets includes categories with different consequences and different cross-sections. Figure 2 indicates the various categories we suspect exist for the Virtex FPGA. Most categories (flip-flops, JTAG TAP controller, Block SelectRAM, transients) are common with many digital devices. The Virtex has unique sensitivities such as the configuration bitstream, half-latches, and the configuration management controller (which is referred to as POR—Power On Reset—in Figure 2 due to its reset register sensitivity).

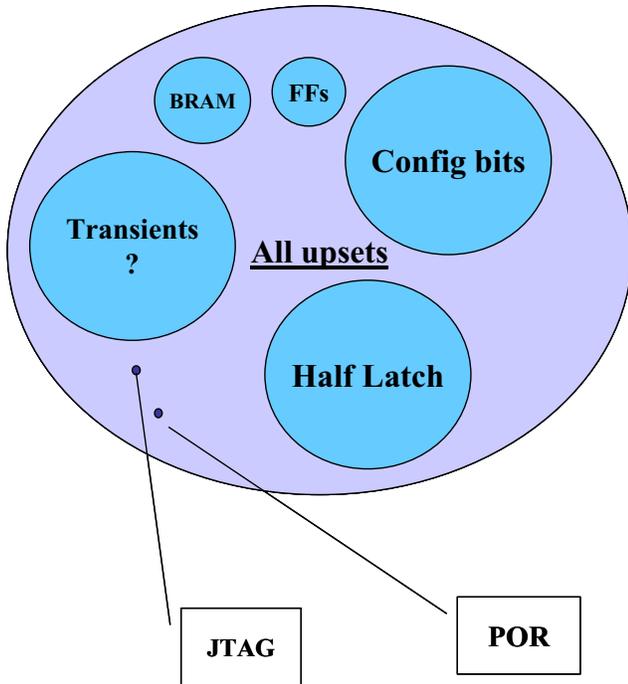


Figure 2. The Set of All Upsets Categorized for Observability/Mitigation Considerations

Heavy-ion testing [7][11] has shown that the average saturation cross-section per bit in Table 1 is $8x10^{-8}cm^2$ and that the cross-section measured for the Single Event Functional Interrupts (SEFIs) is $1x10^{-5}cm^2$ total per device. Those results should scale from the 300,000-gate device tested previously to the 1-million-gate device discussed here. The relative contribution from half-latches is difficult to measure because they are not directly observable. However, a mitigation scheme to eliminate the half-latch contribution is discussed later in the paper.

The relative percentages called out in Table 1 are in reference to the total static-test cross-section that does not include the

Table 1. Relative Size of Contributors to the Virtex XQVR1000 Sensitive Cross-section

Resource	Contribution (in bits)	% of Total Cross-section
User Flip-Flops	26,112	0.4%
LUT Bits	393,216	6.4%
Block SelectRAM	131,072	2.1%
Configuration	5,603,456	91.0%
Single-Event Functional Interrupts	?	<.0021%
Transients	?	?
Half-latches	?	?

possible contribution from half-latches and transients, which only manifest themselves in dynamic testing. The lookup table (LUT) and configuration bits are both observable in the configuration readback bitstream, which the device can provide through the SelectMAP configuration interface while the design is in operation. This makes the vast majority, more than 97%, of static upsets directly observable while the system is in service. For our purposes, LUT bits are implicitly included when discussing configuration bits unless specifically noted.

Transient Effects

The existence of transient induced upsets has not been established. Proton accelerator experiments [11] suggested that approximately half of all upsets detected during dynamic testing were not due to configuration bitstream upsets. Those experiments failed to show any detectable clock rate dependence that would suggest the presence of transient effects. There may be an unknown contributing cross-section or the test may have been faulty. The test had drawbacks, for instance, half-latches had not been removed from the designs; in fact, the test helped prompt their discovery. In addition, the design was not a “golden chip” test but, due to the complexity of the test fixture, the design under test was self-checking. An upcoming experiment will revisit the issue having resolved both concerns regarding the previous test.

Configuration Bitstream

Upsets in the configuration bitstream may result in erroneous processing. Many Virtex resources are left unused even, in very dense designs, so not every upset will result in incorrect processing. The Virtex SelectMAP interface allows the device’s configuration data to be read back while the device is in use—a feature we exploit to detect upsets. In addition, the Virtex can be partially configured, which speeds the recovery time.

The architecture of the configuration bitstream is shown in Figure 3. The Block SelectRAM (or BlockRAM) bits are on the outside edges of the device and are not included in a readback of the central, CLB section of the device, i.e., the CLB

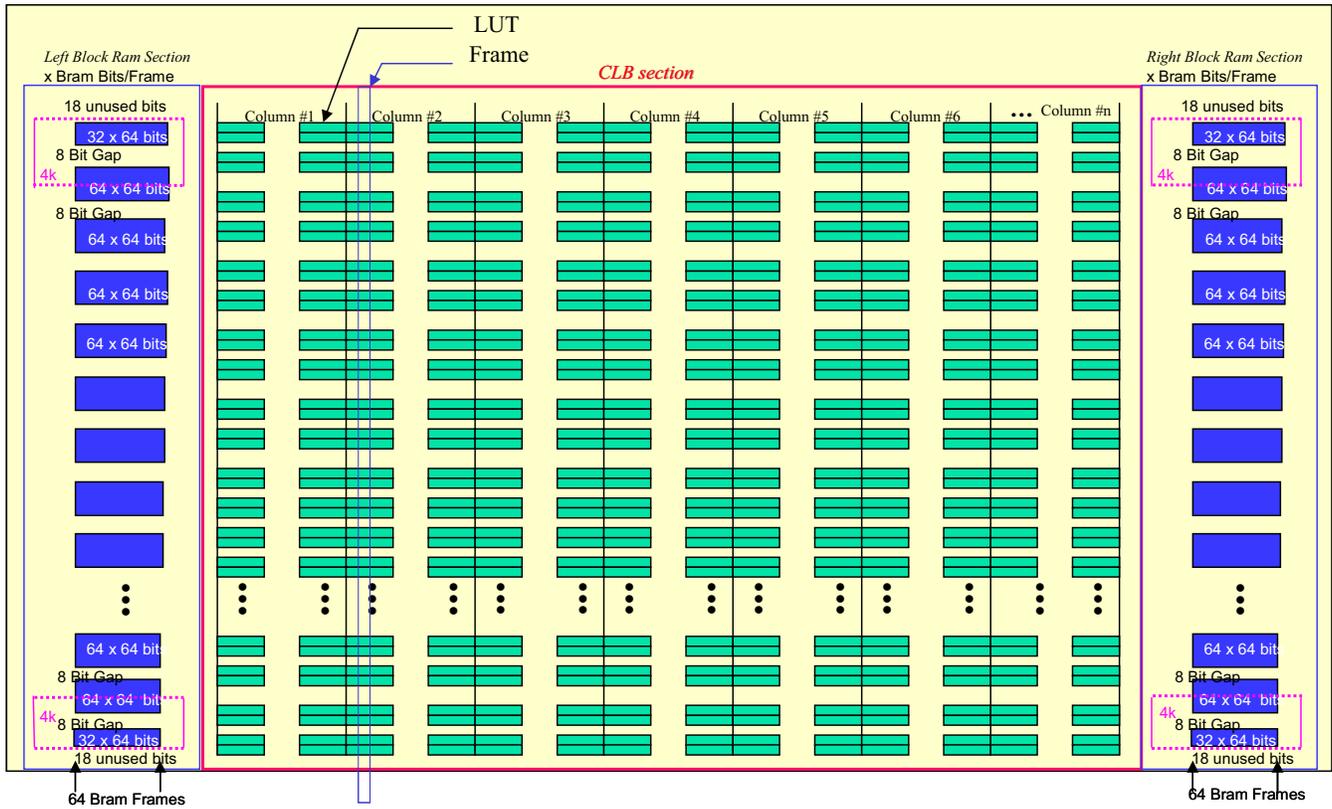


Figure 3. Virtex Configuration Bitstream Architecture

section includes all of the non-BlockRAM-related configuration bits. The XQVR1000 has a CLB section divided into 4778 individually accessible configuration frames, which run vertically. A frame is the smallest amount of data that can be read back or partially configured into the device.

Considering this, our bitstream upset detection and correction scheme does not track the exact location of a bitstream upset, just which frame contains incorrect data, i.e., each upset is treated with equal significance. Our bitstream upset management scheme [11][12][13] uses a controller to continually readback the configuration of the Virtex device and calculate a Cyclic-Redundancy Check (CRC) word for each frame in the bitstream. Each frame’s CRC word is compared to a CRC word that has been precomputed on the ground. An error indicates the presence of an upset in the frame. When an error is detected, the frame is reconfigured with the correct data. The continuous readback and CRC calculation take place in hardware in the configuration controller, relieving the system microprocessor from the task. Since readback is performed continually after initial configuration, the SelectMAP interface pins must be reserved for readback during normal operation and cannot be used as user IO.

Block SelectRAM

Because the BlockRAM cannot be read safely via the SelectMAP interface while it is in use, upsets must be tolerated

or mitigated using error control coding (ECC) for detection and correction if desired; for example, checksums, parity, or CRCs can be used for upset detection. Using BlockRAM is not materially different from using external SRAM sensitive to upset—any mitigation must be implemented in logic with penalties in density and power.

Flip-Flops

Reducing the consequences of upsets in user flip-flops (both CLB and IOB) requires the use of redundancy. While the state could be observed using the “capture” mode of readback, there is frequently no way to predict *a priori* a design’s complete, correct state if it is performing a complex function. The impact of an upset in a flip-flop in our system varies with the application. In finite-impulse response (FIR) portions of the design, the errors will eventually be flushed out. In infinite-impulse response (IIR) portions, such as many finite state machine controllers and some signal processing hardware, the design may never recover on its own. This suggests that if we are to spend logic and power on reliability, we gain more by focusing on IIR structures. Our project intends to manage these on an algorithm design basis, so each algorithm design can have a unique reliability versus throughput and power trade-off.

Half-Latch Structures

Another problem encountered when using Xilinx Virtex FPGAs in a radiation environment is that certain circuits, called half-latches, which generate many of the constant “0” and “1” values used by designs on Virtex FPGAs, are also susceptible to SEUs. When upset, the output values of these circuits will remain inverted until the device is fully reprogrammed. Further, this inversion is not directly observable through reading back the configuration bitstream.

An example of the half-latch issue is shown in Figure 4. In this case, the unused clock enable input is driven by a half-latch. If the half-latch is upset, it will disable the flip-flop, modifying the intended function of the circuit. This modification cannot be observed through reading back and checking the configuration bitstream.

At an architectural level within Virtex FPGAs, half-latches drive IOB, slice, Block SelectRAM and other resource inputs when there are no direct sources for the input, i.e., when the inputs are left unconnected. Half-latches are very efficient and ubiquitous sources of “0” and “1” values throughout the device—no LUTs or other resources are required to generate constant logic values. Consequently, the Xilinx implementation tools generously use them throughout most designs. The resources or inputs shown in Table 2 can be driven by half-latches in the Virtex architecture.

Table 2. List of Known Half-latches for the Virtex Family

Resource	Inputs
BLOCKRAM	WEAMUX, ENAMUX, RSTAMUX, WEBMUX, ENBMUX, RSTBMUX
BSCAN	TD01MUX, TDO2MUX
CAPTURE	CAPMUX
DLL	RSTMUX
GCLK	CEMUX
IOB/PCIOB	SRMUX, TRIMUX, TCEMUX, OMUX, OCEMUX, ICEMUX
PCILOGIC	I1MUX, I2MUX
SLICE	BYMUX, BXMUX, CEMUX, SRMUX, F1-F4, G1-G4
STARTUP	GWEMUX, GTSMUX, GSRMUX
TBUF	TMUX, IMUX

Within the FPGA Editor tool provided by Xilinx, the use of a half-latch is expressed as a constant “0” or “1” input into the above mentioned muxes. In reality, these muxes only have the ability to select their inputs or inverted versions of their inputs and the constants illustrated in FPGA Editor are values produced by half-latches.

In general, the half-latches driving the inputs to the above mentioned muxes are the critical half-latches. Modification to the values at the inputs of these muxes can have serious consequences to the operation of circuits. On the other hand, the unused inputs to the LUTs (F1-F4, G1-G4) are not as crit-

ical since the logic functions are encoded redundantly within the LUT such that “0” or “1” values on the “don’t care” inputs result in the same output value.

The solution to the hidden half-latch inversion problem is to remove the reliance on half-latches that produce the constant “0” and “1” values in designs by routing an explicit “0” or “1” value to these mux inputs. The sources of these explicit “0” and “1” values should be created by using resources which can be explicitly configured with the bitstream so that SEU errors which cause changes to these constant values can be handled via bitstream error detection and correction methods. Figure 5 illustrates some potential sources for constant values: (a) FPGA input pins driven externally by a logic “0” or “1”, (b) LUTs filled with an all “0” or all “1” values (as appropriate), or (c) flip-flops. Note that the flip-flop circuits self-correct if the flip-flop state is upset via radiation. Other constant sources are possible as well. Again, the important feature is that if these sources of constant logic values are disturbed by SEUs, the error in their configuration can be detected using bitstream readback techniques. In addition to remaining vulnerable to SEU, these sources of constant logic values generally require the use of additional FPGA resources.

The half-latch removal process can be performed at several different stages in the design flow, from the HDL or schematic-entry level down to the bitstream level, as Figure 6 shows. As a general comment, the level of design abstraction decreases as a design moves from left to right in the figure.

Approaches that modify designs earlier in the process are less likely to eliminate all half-latches since synthesis, technology mapping, and, possibly, placement and routing may introduce half-latches into the design implementation. Careful design may insure that half-latches are never used in the design in the first place, but this will require fairly involved design practices, such as requiring that explicitly generated constant “0” and “1” values be connected throughout the design and that HDL descriptions be carefully structured so that half-latches are not introduced. As an additional complication to these approaches, design practices which worked for older synthesis and technology-mapping tools may not work for new tools as synthesis and technology mapping techniques evolve.

Of course, if the use of constants and, thus, half-latches were controllable in the synthesis and technology mapping stages via a switch or parameter, most, if not, all of the half-latch problems would not exist. Unfortunately, this problem is of relatively little interest to the commercial companies which provide the designs tools and is unlikely to be fixed by these tool companies.

The *Physical Database* and the *Bitstream* representations of the design are probably the most promising design representations to modify since they can or do represent the final placed-and-routed forms of the designs. So, if the half-

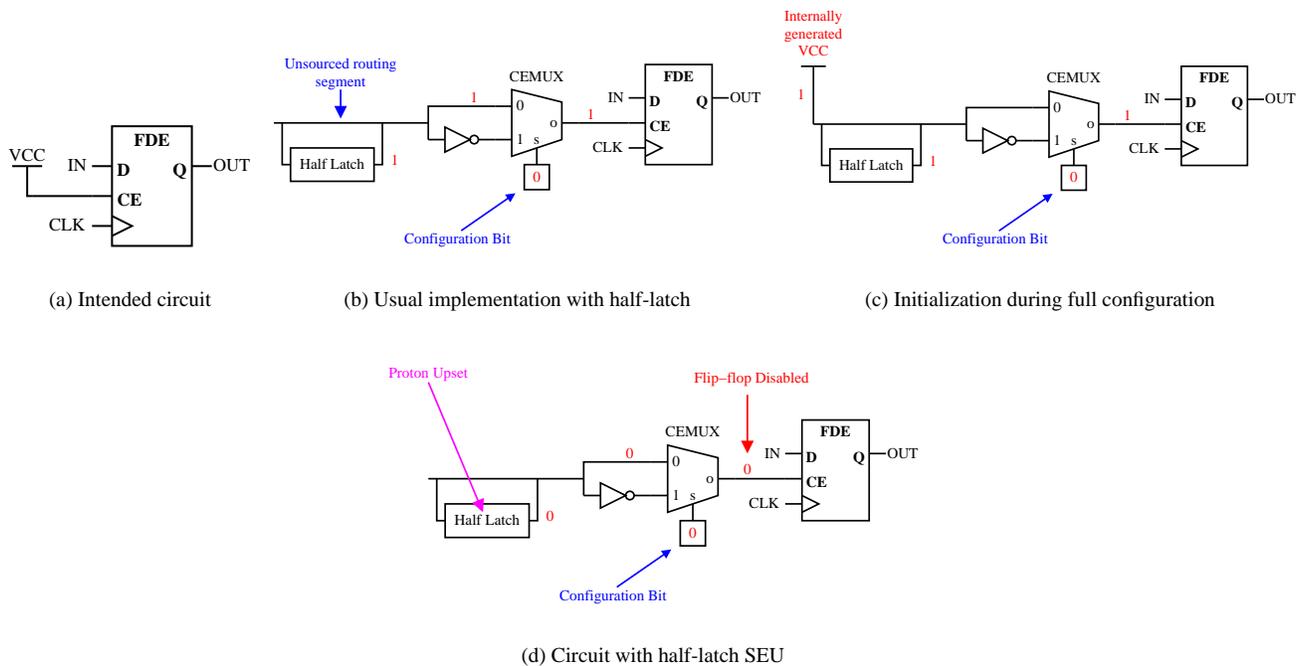


Figure 4. Half-latch Illustration Showing the Intended Circuit, Half-latch Initialization, and the Result of a Half-latch Upset

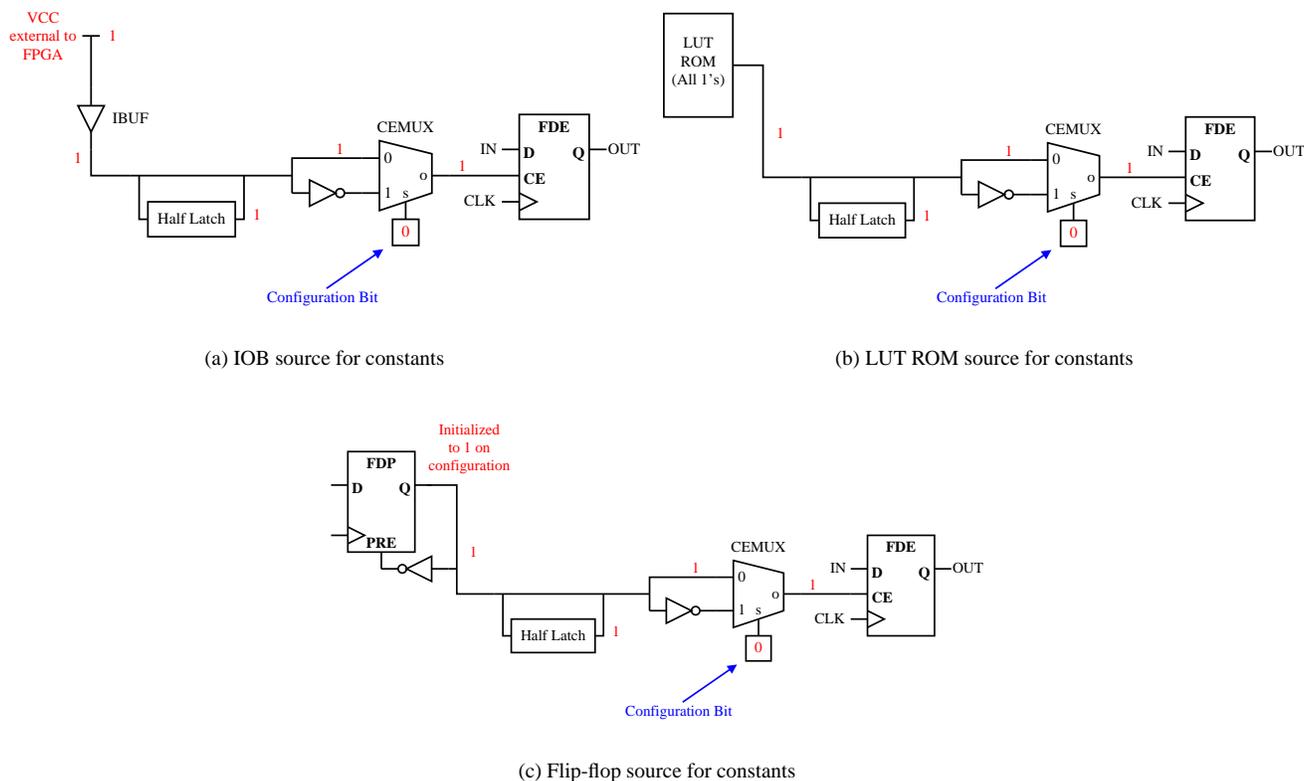


Figure 5. Observable Source Alternatives to Half-latches

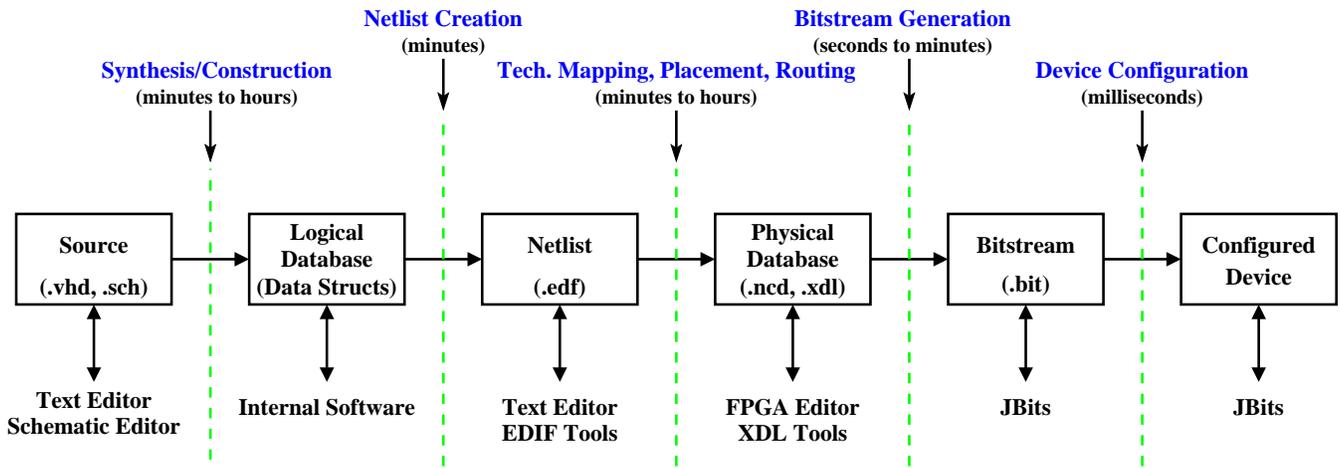


Figure 6. Xilinx FPGA Design Flow and Design Forms

latches are removed at these points in the design flow, half-latch problems should not exist. With these latter design representations, the designer is dealing with the design expressed in terms of FPGA resources at a very low level of abstraction.

Modifying designs while in these low-level forms does have a few disadvantages. First, only a handful of tools are useful for manipulating designs at this level. As a second disadvantage, because of the low abstraction level, a detailed knowledge of the devices is required to make the modifications or design tools to automate modifications. Finally, because designs are placed and routed when they are being modified, there is a slight possibility that routing or logic resources may be too limited to remove the half-latches.

As referenced in Figure 6, *FPGA Editor* from Xilinx can be used to modify the design in the form of a Native Generic Database File (.ncd). Further, Xilinx provides a program called *xdl* that converts the proprietary Native Generic Database File format into a published, open format called the Xilinx Design Language (XDL). Once converted to XDL, third-party tools can be written to manipulate the XDL and the modified design can be converted back to the Native Generic Database format for bitstream creation. *JBits*, another Xilinx design tool, is also a possibility for modifying designs at the bitstream level, but currently less than 100% of Virtex and Virtex-E devices are configurable via *JBits* (as of version 2.8).

Currently, we have created a tool, *RadDRC*, which parses the XDL representation of a design to locate half-latch issues and then generates a script for *FPGA Editor* to automate the removal of all half-latches. The initial version of the tool uses the approach illustrated in Figure 5(a) where a single externally driven FPGA pin is used to generate a logic “1” and that source is routed to all muxes initially having half-latch sources. The muxes are configured as inverting (to produce a “0”) or non-inverting (to produce a “1”) depending on the

constant value required. Figure 7 illustrates the tool flow which incorporates *RadDRC* into the design flow to remove half-latches from designs.

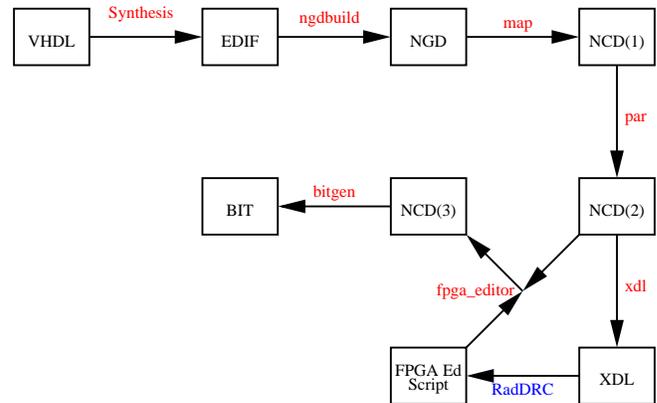


Figure 7. Xilinx Design Flow when Using RadDRC 0.1

A new version of the tool which modifies the XDL representation of the physical database directly is currently being tested as well. In addition to the externally driven I/O approach, this newer version also supports the approach of Figure 5(b), which uses the LUT ROMs to remove half-latches. In the coming months, we will be testing a few different half-latch removal techniques under proton radiation to understand which techniques are best.

Single-Event Functional Interrupts

Single event functional interrupts (SEFIs) are those SEUs that have unusually far reaching consequences. The SEFIs that we believe exist in the Virtex architecture include:

1. JTAG TAP controller
2. Configuration control state machine reset (FSM POR), and
3. SelectMAP configuration pin upsets.

The total cross-section for these three signatures is measured less than $1 \times 10^{-5} \text{ cm}^2$, as mentioned previously, and all can be detected by the unusually large number of “upsets” detected during a device readback. Each of these SEFIs interferes with device readback or clears the device configuration and results in reading many more errors than anticipated for the radiation exposure.

The JTAG TAP controller in the Virtex device does not make the reset pin available to the user to hold the controller in reset while deployed. It has been reported [14] that JTAG TAP controllers can upset, moving the device into an undesirable state. The approach to managing this when the reset signal is unavailable is the same for the Virtex as for other devices: place a pull-up resistor on the mode pin and wire a free running clock to the test clock input. In the event of an upset, the controller will return to the reset state in 5 clock cycles. It is important that the test clock input is not shared with any other pins on the Virtex to prevent contention on the clock in the presence of an upset, thereby preventing recovery. Though we have not observed a TAP controller upset in accelerator experiments on Virtex devices, the possibility must exist, as with any device using a JTAG implementation which leaves the reset signal inaccessible. The cross-section must be small in relation to dominant upset signatures.

The configuration management circuit also has a sensitive cross-section. When upset, the device behaves as though *PROGRAM* has been asserted, i.e., the configuration is cleared. Because of the infrequent nature of this upset mode we do not have a mitigation plan for it except to reconfigure and begin processing again with discarded data.

The configuration of the device pins used for the SelectMAP interface (*D0-7*, *WRITE*, *CS*, *BUSY*, etc.) may also be upset. This results in an inability to read or write configuration data to the Virtex. It can be detected when reading back bitstream data because a significant percentage of the bitstream will be wrong. Asserting *PROGRAM* and then reconfiguring the device can correct the problem.

3. SYSTEM DESIGN

SEU Simulation

An in-system SEU simulation/injection capability is extraordinarily useful in verifying the upset detection portion of a system. Partial configuration can be used to inject frames with single-bit errors into the Virtex device so the system’s behavior can be observed to understand upset consequences. This is most conveniently implemented with microprocessor access to the device SelectMAP interface and a source copy of the bitstream.

For our work, we have also used a PC-based SEU simulator to test the half-latch elimination scheme and to identify other bitstream-related SEU sensitivities, such as the sensitivity of the SelectMAP interface. The simulator, a USC/ISI SLAAC-

1V board [15] with custom software, loads the same design into two Virtex FPGAs and injects single-bit errors into one of the device’s configuration bitstream through partial configuration. A third FPGA is used to do a real-time comparison of the two FPGAs’ outputs to identify when a design’s function has been changed due to a simulated bitstream SEU. In the future, this simulator will be used to establish the usefulness of further SEU mitigation schemes. Figure 8 illustrates the test setup on the SLAAC-1V board.

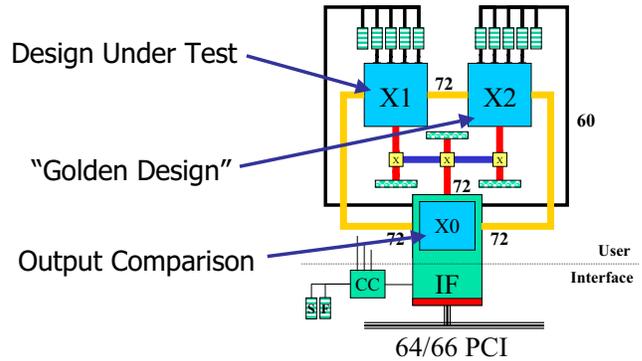


Figure 8. Test Setup for SEU Simulation on SLAAC-1V

For SRAM FPGA systems exposed to radiation, the possibility of internal contention that is initiated by a corrupted programming bitstream and causes permanent damage in the device is a concern. In our studies we have upset millions of bits, one at a time, and never observed a permanent fault. In addition, during static SEU testing a significant percentage of the bitstream was upset (25% or more), with the accumulated upsets resulting in larger (an increase of more than 1 Amp) standby current consumption. This condition persisted for a few minutes and was repeated about 10 times per device on several different devices. No permanent damage has ever been detected in our experiments, however, the devices in question have never been retested by the factory to guarantee full compliance with specification.

A concept for silicon testing the Virtex with a portfolio of configurations that exercise different elements of the architecture has been presented [16]. This has intriguing implications for the assurance of deployed systems. In addition, it is possible that an extension of this concept to fault identification would allow designing around a defect, perhaps squeezing more life out of a degraded system.

Device I/O Upset Implications

One important consideration for our system that SEU simulation revealed is that SEUs in the configuration data for bi-directional, tri-statable I/O pins can cause contention. The upset controller and the three Virtex FPGAs in our reconfigurable processing payload all share a local bus that the payload controller uses to interface to the module. When the SEU controller detects an error in the bitstream, it generates an in-

tterrupt to the payload controller, which then retrieves status information and clears the interrupt via register accesses over the local bus. When a Virtex device inadvertently drives this bus, the payload controller cannot repair the bitstream nor can it clear the interrupt. We manage this event by resetting the entire reconfigurable module, which clears the Virtex configurations and releases the bus. We have not observed any permanent faults in the Virtex or other components of the system in our simulations that result in this contention, but it is inadvisable to let the condition persist. Additionally, we have not observed any inputs being turned into outputs as a result of a single configuration upset. Exhaustive testing suggests single bit errors cannot have this effect [17].

Readback and Configuration Considerations

There are a few details to consider when performing readback on Virtex devices. First, do not perform readback on designs that use SelectRAM primitives such as SRL16. The combination of using LUTs as RAM and performing readback interferes with design operation and, potentially, device configuration—the two are incompatible.

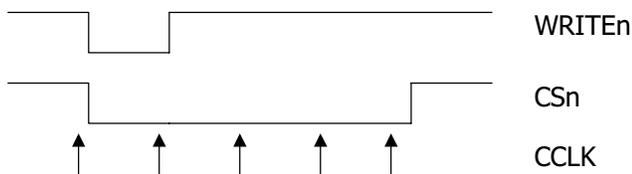


Figure 9. The Abort Sequence Performed Prior to Any Command Sequence Issued on the Virtex SelectMAP Configuration Interface

Second, always guarantee that the configuration controller is in a state ready for receiving configuration and readback commands. To do this, always perform an abort sequence before issuing commands to the device, as shown in Figure 9. That will insure that the FSM is in a known state each time commands are issued. Then, when issuing commands, always start with a dummy and sync word before sending configuration commands, as shown in Figure 10. This insures that the configuration FSM is resynchronized within the Virtex device so that the FSM correctly assembles the bytes on the SelectMAP interface into the proper 32-bit command words.

```
EXAMPLE READBACK COMMAND SEQUENCE
x"FF",x"FF",x"FF",x"FF", -- dummy word
x"AA",x"99",x"55",x"66", -- sync word
x"30",x"00",x"20",x"01", -- write to FAR
x"00",x"00",x"00",x"00", --
data: start frame address
x"30",x"00",x"80",x"01", -- write to CMD
x"00",x"00",x"00",x"04", -- data: RCFG
x"28",x"00",x"60",x"00", -- read from FDRO
x"48",x"02",x"D8",x"0D" -- data: # of data words
```

Figure 10. Example Readback Command Sequence for a Full Readback of a Virtex 1000 Device (Note the inclusion of the dummy and synchronization words.)

When commanding the Virtex for a readback, always read back the amount of data that was requested in the command sequence before issuing new commands. Otherwise, the internal counter providing the readback data will not reset, leaving the Virtex readback logic in a confused state. Recovery from this confused state is possible by performing a partial configuration of a frame after an aborted readback—this will reset the configuration controller’s internal counter. This recovery scheme may be useful when an SEU is detected midway through the readback of a device; in this case, the system can abort the readback and reconfigure the offending frame.

Pad data is required when partially configuring Virtex FPGAs. It is important that “0” values be used for pad data because the pipeline registers return some pad data in the readback stream. Using nonzero pad data may result in false detection of SEUs in the readback bitstream if pad data is not carefully masked out of the CRC calculation.

4. MITIGATION TECHNIQUE DEVELOPMENT

To this point, our methodology for developing SEU mitigation techniques that allow SRAM-based FPGAs to be employed in radiation environments can be summarized as a process which involves (1) device characterization, (2) mitigation technique development, and (3) mitigation automation. Each of these will be discussed briefly below.

Device Characterization

In this first step, the radiation tolerance of the SRAM-based FPGA is characterized. The device is tested for total ionizing dose effects and the immunity to single-event effects—latch-up (SEL), transients (SETs), and upsets (SEUs). The single-event effects characterization of devices is generally done in various particle accelerators to simulate radiation environments while total ionizing dose testing is normally done by exposing devices to a C_{o60} or similar source for accelerated dose testing.

As mentioned above, we believe that some of the SEU characterization experiments can and should also be done using partial configuration mechanisms in the Xilinx devices themselves. This latter approach can be very useful because it allows for precise, targeted, and inexpensive experiments regarding SEU effects—something that is not possible within the random process of irradiation at an accelerator. In fact, the speed and precision of the experiments that can be performed with the simulator have helped to uncover effects that were either too hard to distinguish or too small to be found in limited accelerator tests. Additionally, this simulation approach allows for inexpensive *dynamic* SEU testing of devices and user designs.

Mitigation Technique Development

As the devices are being characterized, techniques are then developed for mitigating any of the adverse effects of radiation. Of course, since we are assuming that the transistor-

level designs for the FPGAs have already been completed, this involves the creative use of existing devices rather than modifying the devices' low-level transistor designs. For FPGAs, mitigation techniques will generally involve the modification of a user's logic design (as with half-latch removal) or external support circuitry (as in the case of configuration bitstream upsets or the JTAG controller). These mitigation techniques will most likely pertain to SEUs and SETs since problems with total dose tolerance and SEL generally require modification of a device's silicon implementation, which cannot practically be done. Once developed, the mitigation techniques must be validated via the same simulation methods mentioned in the device characterization section to gain some confidence that the techniques will actually work.

Mitigation Automation

To make the above effort applicable to many designs and projects, the next step is to create FPGA design rules, automated design rule checking (DRC) tools, and automated design hardening tools to ready designs for radiation environments. For the Virtex FPGA, for instance, we have developed and are continuing to develop tools which automate half-latch removal, check for the presence of LUT RAM structures, and perform targeted FPGA resource tests for SEU sensitivities. We intend to expand these efforts to creating additional mitigation tools which automatically modify user designs to remove sensitivities to SEUs in the configuration bitstream and user design state. Again, these tools and their results must also be tested in a simulated radiation environment for validation.

5. FUTURE ARCHITECTURES

Of course, FPGA architectures continue to evolve and improve. Some of the benefits of newer, advanced FPGA architectures such as Virtex-II and Virtex-II Pro include reduced power consumption for a given application, increased device density, specialized hardware for computation, and, even, the addition of multi-gigabit serial transceivers (MGSTs) and on-chip microprocessors, in the case of Virtex-II Pro. With the MGSTs and on-chip microprocessors, reconfigurable computing systems will likely provide scalable and more integrated computing solutions than before. Integrating these next families of SRAM-based FPGAs should be very tempting propositions for future space-based computing projects.

With the increased computation and communication capabilities of these FPGAs come increased complexity in using the devices as well as managing their operation in a radiation environment. For Virtex-II Pro and its advanced CMOS process (130-nm, 9-layer copper with a low-k dielectric and high-speed 90-nm transistors), it is not perfectly clear how immune it might be to radiation effects. Additionally, the complex MGSTs and on-chip PowerPC 405 processors may or may not be useful in the presence of radiation, though, it is a possibility. For instance, in its biggest configuration, Virtex-II Pro will have 4 PowerPCs which might be usable in some

sort of TMR or TMR-with-spare arrangement. Further, the interfaces between the new advanced, fixed resources of these FPGAs and the normal FPGA resources may have SEU sensitivities which may be catastrophic in terms of device damage if the wrong bits are flipped. Though, this last scenario may be unlikely, a designer will have to learn quite a bit to be able to use these new FPGAs effectively.

The process described in the previous section for developing SEU mitigation techniques should be applicable to future radiation tolerant SRAM-based FPGA families, such as Virtex-II and Virtex-II Pro, despite the increased complexity of the devices. These newer FPGA architecture introduce hardware multipliers, larger RAM structures, bitstream encryption mechanisms, newly supported I/O standards, and, of course, microprocessors and multi-gigabit serial transceivers in the case of Virtex-II Pro. The complexity of these devices, we believe, will make SEU simulation via partial configuration more important, since the time and cost required for gaining a full understanding of the complex SEU behavior of these devices will be prohibitive if tested solely at an accelerator.

Thankfully, much of the work performed for understanding Virtex should have application to these newer architectures. In fact, these new architectures have much in common with the Virtex FPGA architecture and should have some common issues regarding SEU device cross-sections, including half-latches, on-chip RAMs, user flip-flops, configuration controllers, and configuration bitstreams. We expect the mitigation approaches for half-latches, on-chip RAMs, and other resources on these newer devices should be quite similar to those used in Virtex. Of course, the new chips with their new resources will also have some unique error signatures as well.

6. CONCLUSION

This paper has described various single-event upset categories for the Virtex FPGA, each category having unique observability properties. Bitstream upsets have excellent observability and simple mitigation, while half-latch upsets are not observable and require user design modifications for mitigation. A methodology for eliminating half-latch sensitivities has also been described. Additionally, known SEFI signatures are described along with suggestions for identifying and recovering from them. Several practical design considerations have been presented which should help others avoid various pitfalls relating to SEU mitigation and Virtex devices. We believe that we can successfully use Virtex FPGAs for processing our remote sensing data on orbit. Future work includes proton testing to validate the performance of our half-latch removal concepts, to verify the accuracy of the PC-based SEU simulator, and to take another look at the upset categories to be sure that all have been accounted for. Determining the presence of transient upsets, or their absence, will also be an objective of our proton testing.

Additionally, we have briefly discussed the challenges and

benefits of using newer FPGA architectures for space-based reconfigurable computing. Though the FPGAs promise to be more complex, much of the work done for the current, established family of radiation-tolerant Virtex FPGAs should be applicable to these newer FPGAs. Also, SEU simulation through partial configuration will be an important tool for better understanding the complex SEU behavior of these newer devices.

7. ACKNOWLEDGMENTS

This work was funded by the United States Department of Energy. Additionally, thanks are in order to Xilinx and, more specifically, Carl Carmichael and Joe Fabula for their support of this work.

REFERENCES

- [1] D. P. Lopresti, "Rapid implementation of a genetic sequence comparator using field-programmable gate arrays", in *Advanced Research in VLSI: Proceedings of the 1991 University of California/Santa Cruz Conference*, C. Sequin, Ed., Santa Cruz, CA, Mar. 1991, pp. 138–152.
- [2] J. Vuillemin, P. Bertin, D. Roncin, M. Shand, H. Touati, and P. Boucard, "Programmable active memories: Reconfigurable systems come of age", *IEEE Transactions on VLSI Systems*, vol. 4, no. 1, pp. 56–69, 1996.
- [3] M. Rencher and B. Hutchings, "Automated target recognition on Splash 2", in *Proceedings of IEEE Workshop on FPGAs for Custom Computing Machines*, J. Arnold and K. L. Pocek, Eds., Napa, CA, Apr. 1997, pp. 192–200.
- [4] J. Villasenor and W. H. Mangione-Smith, "Configurable computing", *Scientific American*, pp. 66–71, June 1997.
- [5] N. Shirazi, P. M. Athanas, and A. L. Abbott, "Implementation of a 2-D Fast Fourier Transform on an FPGA-based custom computing machine", in *Field-Programmable Logic and Applications. 5th International Workshop on Field-Programmable Logic and Applications*, W. Moore and W. Luk, Eds., Oxford, UK, Sept. 1995, pp. 282–292, Springer-Verlag.
- [6] P. Graham and B. Nelson, "FPGA-based Sonar Processing", in *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, Monterey, CA, February 1998, ACM/SIGDA, pp. 201–208, ACM.
- [7] 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.
- [8] 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.
- [9] C. Carmichael, "Triple module redundancy design techniques for Virtex FPGAs", Tech. Rep., Xilinx Corporation, November 1, 2001, XAPP197 (v1.0).
- [10] F. Lima, C. Carmichael, J. Fabula, R. Padovani, and R. Reis, "A fault injection analysis of Virtex FPGA TMR design methodology", in *Proceedings of the 6th European Conference on Radiation and its Effects on Components and Systems (RADECS 2001)*, 2001.
- [11] E. Fuller, M. Caffrey, A. Salazar, C. Carmichael, and J. Fabula, "Radiation testing update, SEU mitigation, and availability analysis of the Virtex FPGA for space reconfigurable computing", in *3rd Annual Conference on Military and Aerospace Programmable Logic Devices (MAPLD)*, 2000, p. P30.
- [12] 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).
- [13] C. Carmichael, E. Fuller, J. Fabula, and F. D. Lima, "Proton testing of SEU mitigation methods for the Virtex FPGA", in *4th Annual Conference on Military and Aerospace Programmable Logic Devices (MAPLD)*, 2001, p. P6.
- [14] 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.
- [15] B. Schott, S. Crago, R. Parker, L. Carter, C. Chen, J. Czarnaski, M. French, T. Tho, and T. Valenti, "Reconfigurable architectures for system-level applications of adaptive computing", *VLSI Design*, vol. 10, no. 3, pp. 265–279, 2000.
- [16] P. Sundararajan, "Testing FPGA devices using JBits", in *Proceedings of the 4th Annual International Conference on Military and Aerospace Programmable Logic Devices (MAPLD)*, September 2001, p. E5.
- [17] N. Rollins, M. Wirthlin, M. Caffrey, and P. Graham, "Reliability of programmable Input/Output pins in the presence of configuration upsets", in *Proceedings of the 5th Annual International Conference on Military and Aerospace Programmable Logic Devices (MAPLD)*, September 2002, To Be Published.



Paul Graham is a member of the technical staff at Los Alamos National Laboratory. His research interests include design automation tools for FPGA-based reconfigurable computing, applications of reconfigurable computing, and FPGA architecture. Dr. Graham obtained an M.S. degree in electrical engineering from Brigham Young University in 1996, and a Ph.D. in Electrical Engineering from Brigham Young University in 2001. He is a member of IEEE.



Michael Caffrey is a member of the technical staff at Los Alamos National Laboratory. His current work involves the production of a FPGA-based reconfigurable computing platform for space and has interests in reconfigurable computing and digital signal processing. Mr. Caffrey obtained an M.S. degree in electrical engineering from the University of New Mexico in 1995.



Michael Wirthlin is an associate Professor at Brigham Young University in Provo, UT. His research interests include tools and applications for FPGA-based reconfigurable computing, power efficiency in FPGA designs, and synthesis techniques for digital systems. Dr. Wirthlin obtained a Ph.D. in electrical engineering from Brigham Young University in 1997.



D. Eric Johnson is an undergraduate student in the electrical and computer engineering program at Brigham Young University with interests in digital signal processing and reconfigurable computing.



Nathan Rollins is an undergraduate student in the electrical and computer engineering program at Brigham Young University with interests in reconfigurable computing.