LA-UR-

*Approved for public release;
distribution is unlimited.*

Title:

Author(s):

Submitted to:

Los Alamos
NATIONAL LABORATORY

Consequences and Categories of SRAM FPGA Configuration SEUs

Paul Graham[1], Michael Caffrey[1], Jason Zimmerman[1], Prasanna Sundararajan[2], Eric Johnson[1], and Cameron Patterson[2]

[1]NIS-3, Los Alamos National Laboratory
[2]Xilinx Research Laboratories, Xilinx, Inc.

Abstract— **Understanding the SEU induced failure modes specific to the Virtex SRAM FPGA is needed to evaluate the applicability of various mitigation schemes since many mitigation approaches were originally intended for ASICs and may not be effective or efficient within FPGAs due to the unique failure modes and architectures found in SRAM-based FPGAs. Through this work, we have shown that SEUs in FPGAs' programming data may result in five main categories of design modifications, specifically, changes in: mux select lines, programmable interconnect point states, buffer enables, LUT values, and control bit values. These effects are fairly unique to SRAM FPGAs and occur in addition to SEUs in a design's memory elements. Through analyzing and classifying the bitstream-SEU-induced circuit failures for some test designs, we have been able to confirm and/or discover the following for the test designs: (1) failures in routing structures account for most of the design failures (78% to 84.8% of the failures); (2) the remaining 20% (approximately) were due to upsets in control bits and LUT value changes; (3) of the failure modes, routing mux changes have the most significant impact on bitstream SEU reliability, accounting for as many as 73% of the test designs' sensitive programming bits; and, (4) the elimination of any single failure mode will not result in a 10x improvement in SEU reliability.**

I. INTRODUCTION

Reconfigurable FPGAs promise performance, flexibility, and schedule benefits to the space community. The main challenge to using the current generation of radiation-tolerant SRAM FPGAs is that they are based directly on commercial designs, meaning that they are sensitive to single-event upsets (SEUs) in rather dramatic terms [1]–[3]. Upsets can occur both in the FPGA programming data and the on-chip user memory structures, leading to changes in the user's circuit and the state of the user's data, respectively. Given the size of the configuration memory, SRAM-based FPGAs result in extremely sensitive designs if no mitigation is employed.

Previous papers [1], [4]–[14] have described the SEU characteristics and mitigation techniques for SEU sensitive structures within the Xilinx Virtex and XQVR families of FPGAs. Mitigation techniques have been developed for configuration and user memory structures, "half-latch" structures, and other SEU sensitive on-chip circuits. These mitigation techniques involve mainly system-level and user-design-level changes to improve SEU-related reliability.

Several [6], [8], [13], [14] of these studies have explored SEU mitigation through redundancy for SRAM FPGA designs. These techniques provide additional protection to the integrity of an application's on-chip data by replicating both circuitry

and data storage. In these studies, triple-modular redundancy (TMR) is applied in a variety of ways to improve a circuit's reliability.

As these papers suggest, if one assumes that leveraging commercial technology is beneficial and that not every application requires near 100% reliability, a "spectrum" of mitigation strategies may exist allowing designers to trade the costs of power and processing bandwidth with SEU reliability. Though a full TMR solution can be very robust, a full TMR solution, in some cases, may be too expensive in logic or I/O resources or in terms of power that such a solution is not practical. Clearly, more SEU mitigation choices that fill in the mitigation strategies "spectrum" will be beneficial to the space community.

To better understand what SEU mitigation techniques may be effective, we have extensively tested the Virtex FPGA for failure modes caused by upsets in its programming (or *configuration*) data. An extensive database of failures, for a variety of test designs, was compiled using a high performance SEU simulator together with radiation testing. This paper analyzes the errors to explain the failure modes of Virtex FPGAs related to programming data SEUs. Understanding the failures may lead to FPGA-specific mitigation techniques and may provide insight into why some traditional SEU mitigation techniques may not be applicable to SRAM-based FPGAs. Further, FPGAs are complex devices and no design uses all the resources in the device; thus, there may be opportunities to exploit the FPGA architecture to gain some reliability at moderate cost in power or processing bandwidth.

II. BACKGROUND

Before detailing the failure modes of the Virtex family due to programming data SEUs, we will briefly cover the architecture of this FPGA family and the SEU simulation technology used to analyze the test designs.

A. Virtex Architecture

As mentioned in [15] and [16], the Xilinx Virtex family of FPGAs provides a variety of resources for implementing user designs, including programmable logic, I/O, routing, clocking, and RAM resources. Figure 1 provides a simplified top-level overview of the architecture. The edge of the chip is surrounded by programmable I/O block (IOB) resources. Near the edge of the chip are Block SelectRAM (or simply BlockRAM)

resources providing configurable 4 Kb memories. At the center of the chip is a two-dimensional array of configurable logic blocks (CLBs), the main type of programmable logic resource. Each CLB consists of two *slices* and each slice has two 4-input look-up tables (LUTs), two flip-flops, carry logic, and routing. Figure 2 illustrates the resources and programmability of a single slice. The "bits" labeled with single letters represent the programming data bits used to control the function of the slice.
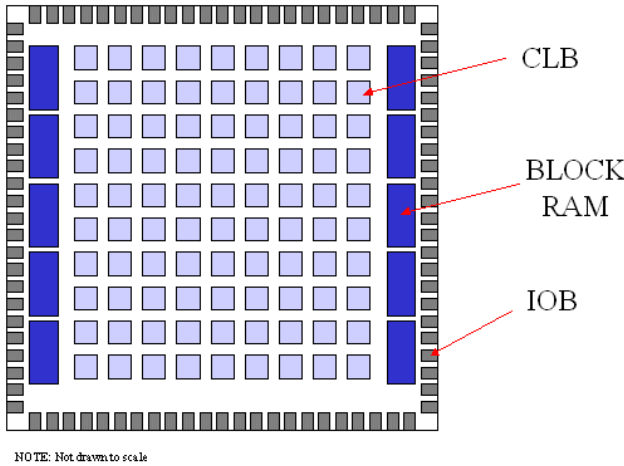

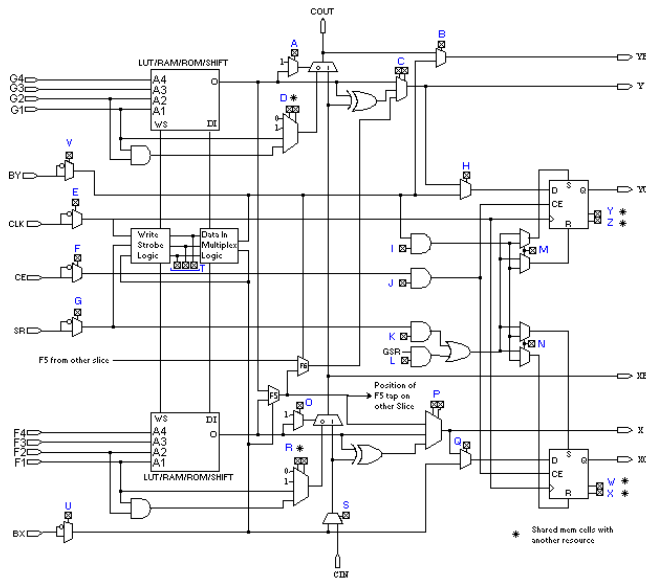Fig. 1. Simplified View of Xilinx Virtex FPGA [16]

Fig. 2. One Slice from the Virtex Architecture's CLB [16]

Of course, in addition to these resources, a large amount of programmable routing is available for interconnecting the logic, I/O, clocking, BlockRAMs, and other resources used by a designer. Figure 3 provides a simplified view of the Virtex routing architecture within the CLB array. The routing for the CLB array is made up mainly of wires that connect adjacent CLBs (single-length wires) and those that connect CLBs that are 6 rows or columns away (hex-length wires)[1]. The *Single Switch Box* in the figure, which provides the programmable interconnection to single-length wires, consists of a matrix of programmable interconnect points (PIPs). Each PIP is a pass transistor which can connect two wire segments. The *Hex Switch Box* in Figure 3, which provides the programmable interconnection to hex-length wires, consists mainly of multi-plexers (or *muxes*) and buffers. The buffers in the *Hex Switch Box* are only associated with bi-directional hex-length wires. The *Main Switch Box* consists of a combination of PIPs, buffers, and muxes.
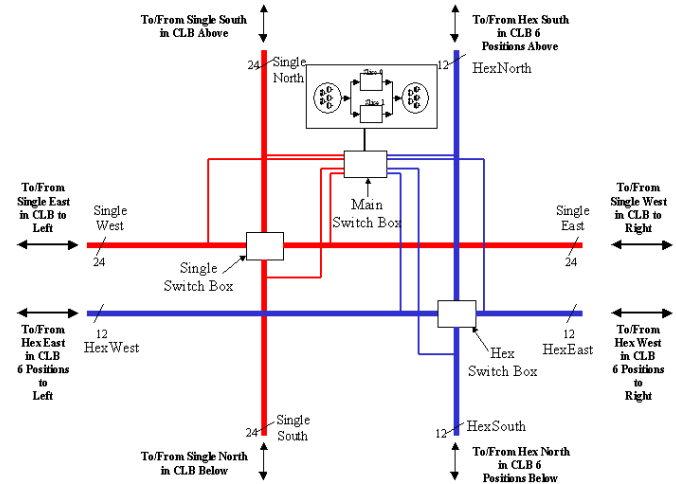

Fig. 3. Simplified CLB Routing Architecture for Virtex [16]

Immediately around the two slices within a CLB are collection of input muxes (*imuxes*) and output muxes (*omuxes*) which are used to route input and output signals to and from the slices. Figure 4 provides an exploded view of this form of programmable routing.

Based on this brief overview, the contents of the configuration data for an SRAM FPGA define the values held in LUTs and RAMs, the interconnection between resources, and the modes of the resources (e.g., I/O standards, I/O drive strengths, LUT modes, etc.).

B. Virtex SEU Simulator

In this analysis work, we have used a PC-based bitstream SEU simulator [17] created by Brigham Young University and Los Alamos National Laboratory to identify bitstream-related SEU sensitivities. The simulator is composed of a USC/ISI SLAAC-1V FPGA board [18] and some custom software. During operation, the simulator loads the same design into two Virtex FPGAs (labeled *X1* and *X2* in Figure 5) and injects single-bit errors into one of the device's configuration bitstream through partial configuration. A third FPGA is used to provide input data and do a real-time comparison of the two

---

[1] Some hex-length wires have mid-wire connections that allow CLBs to connect other CLBs which are only three rows or columns away.
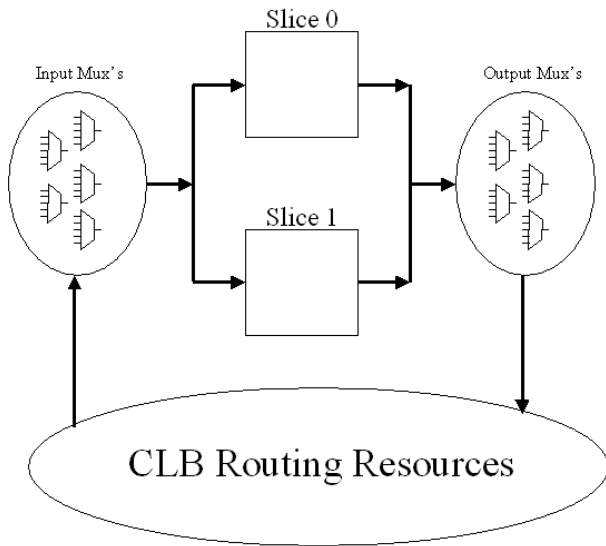
Fig. 4. Input and Output Muxes Surrounding the Slices within a Virtex CLB [16]

FPGAs' outputs to identify when a design's function has been materially changed due to a simulated bitstream SEU. This hardware-based simulation of bitstream SEUs has proven to be over 97% accurate in predicting bitstream SEU sensitivities [17] and provides us with a flexible, inexpensive method for performing bitstream SEU studies.
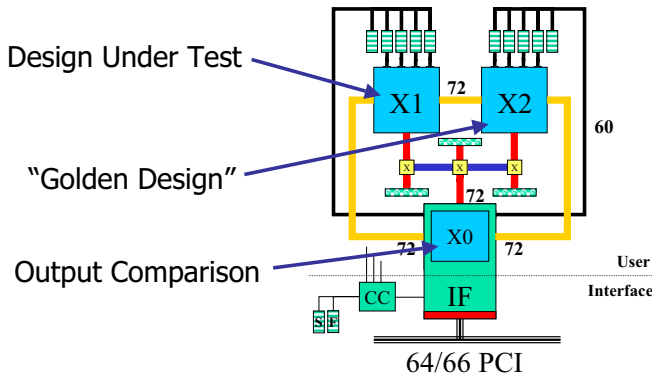

Fig. 5. Test Setup for SEU Simulation on SLAAC-1V

III. FPGA BITSTREAM SEU FAILURE MODES

FPGAs have many SEU-induced failure modes that conventional 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. An input may suddenly be coming from somewhere else. There are many ways to possibly categorize the failure modes. One could do a broad classification based on "routing errors" vs. "logic errors". On the other hand the classification could go to a deeper level; each configuration bit has a specific function, such as turning off a buffer, and failures could be

classified at that level. For this project, seven specific failure modes were classified and considered: mux select, PIP short, PIP open, buffer off, buffer on, LUT value change, and control bit change.

In the Virtex FPGA, multiplexers are a large part of the routing network. Among other things, all circuit inputs and outputs are multiplexed. Multiplexers are very sensitive to SEUs because any change in their select lines will cause a different routing configuration. An example mux select failure is shown in Figure 6. Note that some of the multiplexers in the Virtex architecture are sparsely encoded, and it might be possible to take advantage of that sparseness, but doing that would require complete characterization of each kind of multiplexer on the chip.

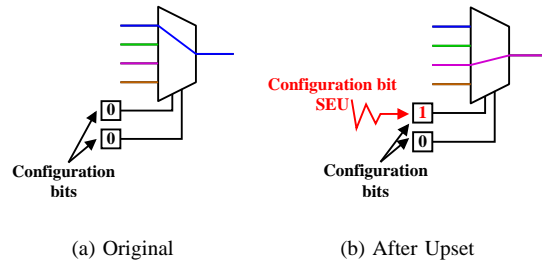(a) Original   (b) After Upset

Fig. 6. Mux Select Failure Example

Another main component of the Virtex routing network is the programmable interconnect point, or *PIP*. A PIP is a pass transistor between two wires that can either be on or off. Thus, the wires are either connected or not connected. PIPs can cause a few different kinds of SEU-induced failures. The first, shown in Figure 7(b), is called a PIP short failure. This is when two wires with different functions in the design are shorted together. A PIP short can produce contention, causing output errors and increased power consumption. The second type of PIP failure is shown in Figure 7(d) and is called a PIP open. This occurs when a PIP, normally turned on in the design, effectively breaks a wire into two pieces. A PIP open causes an interruption in the flow of information from one part of the design to another.

A third, less frequent PIP failure mode is possible, which we call a PIP load failure. In theory, when a normally off PIP between an active wire and an unused wire is turned on, output errors may result from a change in circuit timing due to additional loading on the active wire. Without an analysis of which wires are connected when a PIP is turned "on", it is extremely difficult to distinguish PIP load failures from PIP short failures. Due to the relatively slow speed at which the designs were executed (20 MHz) and the fact that no automated PIP connectivity analysis tool was available, this paper does not consider the effects of the third failure mode in the later analysis. Any PIP load failures that may have been in our data would be classified as PIP shorts. PIPs are seen most often in single-length and edge routing.

The final component of the Virtex routing network considered here is the buffer. A buffer in this context is a driver which can either be turned on or off. As shown in Figure 8,
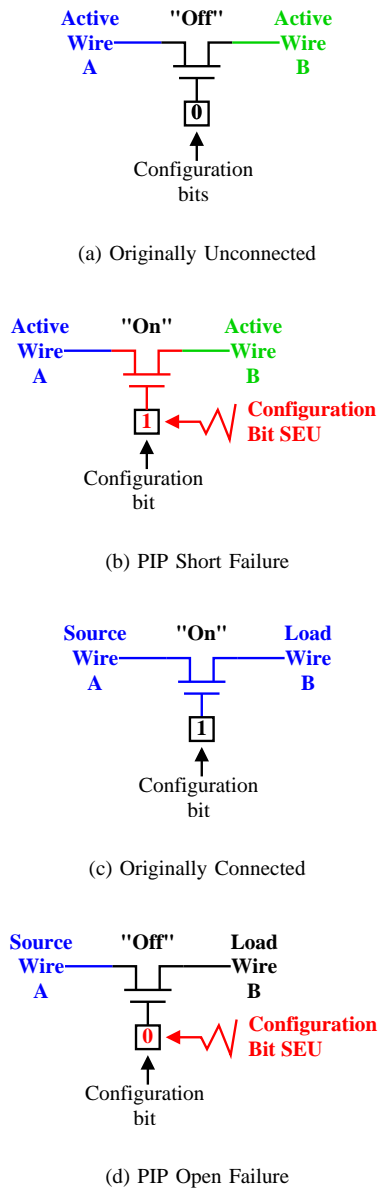
(a) Originally Unconnected

(b) PIP Short Failure

(c) Originally Connected

(d) PIP Open Failure

Fig. 7.   PIP Failure Mode Examples

(a) Originally Unconnected

(b) Buffer Short Failure

(c) Original Connected

(d) Buffer Open Failure

Fig. 8.   Buffer Failure Mode Examples

the buffer has two failure modes associated with it and they are very similar to the PIP failures. The main difference here is that instead of a pass transistor, the failure is being 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. Buffers usually are placed on the outputs of some multiplexers and on bi-directional wires.

The two remaining failure modes, LUT value changes and control bit changes, are related to logic resources rather than to the routing network. The Virtex FPGA uses 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. A good example
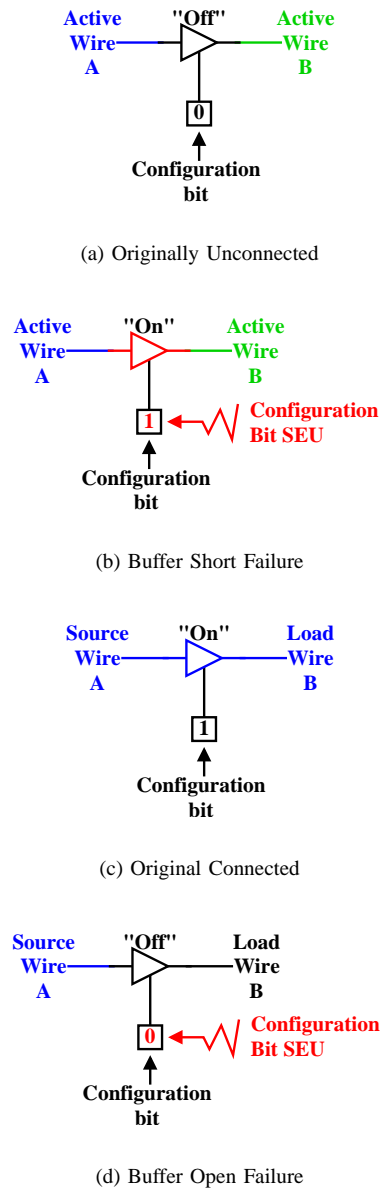
is shown in 9. Here the LUT implements a 4-input AND function. If the one bit that defines the "true" condition is upset, the result is a constant-zero function. For most inputs, the output of the function would still be correct, however there is the one case that would cause problems.

In contrast to LUT value changes, control bit changes generally cause errors for all, or almost all, possible circuit inputs. The Virtex CLBs and IOBs use quite a few control bits to determine miscellaneous functionality. Figure 10, which is an enlarged portion of Figure 2, gives some clues as to what can happen when a control bit is upset. 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, control the powerful Virtex LUT functionality. They determine whether the LUT

LUT

	F1	0	0	0	0
1	F2	0	0	0	0
1	F3	0	0	0	0
1	F4	0	0	0	1

1   O   **1**

O=F1*F2*F3*F4

(a) Original 4-input AND Function

LUT

	F1	0	0	0	0
1	F2	0	0	0	0
1	F3	0	0	0	0
1	F4	0	0	0	**0**

1   O   **0**

← **Configuration Bit SEU**

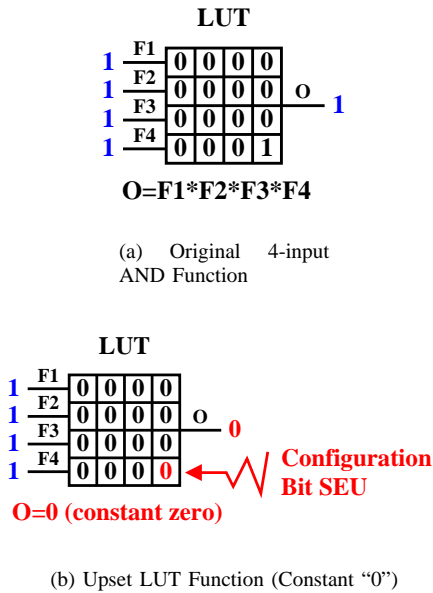**O=0 (constant zero)**

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

Fig. 9. LUT Upset Example

in this 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.
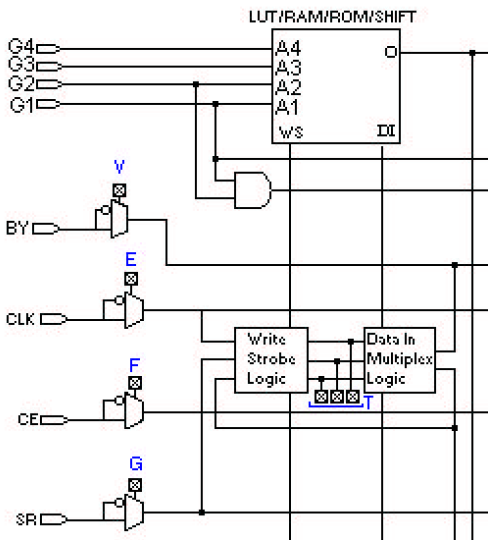

Fig. 10. Control Bit Examples [16]

One additional category that is present in the results given later is the "unclassified" category. This encompasses all configuration bits for which there was not enough to time to fully classify and place into the above categories. As demonstrated later, the unclassified bits account for less than 1% of the total sensitive bits for the larger designs.

Lastly, no attempt was made in this study to quantify the effects of SEUs in the user-visible portions of the Virtex

FPGA's configuration logic, such as the configuration registers. To some extent, modeling upsets in these registers should be possible with our bitstream SEU simulator. Upsets in this state might be classified as "control" or possibly might be better classified as a separate category, such as "configuration control" upsets.

IV. CLASSIFICATION METHODOLOGY

Figure 11 shows how the functional failure classification is done, starting with just a design bitstream. First, the design is run through our SEU simulator. The simulator tells us which configuration bits in the design are sensitive to upsets. These "sensitive bits" will cause errors in the output of the circuit at runtime, and thus are the important configuration bits to examine. However, the simulator only tells us which of the five million configuration bits is sensitive, not what their function is.

To be able to classify the failure mode we must have some idea of what part of the circuit that particular bit affects. Therefore, the next step in the flow diagram is a process that associates information from a Xilinx-proprietary database with each sensitive bit from the simulator. Each bit affects a specific part of a specific circuit on the FPGA. The proprietary information tells us the physical location of that circuit and the function for each bit. Once the function of the bit has been determined, it can easily be classified into one the general categories mentioned above in Section III: mux select upset, PIP upset, buffer upset, LUT upset, control bit upset, unclassified upset.

There is one more component to the classification: we need to know whether the sensitive bit is "turned on" or "turned off" in the design. For example, the functional name can tell us if the configuration bit in question controls a PIP, but not whether the PIP was shorted or opened. If a PIP was "turned on" in the original design, then the failure mode is classified as an "open PIP", but if the PIP was "turned off", the failure would be a "shorted PIP". This additional piece of information is important in classifying PIP and buffer failures. The lower path of the flow diagram shows how this "on-bit" information is obtained by running the design bitstream through another tool that provides the functions of the "on" bits using Xilinx-proprietary information. The final step in classifying the functional failures is the use of a custom tool that pulls together the sensitive bit functions, the "on-bits" information, and the failure classifications to produce the full-design classification. The output of this tool includes a text file matching each sensitive bit with its failure mode, some statistics, and one or more distributions in the form of pie charts.

In the future, the tool flow could be improved to distinguish PIP load failures from PIP short failures. This could possibly be done any number of ways; one way involves leveraging the JBits API. A design could be fed into JBits along with the "sensitive shorted PIPs" information and it might be possible to determine a representation of what each affected wire is connected to, and whether the PIP failure is actually a short or merely a load. An alternative to JBits would be to use the
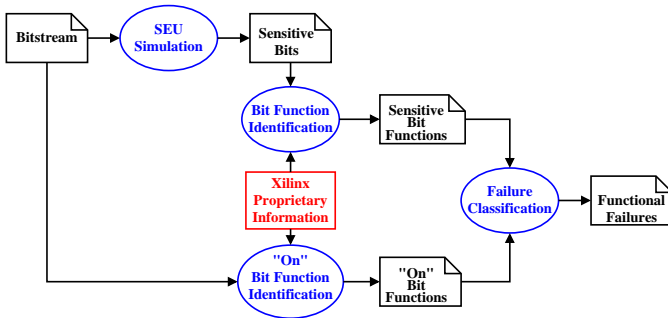
Fig. 11.   Bitstream SEU Failure Classification Process

TABLE I

DEVICE UTILIZATION FOR THREE TEST DESIGNS. THE LAST ROW SHOWS
THE RESOURCES AVAILABLE IN THE XCV1000 FPGA.

Design	Slices	LUTs	Flip-Flops	Sensitive Config. Bits
8-bit Counter	4	8	8	389
72Mult	8,308	10,872	15,264	392,166
72LFSR	8,712	576	8,640	5,810,048
XCV1000	12,288	24,576	24,576	1,175,036

Fig. 12.   72Mult test design.

Fig. 13.   72-bit Linear Feedback Shift Register design.

Xilinx Design Language (XDL) representation of the circuit. XDL takes a simplistic view of the routing structure, making all connections appear as PIPs. It may be easier to create a program that parses through these connections to determine what is actively driving each wire. The classifications themselves can also be refined, to further reduce the size and increase the understanding of the unclassified category.

V. TEST DESIGNS

For this study, three different designs were used to better understand the frequency of the different bitstream-SEU-induced failure modes. A summary of the statistics for each design is provided in Table I. The first design was a simple 8-bit counter used mainly to verify our automated classification technique. The other two designs are significantly larger and represent two different types of designs. The *72Mult* design, shown in Figure 12, is a series of 8 36-bit multipliers followed by an adder tree, representing a feed-forward DSP-style architecture. The *72LFSR* design, shown in Figure 13, is a large collection of 432 20-bit LFSRs whose outputs are combined to create a 72-bit output value. The LFSR design has design elements with feedback (the LFSRs) and is more representative of state machines and other circuits with tight feedback.

The last column of the table provides the number of SEU-sensitive configuration bits identified by the SEU simulator. For the cases of the *72Mult* and the *72LFSR*, about 2 *billion* bitstream upsets within the SEU simulator were performed to identify which configuration bits caused design output errors when upset as well as the probability that each sensitive bit will cause an output error. For instance, while only 20% of the configuration bits in the *72Mult* design are sensitive, only about 60% of these bits cause an error every time. For the rest of the sensitive bits, whether or not they cause an error in the design's output is much more dependent on the data
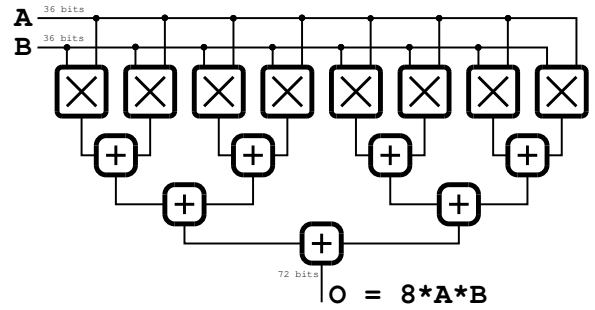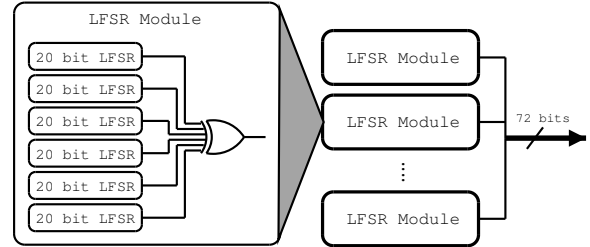
or operation being performed at the time of upset. For more statistics regarding the sensitivity signatures for the two larger designs, please refer to [17] for more information.

These designs mainly exercise the internal CLB array and routing architecture of Virtex FPGAs and do not heavily exercise the various IOB configurations and do not use Block-RAMs or Virtex's advanced clocking features (e.g., delay-locked loops). With this said, they do represent a good mixture of the CLB related logic and routing features, which generally dominate the SEU sensitivities of designs.

Further, none of the designs used logic redundancy to provide any form of bitstream SEU robustness, hence, the large number of SEU sensitive configuration bits. The only form of SEU robustness which was added to each design was to remove the use of half-latches to provide internal logic constants—this is necessary so that half-latch upsets do not appear to be bitstream SEU sensitivities.

VI. CLASSIFICATION RESULTS

Table II provides a summary of the bitstream SEU classifications for the three designs. Several significant trends can be observed in the data. First, as one might expect in routing-rich FPGA architectures such as Virtex, routing resources (muxes, PIPs, and buffer resources) account for 78% or more of the sensitive bits for each design while the LUT and control configuration bits account for between 15 and 21% of the sensitive bits. Within the routing category, the dominant failure type was the mux select upset, ranging from 58% to 73% of the total sensitive bits.

Within the table, two sets of percentages are provided for the larger two designs: one is weighted according to the probability that an upset in a particular bit will cause an output error and another set that is unweighted with respect to this probability. Though the weighting based on error probability

TABLE II
SUMMARY OF BITSTREAM SEU CLASSIFICATION FOR TEST DESIGNS

Design	Control	LUT	Mux Select	PIP	Buffer	Unclassified	Routing	LUT/ Control	Total
8-bit Counter (unweighted)	12%	4%	59%	16%	7%	2%	82%	16%	389
72LFSR (unweighted)	13%	2%	73%	10%	1.8%	<1%	84.8%	15%	392,166
72LFSR (weighted)	17%	<1%	70%	11%	<2%	<1%	82%	18%	392,166
72Mult (unweighted)	7%	14%	58%	19%	1%	<1%	78%	21%	1,175,036
72Mult (weighted)	9%	9%	59%	22%	<2%	<1%	82%	18%	1,175,036

has an effect on the data, the general distributions do not significantly change.

Figures 14, 15, and 16 illustrate these failure distributions more visually and include some additional granularity in the classifications. The following color code is used in the figures for the different more general failure classes:

- purple—LUT upset failures,
- red—control bit upset failures,
- green—mux select failures,
- dark-blue—buffer upset failures,
- light-blue—PIP upset failures, and
- orange—unclassified.

Within the "mux" category, seven subcategories exist:

- *slice_omux*—the output muxes associated with the slices,
- *slice_imux*—the input muxes associated with the slices,
- *slice_mux*—the muxes internal to the slice,
- *iob_omux*—the output muxes associated with the IOBs,
- *iob_imux*—the input muxes associated with the IOBs,
- *iob_mux*—the muxes internal to the IOBs, and
- *hex_mux*—the muxes in the hex-length wire switch boxes.

Fig. 15. Distribution of SEU Sensitive Bitstream Bits for *72LFSR* Weighted by Probability of Error
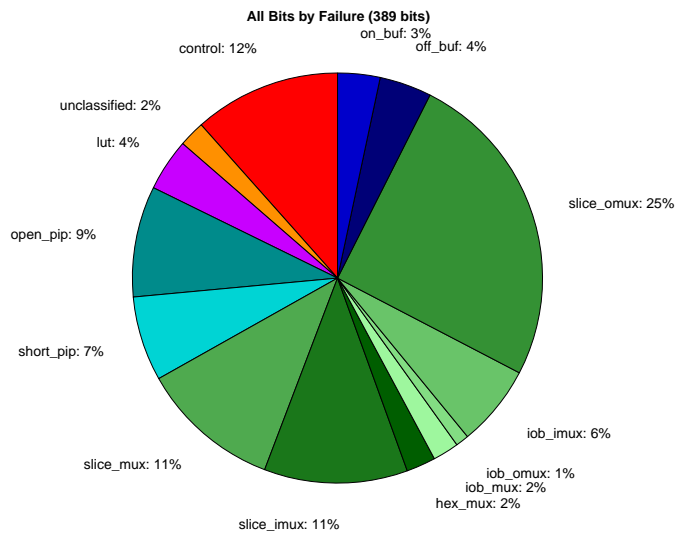

Fig. 14. Distribution of SEU Sensitive Bitstream Bits for 8-bit Counter

With this additional information, it is clear that two particular types of resources appear to dominate the failures: slice input muxes and slice output muxes. The importance of these resources is not very surprising since these muxes are crucial for properly routing the inputs and outputs for slices—resources where almost all of a circuit logic is performed.
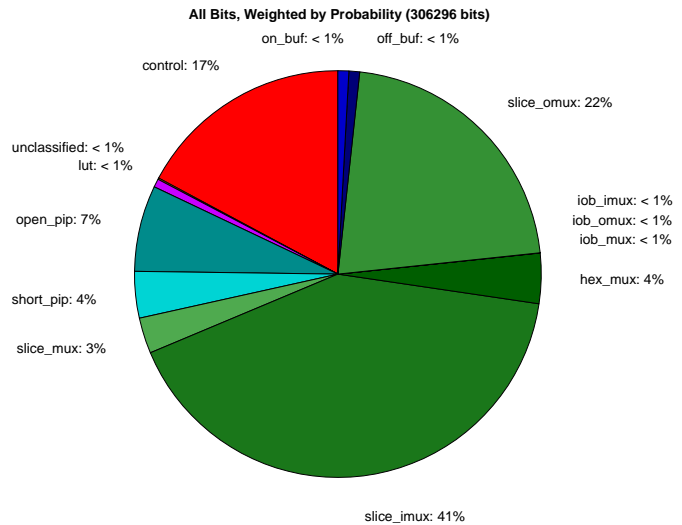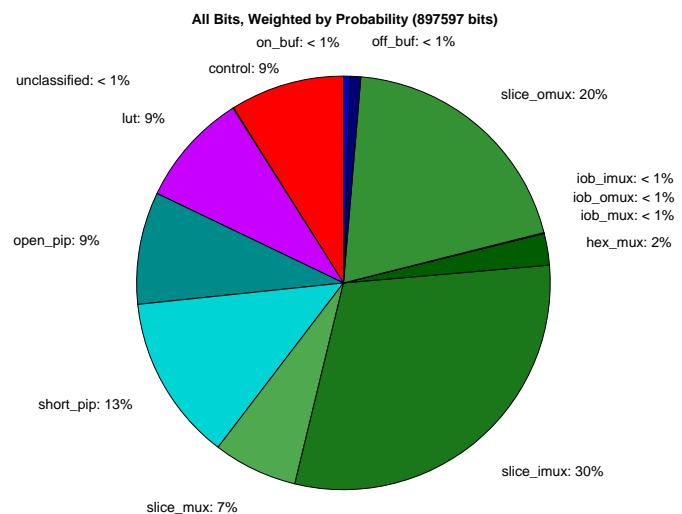

Fig. 16. Weighted Distribution of SEU Sensitive Bitstream Bits for *72Mult* Weighted by Probability of Error

Further the sensitivity of these resources is directly related to the fact that these muxes are reasonably wide (ranging from 12 to 28 input bits) so multiple configuration bits determine the routing of a single signal. If even one of the bits is upset, the slice input or output (as appropriate) will be routed differently, thus, disturbing the circuit.

Finally, Table III provides some information on the distribution of the failure types for those bits which cause output errors every time if upset. While the distributions are clearly design dependent, they still follow the trends noted above—mux select failures dominate and around 75% or more of the failures are routing related. From this analysis, it is also clear that very few LUT bits are guaranteed to cause output failures every time when upset. By contrast, control bit upsets are more prone to cause failures if upset.

VII. Conclusions

Understanding the SEU induced failure modes specific to the Virtex SRAM FPGA is needed to evaluate the applicability of various mitigation schemes since many mitigation approaches were originally intended for ASICs and may not be effective or efficient within FPGAs due to the unique failure modes and architectures found in SRAM-based FPGAs. Further, we do believe that additional work regarding mitigation schemes other than just full TMR is beneficial since the reliability requirements differ from mission to mission or application to application and being able to trade off power and/or computational power for SEU reliability may make sense in some systems and designs.

Through this work, we have shown that SEUs in FPGAs' programming data may result in five main categories of design modifications, specifically, changes in: mux select lines, programmable interconnect point states, buffering enables, LUT values, and control bit values. The possible effects of these changes to the routing portions of these chips include re-routing nets, shorting independent active nets, breaking (opening) existing nets, and adding additional capacitive loading that can affect design timing. Changes to LUT values can affect the logic functions LUTs provide or, in the case of LUT RAM, affect the contents of the design's RAMs. Upsets that affect control bits in an FPGA's programming data can have drastic effects, such as changing an I/O standard, modifying the programmable inversion of a signal, or changing the mode of operation for a delay-lock loop or LUT circuit. These types of effects are fairly unique to SRAM FPGAs and occur in addition to SEUs in a design's memory elements.

To better understand the frequency of these failure types for general FPGA designs, the sensitive configuration bits of three designs were identified and analyzed. These designs specifically did not employ SEU mitigation through redundancy so that the failure modes could be clearly identified and analyzed. As a result of this analysis, several things have been confirmed or discovered. First, as expected, failures in routing structures account for most of the design failures seen in our test designs, ranging from 78% to 84.8% of the failures. The remaining 20% (approximately) of the observed design failures were related to upsets in control bits and LUT value

changes. Another outcome is that, while routing muxes have the most significant impact on reliability (accounting for as much as 73% of the sensitive bits in the programming data), the elimination of any single failure mode will not result in a 10x improvement in SEU reliability.

Typically, FPGA designs leave many resources in the device unused. The surplus routing and logic may be useful for low-cost mitigation insertion. Now that we understand the SEU failure modes and have a way of evaluating the types of errors that occur, future efforts will examine techniques that exploit the unused resources of the FPGA for additional reliability.

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] G. Swift, C. Yui, and C. Carmichael, "Single-event upset susceptibility testing of the Xilinx Virtex II FPGA", in *Proceedings of the 5th Annual International Conference on Military and Aerospace Programmable Logic Devices (MAPLD)*, Laurel, MD, September 2002, NASA Office of Logic Design.

[3] M. Ceshia, M. Bellato, A. Paccagnella, S. C. Lee, C. Wan, A. Kaminski, M. Menichelli, A. Papi, and J. Wyss, "Ion beam testing of Altera Apex FPGAs", in *Proceedings of the 2002 IEEE Radiation Effects Data Workshop*, S. Crain and T. Turflinger, Eds., Phoenix, AZ, July 2002, IEEE Nuclear and Plasma Sciences Society, pp. 45–50, IEEE.

[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] 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.

[6] 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.

[7] 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.

[8] C. Carmichael, "Triple module redundancy design techniques for Virtex FPGAs", Tech. Rep., Xilinx Corporation, November 1, 2001, XAPP197 (v1.0).

[9] W.-J. R. Huang, *Dependable Computing Techinques for Reconfigurable Hardware*, PhD thesis, Stanford University, June 2001.

[10] 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.

[11] P. Graham, M. Caffrey, M. Wirthlin, D. E. Johnson, and N. Rollins, "Reconfigurable computing in space: From current technology to reconfigurable systems-on-a-chip", in *Proceedings of the 2003 IEEE Aerospace Conference*, Big Sky, MT, March 2003, pp. T07_0603.1–12, IEEE.

[12] P. Graham, M. Caffrey, M. Wirthlin, D. E. Johnson, and N. Rollins, "SEU mitigation for half-latches in xilinx virtex fpgas", December 2003, Submitted.

[13] 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., September 2003, NASA Office of Logic Design, AIAA, p. P63.

TABLE III

SUMMARY OF BITSTREAM SEU CLASSIFICATION FOR THE SENSITIVE BITSTREAM BITS THAT CAUSE FAILURES 100% OF THE TIME.

Design	Control	LUT	Mux Select	PIP	Buffer	Unclassified	Routing	LUT/ Control	Number of bits	% of total sensitive bits
72LFSR	24%	<1%	65%	9%	<2%	<1%	75%	24%	217,235	55%
72Mult	11%	<1%	66%	21%	<2%	<1%	88%	11%	705,734	60%

[14] P. K. Samudrala, J. Ramos, , and S. Katkoori, "Selective triple modular redundancy for SEU mitigation in FPGAs", in *Proceedings fo the 6th Annual International Conference on Military and Aerospace Programmable Logic Devices (MAPLD)*, Washington, D.C., 2003, NASA Office of Logic Design, AIAA, p. C1.

[15] Xilinx, "Virtex 2.5V field programmable gate arrays", Data Sheet DS003-1, Xilinx, San Jose, CA, April 2001.

[16] Xilinx, "Virtex architecture guide", Tech. Rep., Xilinx, San Jose, CA, September 2000, Provided as a part of the JBits 2.8 SDK.

[17] E. Johnson, M. Caffrey, P. Graham, N. Rollins, and M. Wirthlin, "Accelerator validation of an FPGA SEU simulator", *IEEE Transactions on Nuclear Science*, December 2003, Submitted.

[18] 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 adapative computing", *VLSI Design*, vol. 10, no. 3, pp. 265–279, 2000.