Hardness By Design Techniques for Field Programmable Gate Arrays

Michael Wirthlin†, Nathan Rollins†, Michael Caffrey‡, and Paul Graham‡
†Department of Electrical and Computer Engineering
Brigham Young University, Provo, UT
‡Los Alamos National Laboratory, Los Alamos, NM

Abstract

FPGAs are an attractive alternative for many space-based computing operations. While radiation hardened FPGAs are available, SRAM-based FPGAs are susceptible to Single-Event Upsets (SEUs). Several FPGA design hardening techniques are investigated to improve the reliability of FPGA designs operating in a radiation environment. The improved design reliability provided by these techniques are measured using a single-event upset simulation environment.

I. Introduction

There is increasing interest in the use of field programmable gate arrays (FPGAs) for many space-based computing operations[1]. The circuitry of high-density commercial FPGAs can be configured to perform application-specific operations on spacecraft sensor data. Further, the hardware configuration of the FPGA can be reprogrammed for varying mission objectives or multiple system instruments.

While FPGAs offer several benefits for space-based electronics, they are sensitive to single event upsets[2]. Single-event upsets in the FPGA affect the user design flip-flops, the FPGA configuration bitstream, and any hidden FPGA registers, latches, or internal state. Configuration bitstream upsets are especially important because such upsets affect both the state and operation of the design. Configuration upsets may perturb the routing resources and logic functions in a way that *changes* the operation of the circuit.

The purpose of this work is to investigate design *hardening* techniques for improving the reliability of FPGA designs in the presence of configuration memory upsets. Like many design hardening approaches, the techniques described in this paper involve the addition of redundant hardware. By addition appropriate redundant hardware and voting circuitry, upsets within the FPGA configuration can be tolerated.

This paper will begin with an overview of radiation effects on FPGAs. Next, several techniques for improving the reliability of FPGA circuits will be presented. Each of these techniques involves the use of redundant circuitry and will consume additional hardware resources. The hardware costs of each technique will be identified and contrasted. The effectiveness of each mitigation technique will be evaluated by using a SEU simulation environment completed in previous work. The paper will conclude by summarizing the design hardening techniques and discussing future work.

II. Radiation Effects for SRAM-based FPGAs

Electronic circuits operating outside the earth's atmosphere are exposed to levels of radiation much higher than the radiation level found on Earth. Users of FPGA devices must consider these radiation effects before including an FPGA within a space application. To address the need for radiation tolerant FPGAs, Xilinx has introduced the QPro high-reliability family of FPGA devices[3]. This set of radiation-tested FPGAs includes both the Virtex and Virtex-II FPGA families.

The Virtex QPro FPGA is manufactured on a thin-epitaxial 0.22 μmm CMOS process and based on the commercially available Virtex FPGA architecture. The XQVR high-reliability Virtex FPGA has been tested extensively for radiation tolerance and has been shown to tolerate a total dose in range of 80 to 100 krads(Si). This total dose tolerance is acceptable for many space applications. In addition, this device is immune to latch-up up to an LET of 125 MeV=cm^2/mg.

A. FPGA Sensitive Cross Section

While the XQVR Virtex FPGA is immune to latch-up and has an acceptable total-dose tolerance, it is sensitive

to single-event upsets (SEUs). Single-event upsets are changes in the state of a flip-flop, latch, or register caused by high-energy protons or heavy ions. Devices that contain dense arrays of memory cells are especially sensitive to such SEUs due to the large amount of memory state within a relatively small amount of circuit area. Much like SRAM and DRAM, SRAM-based FPGAs contain large amounts of memory cells within a device and are especially sensitive to radiation induced SEUs.

As an example, the Virtex V1000 FPGA contains almost six-million bits of internal state. As suggested in Table I, this relatively large amount of internal state is used for several important purposes.

Memory Type	# Bits	%
Configuration	5,810,048	97.4%
Block RAM	131,072	2.2%
User Flip-Flops	26,112	0.4%
Total	5,967,232	100%

TABLE I. Memory Bits Within the Virtex XCV1000

1) User Flip-Flops: An important architectural component of all FPGAs are user programmable flip-flops. User designs exploit these flip-flops to implement common sequential logic circuits such as state machines, counters, and registers. User flip-flops in most digital technologies are susceptible to radiation-induced single-event upsets. Many digital circuits operating in a radiation environment exploit redundancy (i.e. multiple flip-flops) to mitigate against such single-event effects [4].

2) User Memory: Modern FPGAs provide blocks of internal memory larger than the typical look-up table. This block memory is used for traditional random access memory functions such as data storage, buffering, FIFO, etc. The Virtex family includes a set of internal dual-ported BlockRAM memories that each provide 4096-bits of randomly accessible memory. With 32 BlockRAM memories, the Virtex V1000 FPGA offers 131,072 bits of internal memory. Dense static memory such as the BlockRAM is especially susceptible to radiation-induced SEUs. Well-known error-correction coding techniques are often used within a user design to detect and correct such upsets[**?**].

3) Configuration Memory: As shown in Table I, 97% of the known memory cells within the Virtex V1000 device are devoted to configuration memory. These memory cells define the operation of the configurable logic blocks, routing resources, input/output blocks, and other programmable FPGA resources. Like other static memory cells, configuration memory is susceptible to single-event upsets. Upsets within the configuration memory are especially troublesome as they may *change* the operation of the circuit. Any spacecraft that utilizes SRAM-based FPGAs must anticipate and mitigate against upsets within the device configuration memory.

III. Configuration Memory Upset Simulator

The presence of large amounts of configuration memory within an FPGA poses unique challenges to users of SRAM-based FPGAs within a hostile radiation environment. Systems that use these FPGAs must understand the impact of configuration upsets and apply appropriate mitigation or recovery techniques. To learn more about the impact of configuration upsets, a simulator was created to study the effects of upsets within the configuration memory of a Virtex V1000 FPGA[5], [6].

This simulator models the impact of SEU induced upsets within the configuration memory by *artificially* inserting faulty bits within the configuration bitstream of the FPGA. This configuration upset simulator is based on the architecture shown in Figure 1. Two FPGAs are configured with equivalent designs and are run with an identical clock and test vectors. Under normal operating circumstances, the two FPGAs produce identical results. During SEU simulation, the FPGA under test (PE2) is subjected to artificial modifications in the configuration bitstream. A third FPGA (PE0) monitor the behavior of the FPGA under test by comparing its output with that of the golden FPGA design (PE1).

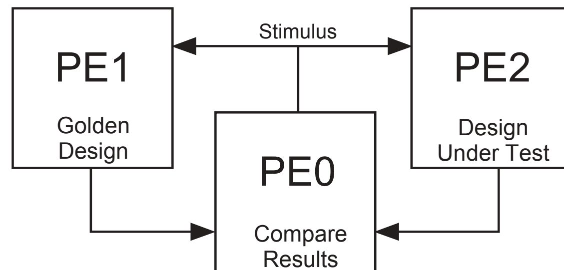

Fig. 1. Simulator Architecture.

The purpose of the simulator is to identify configuration upsets that cause design failures. During the course of a typical simulation, *every* bit within the configuration memory is independently toggled within the FPGA under test. As configuration bits are artificially upset, the operation of internal FPGA resources changes and may negatively impact the behavior of the design operating within the FPGA. The comparison FPGA monitors the operation of this FPGA design and identifies discrepancies between the outputs of the design under test and the golden design. If design failures are identified, the corresponding configuration bit is recorded and the bitstream is repaired.

The simulator was used to evaluate the impact of configuration upsets on a variety of designs. Because

each FPGA design uses different FPGA resources, the impact of configuration upsets on different designs will vary significantly. Results from the simulator suggest that larger designs are more sensitive to configuration upsets than smaller designs using fewer FPGA resources. Further, designs that exploit FPGA-specific features to increase the density of logic resources are more sensitive than other generic logic implementations. Details of the simulator results can be found in the following publications [5], [6].

IV. Design Hardening Techniques

Several techniques have been proposed and tested for mitigating SEUs in FPGAs. Many techniques use hardware redundancy to reduce the probability of failure[7]. By replicating the desired circuitry and comparing the results, faults in the configuration can be detected and reported. Previous efforts have reported on the successful use of redundancy to improve the reliability of FPGA circuits[8], [4]. These techniques use triple-modular redundancy (TMR) to replicate the design within the FPGA. Upsets occurring within the configuration memory of a TMR design will only affect one of the three copies of the circuit module. Through appropriate voting, the remaining two copies of the circuit module will properly generate the correct value.

Other techniques rely on temporal redundancy to mitigate against transient faults [9], [10]. These techniques reuse hardware resources in time to identify temporary faults in the hardware or circuit state. While most of these techniques do not mitigate against permanent faults, they are a relatively inexpensive way of identifying permanent faults in the design.

A unique form of SEU mitigation for FPGAs is the use of frequent device configuration to "clean" the device bitstream. Called configuration "scrubbing", this technique repeatedly configures the device configuration memory to remove any upsets caused by radiation[11]. By continually scrubbing the configuration memory, maximum limits can be placed on the time in which a configuration upset is present within a device. Mitigation techniques used with scrubbing are designed to tolerate SEUs within the configuration memory for this maximum time limit.

V. Counter Design Tests

An important part of this work is the validation of a variety of techniques used to improve the reliability of FPGA circuits. The improvements in reliability provided by these techniques will be tested using the artificial SEU simulator developed at BYU[5]. This simulation environment artificially inserts upsets within the configuration memory and identifies design failures caused by configuration upsets. Results from this simulation environment will be presented for several TMR designs.

To test the effectiveness of using redundant hardware, several redundant hardware techniques were applied to a simple 8-bit counter. Each technique provides various amounts of redundant hardware to reduce the sensitivity of the design to configuration upsets. The results of these techniques are summarized in Table II.

A. Baseline Counter

The baseline design used in this study is a simple 8-bit counter with no hardware redundancy. As shown in Figure 2, this simple design includes a global clock buffer (`bufg`), an 8-bit counter, and an output buffer (`obuf`) for each of the 8-bits of the counter. This counter exploits the internal carry-chain of the Virtex FPGA to fit a single bit of the counter within a single 4-input look-up table of the device. The total size of this counter is 8 look-up tables (LUTs) or 4 logic "slices".

Fig. 2. 8-Bit Counter Without Redundancy

The SEU simulator was applied to this design to test the sensitivity of the counter to configuration upsets. With no hardware redundancy, every hardware resource used by this design is sensitive to configuration upsets. The SEU simulation on this design found that the design is sensitive to 389 configuration bits. While this is an extremely small fraction of the total number of configuration bits, the normalized sensitivity is 48.6 failures per LUT.

It is interesting to note that the configuration sensitivity depends upon the placement of the 8-bit counter. Three different placements of this 8-bit counter were tested to demonstrate the impact of placement on configuration sensitivity. Figure 3 demonstrates the location of three different placements of the 8-bit counter. Example #1 is placed on the left-side of the FPGA close to the IO outputs of the device. This placement resulted in the 389 sensitive configuration bits described earlier. Example #2 is placed on the bottom right corner of the FPGA and is slightly offset from the IO outputs. This placement resulted in 408 sensitive configuration bits. Example #3 is placed in the center of the chip near the top. The increased routing needed to connect to the IOBs increases the configuration sensitivity to 781.

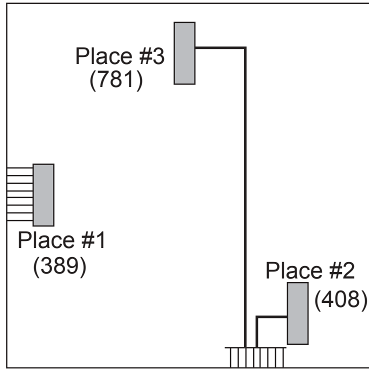

Fig. 3. 8-Bit Counter Placement Sensitivity.

B. Simple TMR Counter

The first approach for SEU mitigation is to add triple-modular redundancy (TMR) for the 8-bit counter. Three copies of the 8-bit counter are created with one voter as shown in Figure 4. The added hardware required for the redundant counters and voter increase the resource requirements to 32 LUTs (4 times the size of the non-redundant counter). Each bit of the voter consumes 1 LUT to perform the "best of three" voting circuit.

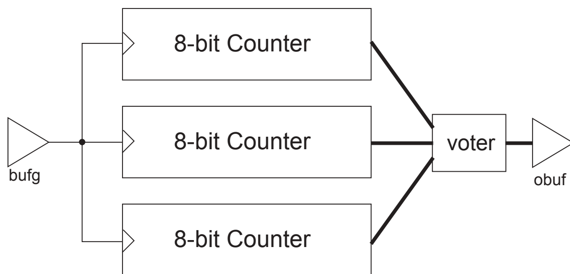

Fig. 4. 8-Bit Counter With TMR and One Voter.

By adding redundant counters and a voter to select the proper count value, this circuit should tolerate single-bit failures in any of the three counter circuits. The use of a single voter, however, results in a single-event susceptibility in the voting logic. The SEU simulation for this design identified 418 sensitive configuration bits or 13.1 failures per LUT for this 8-bit TMR design.

While the normalized failure rate of this design is lower than the non-redundant counter, this technique actually *increases* the SEU sensitivity of the counter. This surprising result can be justified by comparing the sensitive design resources in both approaches. For the non-redundant counter, all 8 LUTs of FPGA resources are sensitive to configuration upsets. For the single voter TMR design of Figure 4, only the single voter is sensitive to configuration upsets. Since the size of the voter is the same as the original 8-bit counter, no improvement in design sensitivity is achieved. A single voter should only be used if the size of the non-redundant design is significantly larger than the voter circuit.

C. TMR Counter With Three Voters

To reduce the configuration sensitivity of this counter, a triple redundant counter is created with *three* voters as shown in Figure 5. The added hardware resources required for the three voters increases the counter size to 48 (6 times the size of the non-redundant counter).

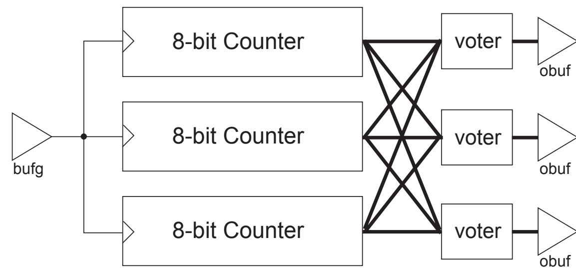

Fig. 5. TMR Counter With Three Voters.

Adding three voters to the redundant counter significantly improves the configuration sensitivity of the design. The SEU simulation for this design identified only 29 sensitive configuration bits for the redundant counter. The normalized sensitivity reduces to 0.6 sensitive configuration bits per LUT. This redundant counter is 13 times less sensitive than the original non-redundant counter.

D. TMR Counter With Three Voters and Clocks

The final design mitigation strategy involves the use of three unique clock buffers as shown in Figure 6. The use of a custom global clock buffer for each counter insures that failures in the global clock buffer do not disable all three of the independently executing counters. Simulation results for this design identify only 27 sensitive configuration bits (2 less than the TMR with one clock). The use of redundant clock buffers is 14 times less sensitive than the original non-redundant counter.

VI. Feedback TMR

The previous techniques for using redundancy have been shown to significantly reduce the sensitivity of an 8-bit counter to upsets within the configuration memory. Each of these techniques, however, suffer in their inability to restore the count sequence once the counter has been repaired. Ideally, the redundancy technique used within the

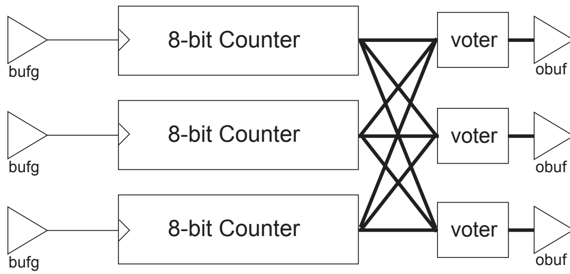

Fig. 6. TMR Counter With Three Voters and Clocks.

FPGA will automatically restore the proper count sequence when the damaged counter has been repaired.

This problem can be demonstrated with the sequence shown in Figure 7. In this example, a configuration fault occurs that forces the clock enable to counter #3 into a stuck-at-0 condition. Because of this fault, the counter does not increment and remains in the same count state until the counter is repaired. Once the counter has been repaired, it continues its count sequence from the state in which it was stuck. Although repaired and operating properly, this counter is out of sequence with the other two counters. While the TMR voter circuitry will properly ignore the incorrect count value, any additional faults in the other counters will cause the redundant circuitry to fail.

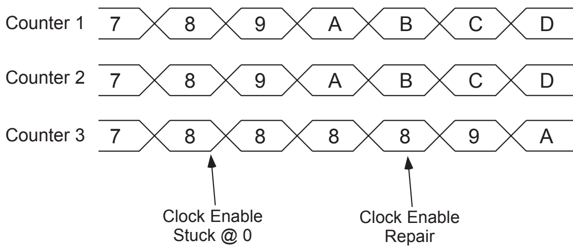

Fig. 7. Counter Failure and Repair Sequence.

This counter example can be self-restoring by putting the voting circuitry within the *feedback* path of the counter. As described by Xilinx Application Note 197[7], this technique insures that the proper state value is restored once the sequential circuit has been repaired. The counter example can be made to be self-restoring by breaking the feedback path between the counter registers and inserting the appropriate voting circuitry.

Figure 8 demonstrates this technique with the 8-bit counter example. The 8-bit counters are replaced by 8-bit sequential incrementers that do not contain feedback. Each incrementer has an 8-bit input and produces an registered output of the input signal plus one. Three voter circuits are added to determine the correct value of the three incrementers. These voted results are fed back to the incrementers to insure that *each* incrementer receives the correct input value.

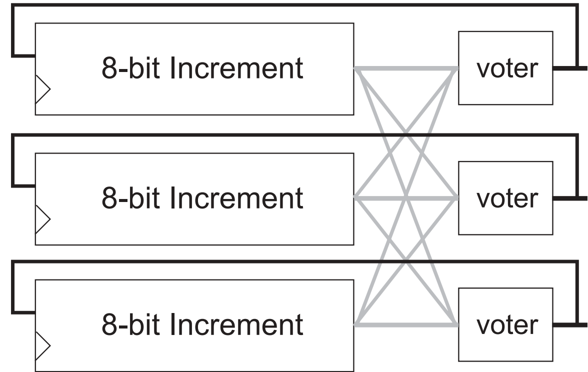

Fig. 8. Redundant Counter with TMR in Feedback Path.

Using the voter within the circuit feedback insures that the proper input value is provided to all the incrementers no matter where the fault lies. The advantage of this technique can be seen using the counter failure sequence in Figure 9. As described in the earlier example, counter #3 experiences a stuck-at-0 fault on its clock enable input. While this fault is present, the incrementer retains the same value and falls out of sequence with the other incrementers. However, the input to this incrementer constantly provides the proper count sequence. The voter circuit that drives this input properly ignores the faulty value to insure the proper value is loaded as soon as the circuit is repaired. Once the circuit is repaired, the proper value is loaded and properly synchronized with the other incrementers.

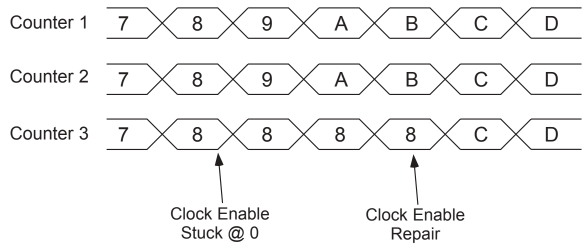

Fig. 9. Counter Failure Sequence Using Feedback.

The modified TMR circuit with feedback was created and tested within the SEU simulator. This circuit consumed the same amount of hardware as the 3-voter TMR circuit (48 LUTs) but operates slower due to the insertion of voters within the feedback loop. With a single global clock buffer, this circuit contains only 20 sensitive configuration bits (0.5 failures/LUT). Using three independent global clock buffers, this circuit contains only 5 configuration failures.

Design	LUTs	Overhead	Failures	Failures per LUT
No Redundancy	8	1.0	389	48.6
TMR - 1 voter	32	4.0	418	13.1
TMR - 3 voters	48	6.0	29	0.60
TMR - 3 voters, 3 clocks	48	6.0	27	0.56
Feedback TMR	48	6.0	24	0.50
Feedback TMR, 3 clocks	48	6.0	5	0.10

TABLE II. Configuration Sensitivity of SEU Mitigation Approaches.

VII. Summary

Several techniques were presented for reducing the sensitivity of FPGA circuit designs to upsets within the device configuration memory. These results, summarized in Table II, suggest that significant improvements can be made to circuit reliability by adding appropriate redundant hardware. The reliability of the 8-bit counter example was improved by a factor of 78 by adding triple redundant hardware within the feedback path of the counter and using three independent clock domains. Although costly, these techniques demonstrate that significant improvements can be made in the ability of FPGA circuits to tolerate radiation induced single event upsets.

The work presented in this paper represents initial studies into the success of design hardening techniques. Future efforts will continue to study these and other techniques on a variety of circuit structures. Additional effort will be aimed at completely eliminating the sensitivity of FPGA designs on upsets within the configuration memory. As these and other techniques are verified, tools will be developed for automatically inserting redundancy within a user design.

References

[1] 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. CSREA Press, June 2002, pp. 49–53.

[2] 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.

[3] P. Brinkley and C. Carmichael, "SEU mitigation design techniques for the XQR4000XL," Xilinx Corporation, Tech. Rep., March 15 2000, xAPP181 (v1.0).

[4] 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 Sysemts (RADECS 2001)*, 2001.

[5] 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. CSREA Press, June 2002, pp. 68–73.

[6] 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. IEEE Computer Society Press, April 2003, to Be Published.

[7] C. Carmichael, "Triple module redundancy design techniques for Virtex FPGAs," Xilinx Corporation, Tech. Rep., November 1, 2001, xAPP197 (v1.0).

[8] 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 *4th Annual Conference on Military and Aerospace Programmable Logic Devices (MAPLD)*, 2000, p. P30.

[9] 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.

[10] F. Lima, L. Carro, and R. Reis, "Reducing pin and area overhead in fault-tolerant FPGA-based desings," in *Proceedings of the 11th Annual International Symposium on Field-Programmable Gate Arrays (FPGA 2003)*, 2003, to Be Published.

[11] C. Carmichael, M. Caffrey, and A. Salazar, "Correcting single-event upsets through Virtex partial configuration," Xilinx Corporation, Tech. Rep., June 1, 2000, xAPP216 (v1.0).