Dynamic SEU Sensitivity of Designs on Two 28-nm SRAM-Based FPGA Architectures

Andrew M. Keller, *Student Member, IEEE*, Timothy A. Whiting, *Student Member, IEEE*, Kenneth B. Sawyer, and Michael J. Wirthlin, *Senior Member, IEEE*

***Abstract*— Two field-programmable gate array (FPGA) designs are tested for dynamic single event upset (SEU) sensitivity on two different 28-nm static random access memory-based FPGAs—an Intel Stratix V and a Xilinx Kintex 7 FPGA. These designs were tested in both a conventional unmitigated version and a version to tolerate SEUs with feedback triple modular redundancy (TMR). The unmitigated design sensitivity and the low-level device sensitivity were found to be similar between the devices through neutron radiation testing. Results also show that feedback TMR and configuration scrubbing benefit both designs on both FPGAs. While TMR is helpful, the benefit of TMR depends on the design structure and the device architecture. TMR and scrubbing reduced dynamic SEU sensitivity by a factor of 4–54×.**

***Index Terms*— Fault tolerace, field programmable gate arrays, neutron radiation effects, redundancy, static random access memory (SRAM) cells.**

I. Introduction

SRAM-BASED field-programmable gate arrays (FPGAs) provide large amounts of reprogrammable resources for use in a wide variety of applications. In addition to logic resources (e.g., look-up tables (LUTs) and flip-flops), FPGAs offer high-speed serial I/O, DSP units, and block memories. These devices are used in high-throughput, low-latency data and control processing applications. They can be remotely reprogrammed an unlimited number of times. Their rich pool of resources and flexibility encourages their use in terrestrial and space environments.

Radiation in space and terrestrial environments can cause designs operating on static random access memory (SRAM)-based FPGAs to malfunction [1]–[3]. Single event upsets (SEUs) in SRAM-based FPGAs can cause functional failures in active designs by corrupting the state and circuit configuration [4]. Functional failures occur when the behavior of a design deviates from its intended operation. Before using an SRAM-based FPGA in space environments, the risk of radiation-induced failure must be addressed. Some applications may require the use of SEU mitigation techniques to improve functional reliability.

Many SEU mitigation techniques have been developed to make FPGA designs more robust in harsh radiation environments. A well-known SEU mitigation technique is triple modular redundancy (TMR), which has been applied to SRAM-based FPGA designs to reduce the risk of radiation-induced failure [5]–[7]. In TMR, three redundant modules and a voting system are used to mask errors that would normally result in functional failures. TMR has been shown to greatly improve functional reliability but this improvement comes at the cost of greater power consumption, resources utilization, and slower timing. Generally speaking, TMR cannot mask simultaneous errors in multiple redundant modules. Configuration scrubbing is often combined with TMR, because it prevents SEU accumulation that would compromise TMR [8]. To preserve the masking effects of TMR, any corrupted state within a module must also be resynchronized with the other redundant modules if it persists in the design (i.e., does not get flushed). This can be done by placing synchronization voters in the feedback paths of the circuit [9]. This is known as feedback TMR or distributed TMR. It is the form of TMR used in this paper. When combined with configuration scrubbing, feedback TMR has demonstrated significant improvements in SRAM-based FPGA design reliability [10].

The risk of functional failure caused by SEUs can be evaluated through dynamic testing and is discussed in terms of dynamic SEU sensitivity. Dynamic testing observes the behavior of an active design while it is undergoing fault injection or radiation testing [11]. If the behavior of a design deviates from its intended operation (e.g., silent data corruption, stalling/hanging, and jumping operational states incorrectly), it is considered as a functional failure. Dynamic SEU sensitivity reflects the probability of a functional failure given that a random SEU has occurred. Dynamic SEU sensitivity is measured in random fault injection as the percentage of faults that cause a functional failure, and it is measured in neutron radiation testing as the neutron cross section of a functional failure event. Many experiments have compared the mitigated and unmitigated dynamic SEU sensitivity of a design (e.g., with and without TMR applied to it) to determine the effectiveness of an SEU mitigation technique [5], [6], [10], [12].

This experiment compares the dynamic SEU sensitivity of the designs operating on two 28-nm FPGAs with different architectures to examine the impact of architecture on reliability. Two designs were selected as reliability benchmark circuits. The dynamic SEU sensitivity of both designs was measured with and without TMR on both FPGAs thorough neutron radiation testing. The neutron cross section of a single configuration random access memory (CRAM) bit was also measured for each FPGA. Similarities were found in

Manuscript received August 4, 2017; revised October 3, 2017 and October 25, 2017; accepted October 26, 2017. Date of publication November 10, 2017; date of current version January 17, 2018. This work was supported in part by the I/UCRC Program of the National Science Foundation under Grant 1265957 and in part by the Los Alamos Neutron Science Center under Grant NS-2016-7268-F.

The authors are with the NSF Center for High-Performance Reconfigurable Computing, Brigham Young University, Provo, UT 84602 USA (e-mail: andrewmkeller@byu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNS.2017.2772288

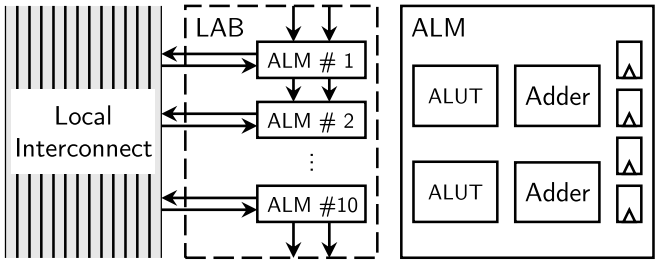

Fig. 1. Intel Stratix V architecture.

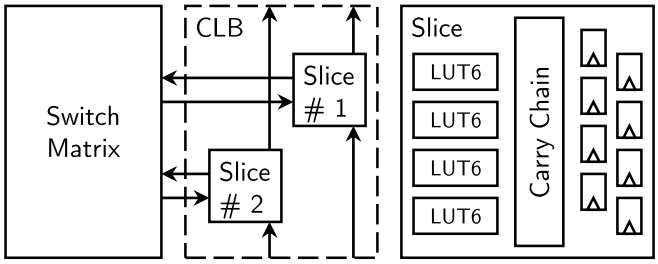

Fig. 2. Xilinx Kintex 7 architecture.

the unmitigated dynamic SEU sensitivity of designs across architectures and in the low-level single-bit cross section between the two devices. TMR and configuration scrubbing improved the design reliability in all cases. The benefit of TMR was the design structure and the device architecture dependent with an improvement ratio of 4–54×.

II. 28-nm FPGA Architectures

The goal of this experiment is to compare the dynamic SEU sensitivity of designs across FPGA architectures. To accomplish this goal, FPGAs with similar process technologies were selected for testing. The first is an Intel Stratix V 5SGXEA7 FPGA and the second is a Xilinx Kintex 7 325T FPGA. Both the Stratix V FPGA and the Kintex 7 FPGA use a 28-nm planar CMOS process technology [13], [14]. The FPGAs were selected from different vendors to promote greater variation in architecture. Devices with similar process technology were selected to focus the comparison of dynamic SEU sensitivity on differences in architecture rather than process technology. Although the density of each device is unknown (i.e., transistors per mm-squared), it is important to note that differences in density would impact results. Having higher density increases the likelihood of an SEU, which in turn increases the cross section of functional failure.

While the architectures between the two FPGAs are different, their architectures do have some aspects in common. Both FPGAs implement an island-style architecture [15]. This style of architecture arranges logic blocks in a 2-D matrix with each block surrounded by interconnects. Both implement column-based resource layouts where columns in the device consist of a single resource type for the most part [13], [16]. The way that resources are implemented and interconnected is different.

In the Stratix V architecture [17], the basic grouping of combinational resources and registers is called an adaptive logic module (ALM). Each consists of two adaptive LUTs (ALUTs), two adders, and four registers. Eight inputs are shared between the two ALUTs. A shared arithmetic chain and a carry chain pass through each ALM to its vertically adjacent neighbors. ALMs are grouped in sets of ten in logic array blocks, which are connected to their own local interconnect resources. This organization with its respective interconnects is represented in Fig. 1. There are a total of 234 720 ALMs in the Stratix V 5SGXEA7 FPGA used in this experiment.

In the Kintex 7 architecture [16], the basic grouping of combinational resources and registers is called a slice. Each slice contains four six-input LUTs, carry chain logic, and eight registers. Slices are grouped in sets of two into configuration logic blocks, which have their own associated switch matrix for interconnects. This is represented in Fig. 2. There are a total of 50 950 slices in the Kintex 7 325T FPGA.

TABLE I
Device Capacity

Stratix V	Logic Elements	622,000
	CRAM Cells	98,681,196
Kintex 7	Logic Cells	326,080
	CRAM Cells	72,778,176

The FPGAs differ in logic capacity and number of configuration memory (CRAM) cells. Table I shows the device capacity of each FPGA in terms of logic elements (LEs) for the Stratix V and logic cells (LCs) for the Kintex 7. Table I also includes the total number of configuration memory (CRAM) cells for each FPGA. LEs and LCs are logically equivalent to a four-input LUT and flip-flop pair [16], [18]. The total number of LEs and LCs for each device was taken from their respective data sheets [13], [14]. The number of CRAM bits was approximated using reports from the design tools.

Because this is an application-level comparison of dynamic SEU sensitivity (i.e., the same design on both FPGAs), the difference in logic capacity and number of CRAM bits between the FPGAs should have minimal impact on results. The Stratix V has approximately 1.9× more logic capacity than the Kintex 7, and it has approximately 1.4× more CRAM cells than the Kintex 7. Although the Stratix V FPGA is larger by these metrics, a comparison of dynamic SEU sensitivity of designs implemented on the Kintex 7 FPGA focused on differences in architecture should still be feasible between these devices.

To facilitate the construction of this experiment, ready-made FPGA development boards were used for each targeted FPGA (see Fig. 3). The Terasic DE5-Net development board [19] was used to test the Stratix V 5SGXEA7 FPGA, and the Xilinx KC705 development board [20] was used to test the Kintex 7 325T FPGA. Both of these boards offer a variety of I/O interfaces, including joint test action group (JTAG). This experiment uses JTAG as the primary interface with the boards. With JTAG accessibility, these boards provide a convenient platform for testing.

III. Test Designs

Comparing the dynamic SEU sensitivity of the same design implemented on different architectures allows reliability to be compared across architectures at an application level. In this way, an FPGA design acts as a reliability benchmark. Like performance benchmarks for CPUs, reliability benchmark circuits can be used a means of reliability comparison between FPGA architectures and SEU mitigation techniques [12]. Designs

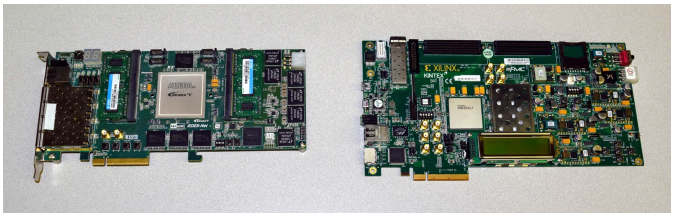

Fig. 3. Development boards: Terasic DE5-Net (left) and Xilinx KC705 (right).

TABLE II
SINGLE DESIGN INSTANCE RESOURCE UTILIZATION (% OF DEVICE)

Stratix V	ALUTs	Registers	ALMs
B13	54	56	33 (.014%)
B13 with TMR	282	168	158 (.067%)
AES	17,296	6,720	14,755 (6.3%)
AES with TMR	51,888	20,160	44,317 (18.9%)
Kintex 7	**LUTs**	**Registers**	**Slices**
B13	34	52	13 (.026%)
B13 with TMR	221	156	82 (.16%)
AES	10,688	6,000	3,061 (6.0%)
AES with TMR	34,052	18,000	9,314 (18.3%)

selected as reliability benchmark circuits will be implemented in different FPGA architectures but will perform the same task. This section introduces the test designs selected and discusses their resource utilization on the targeted FPGAs.

Two designs were selected as reliability benchmark circuits. The first design is a small finite-state machine called the B13 from the ITC'99 benchmark suite [21]. The second is a 128-b advanced encyption standard (AES) cryptography core [22]. Both of these designs are the representative of real-world applications. Both are available in hardware description language (HDL), making them easy to port to the target FPGAs.

The B13 design is one of several benchmark circuits in the ITC'99 benchmark suite. This circuit is used to interface with a weather sensor. This particular circuit has been used in other FPGA reliability experiments [12], [23]. As shown in Table II, a single design instance only consumes a small portion of logic resources. Test vectors are provided to keep the design active.

The AES design used for this experiment is a 128-b cryptography IP core from OpenCores.org [22]. This core implements a common cryptography standard that allows for the encryption and decryption of data. The core is fully pipelined and produces valid data every clock cycle once the pipeline is filled. In this experiment, random data and keys are fed into the core to keep the design active. This design was selected for testing, because a single instance uses many more resources than an instance of the B13 (see Table II). Synthesis attributes were added to force lookup tables in the IP core to be implemented in LUTs rather than block memories. It is interesting to note that, unlike the B13, there are no feedback paths within the AES design (i.e., all logics are feedforward).

Fig. 4 shows the TMR implementation flow used in this experiment. The flow begins with the HDL of the designs. They are ported to each FPGA through logic synthesis and technology mapping. This was performed using Quartus Prime 16.0 Standard Edition for the Stratix V and Vivado v2016.2 for the Kintex 7. The result is a structural netlist for each design

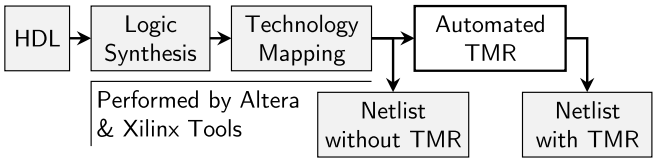

Fig. 4. TMR flow.

on each FPGA. An automated TMR tool takes the original netlist, applies feedback TMR [9] to the netlist, and generates a new TMR netlist. The automated TMR tool used in this experiment is part of the BYU EDIF Tools [24]. Through this process, netlists of each design with and without TMR for both FPGAs are generated.

Table II shows the resource utilization of a single design instance with and without TMR implemented in both FPGAs. The overhead of TMR is reflected in the increase of resource utilization of the TMR'd design versions. Three times as many registers are used in all cases. Combinational resources increase 3–3.2× in the TMR'd AES design and 5.2–6.5× in the TMR'd B13 design. The large TMR overhead in the B13 design is due to having many feedback paths in a small amount of combinational logic with voters placed in every feedback path. Comparison of resources utilization between devices is not made due to differences in architecture.

IV. TEST LOGIC INFRASTRUCTURE

An on-chip test logic infrastructure was created to facilitate testing. This test logic infrastructure allows multiple design instances to be actively tested in parallel, which increases the rate of data acquisition for fault injection and radiation testing (i.e., more design instances and more frequent failure events). The test logic infrastructure is implemented on the same device as the design instances. The test logic infrastructure is protected with TMR to minimize its impact on dynamic SEU sensitivity results. Design netlist instances in the test logic infrastructure are all of the same design and version. The same TMR'd test logic infrastructure is used to test the unmitigated version of a design, and it is used to test the mitigated (i.e., with TMR and configuration scrubbing) version of the same design.

Fig. 5 shows the general layout of the test logic infrastructure for any design. The infrastructure consists of control logic, design instances, and failure detection logic. The control logic provides the design instances with stimulus to ensure that they remain active for dynamic testing. It can also reset the design and record failure events. The design instances can be configured in a number of different ways. Failure detection logic compares the lockstep outputs of design instances against themselves, and a golden output vector if provided, on a clock-by-clock basis. If any discrepancy is found, a function failure is reported.

Different test logic configurations were used for each test design. They are shown in Fig. 6. The B13 test logic configuration instances the B13 512 times and drives a set of input stimulus to all instances in parallel. Their outputs are compared against each other and a set of golden output vectors to detect functional failures. The AES test logic configuration instances the AES 4 times between two chains. After the

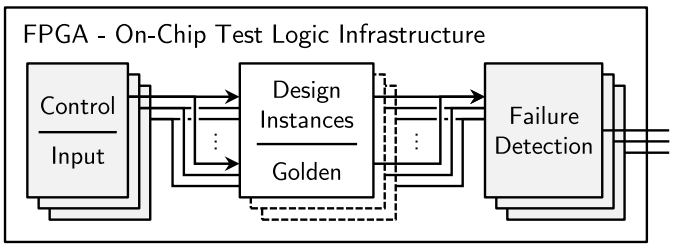

Fig. 5. Generalized on-chip test logic infrastructure.

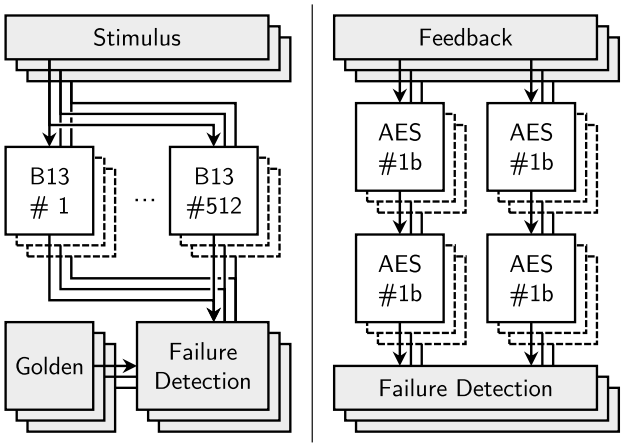

Fig. 6. Test logic configurations: B13 (left) and AES (right).

TABLE III
OVERALL RESOURCE UTILIZATION (% OF DEVICE)

Stratix V	ALUTs	Registers	ALMs
B13	53,096	34,559	30,255 (13%)
B13 with TMR	180,072	91,903	96,926 (41%)
AES	75,452	33,711	64,288 (27%)
AES with TMR	214,844	87,983	183,553 (78%)
Kintex 7	**LUTs**	**Registers**	**Slices**
B13	24,807	27,025	8,757 (17%)
B13 with TMR	137,350	80,273	37,037 (73%)
AES	44,860	25,787	13,030 (26%)
AES with TMR	139,124	73,787	37,821 (74%)

control logic fills the pipeline of the chains, their output is feed back into their input to keep them actively processing pseudorandom data. The outputs of the two chains should be identical; discrepancies are reported as functional failures.

The overall resource utilization of the test logic infrastructure and design instances is shown in Table III. All of these resources are exposed to fault injection and radiation testing. Fewer than three times as many registers are used in the TMR version of the design in all cases. The combinational logic overhead of TMR is less than that of a single design instance as well. This is due in part to the shared test logic infrastructure found in both versions of the design.

Because the test logic infrastructure is placed on the chip with the designs being tested, it is possible for the test logic infrastructure to become corrupted while testing. If the test logic gets corrupted it could result in either a false positive or a false negative. A false positive occurs when the test logic reports a functional failure that did not actually happen. A false negative occurs when the test logic fails to report a functional failure that actually happened. To minimize the impact of test logic corruption on results, TMR has been applied to the test logic with redundancy added to the functional failure detection logic. In [25], redundant on-chip detection logic demonstrated a 99.95%–99.99% accuracy in detecting functional errors.

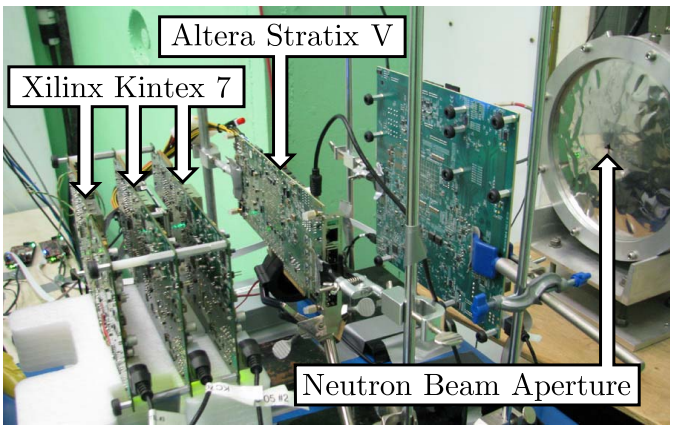

Fig. 7. Neutron beam test setup.

V. RADIATION TESTING

Neutron radiation testing was conducted at the Los Alamos Neutron Science Center (LANSCE) in December 2016. This facility provides a wide spectrum spallation neutron beam source, which is commonly used to measure and report terrestrial neutron soft errors in semiconductor devices [26]. Radiation testing is the commonly accepted standard for validating SEU mitigation techniques and evaluating dynamic SEU sensitivity [27].

In preparation for radiation testing, fault injection was performed. Hardware-based fault emulation techniques were used to mimic SEUs in the configuration memory and observe the response. The fault injection algorithm used in this experiment follows the same basic flow presented in [28]. A random fault injection campaign was used where faults are injected into random bits throughout the device. This approach is similar to statistical fault injection [29] and was used to approximate the dynamic SEU sensitivity of the test designs prior to radiation testing. The detailed fault injection results are available in [30].

While fault injection provides useful information, radiation testing provides better test coverage (e.g., able to upset states that cannot be fault injected, test single event transients, and multibit upsets) [28]. The ratio of the upset rate to scrub rate is different in fault injection and radiation testing. In fault injection, the device is scrubbed before a new fault is introduced. In radiation testing, many faults can occur before a scrub cycle completes. Exaggerated SEU accumulation compared with normal operating conditions can increase the cross section. To further investigate the performance of TMR across the two architectures, radiation testing was conducted.

Fig. 7 shows the Intel Stratix V DE5-Net and three Xilinx Kintex 7 KC705 development boards mounted perpendicular to the neutron beam aperture with the FPGAs aligned to the flight path. Distance from the beam aperture to each board was carefully measured so that the distance degradation could be taken into account. For this experiment, a two-inch collimator was used. For automated logging and testing, the Stratix V FPGA was connected to a host computer via universal serial

TABLE IV
NEUTRON RADIATION RESULTS

Design		ITC'99 Benchmark B13		128-bit AES IP Core	
Description		Unmitigated	TMR w/Scrubbing	Unmitigated	TMR w/Scrubbing
Intel Stratix V	Total Fluence	6.24×10^{10}	1.06×10^{11}	7.88×10^{9}	1.13×10^{11}
	Observed Failures	661	102	365	291
	Cross Section (cm^2) (95% Conf. Interval)	1.06×10^{-8} (9.79×10^{-9}; 1.14×10^{-8})	9.62×10^{-10} (7.75×10^{-10}; 1.15×10^{-9})	4.63×10^{-8} (4.16×10^{-8}; 5.11×10^{-8})	2.58×10^{-9} (2.29×10^{-9}; 2.88×10^{-9})
	Improvement (95% Conf. Interval)	1.00× (0.86×; 1.17×)	11.02× (8.53×; 14.72×)	1.00× (0.81×; 1.23×)	17.93× (14.43×; 22.34×)
Xilinx Kintex 7	Total Fluence	1.08×10^{11}	2.83×10^{11}	4.44×10^{10}	7.84×10^{10}
	Observed Failures	838	41	1,621	720
	Cross Section (cm^2) (95% Conf. Interval)	7.79×10^{-9} (7.26×10^{-9}; 8.32×10^{-9})	1.45×10^{-10} (1.00×10^{-10}; 1.89×10^{-10})	3.65×10^{-8} (3.48×10^{-8}; 3.83×10^{-8})	9.19×10^{-9} (8.52×10^{-9}; 9.86×10^{-9})
	Improvement (95% Conf. Interval)	1.00× (0.87×; 1.15×)	53.85× (38.44×; 77.48×)	1.00× (0.91×; 1.10×)	3.98× (3.53×; 4.50×)

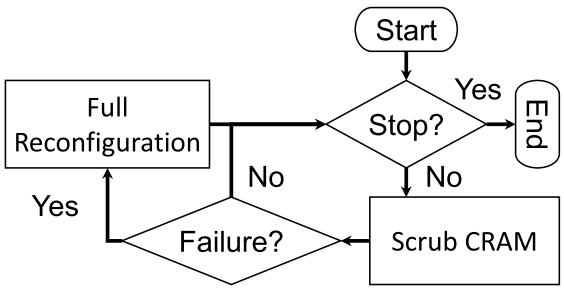

Fig. 8. Test run flow for neutron radiation tests.

bus, and the three Kintex 7 FPGAs were connected to separate JTAG controllers.

Test runs at LANSCE followed the flow shown in Fig. 8. A test run starts with opening the beam shutter after the experiment is initialized. A continuous cycle then follows the configuration memory scrubbing and checking for a failure. If a functional failure is detected, a full reconfiguration is performed and the cycle resumes. This continues until the beam shutter is closed, which ends the test run.

Before the beam shutter is ready to be opened, the experiment has to be initialized. This consists of power cycling the FPGA, configuring the FPGA with a selected design, making sure that the design is operating correctly, and by starting the SRAM scrubbing mechanism and event logger. Correct operation is determined by checking the appropriate status registers over JTAG. All events are logged for analysis. With the experiment initialized, the shutter is ready to be opened.

As the neutron beam passes through the open shutter, configuration upsets begin to occur. They are recorded and corrected by a scrubbing mechanism. Each scrub cycle reports the number of upsets detected and where they occurred. External readback scrubbing was performed on the Kintex 7 FPGA. This consists of reading the configuration memory, comparing the readback against a golden copy, and writing the golden data back to the frames with detected upsets via partial reconfiguration. The Kintex 7 was scrubbed at a rate of 2.5 s per scrub cycle. A combination of external and internal scrubbing was used for the Stratix V. An internal error checking scan is completed every 94–189 ms continually. Detected upsets are then sent to an error message register and to the internal scrubber. It is not known how long it takes the internal scrubber to correct an upset. Internal scrubbing may slow down if upset events happen close together. All detected errors (e.g., single, double-adjacent, and multibit) are read out over JTAG. Each error is reported as corrected or uncorrected by the internal scrubbing mechanism. If any uncorrected upsets are reported, then an external scrub is performed where the correct configuration data are written over the corrupted frames. Reading out upsets and performing an external scrub take 1–25 s with a 4.2-s average. All detected errors are corrected within a subsequent scrub cycle on both FPGAs.

After every scrub cycle on each respective FPGA, a check is performed to see if design failure has occurred (i.e., the functional failure status register is set in the test logic infrastructure). If a failure is not detected, the next scrub cycle will begin. If a failure is detected, the test design will be reset by performing a full device reconfiguration and making sure that the design starts up correctly by checking the status registers over JTAG. SEUs during full reconfiguration may cause the design to initialize improperly. This behavior was observed in both FPGAs in 20% of all programming attempts and occurred 1.6× more frequently when programming a non-TMR design version. When this occurred, the device was reprogrammed again until the design started clearly. Resetting the design was done by reprogramming the FPGA and expecting it to begin operation in a good state. These false starts were excluded from the observed failures in Table IV, because the design in these samples never makes it to a known good operating state.

The test run continues until the beam shutter is closed. The shutter might need to be closed to change the experiment being tested, troubleshoot test setup, or change parts. The shutter was most frequently closed to delimit the previous test run and switch out the design being tested. Collected data were analyzed throughout the test days to help determine, which tests needed more beam time to obtain statistically significant

TABLE V
DEVICE NEUTRON RADIATION DATA

Device	Intel Stratix V	Xilinx Kintex 7
Total Fluence	2.89×10^{11}	1.06×10^{12}
Total Upsets	138,040	551,952
Device Cross Section (95% Conf. Interval)	4.78×10^{-7} (4.75×10^{-7}; 4.80×10^{-7})	5.22×10^{-7} (5.20×10^{-7}; 5.24×10^{-7})
Total CRAM Bits	98,681,196	72,778,176
CRAM Bit Cross Section	4.84×10^{-15}	7.18×10^{-15}

data. Mitigated designs required more beam time to tighten confidence intervals. After the beam shutter is closed, upsets are no longer detected during a scrub cycle. The logging and scrub cycles are stopped, and the next test is set up.

Time stamps were recorded on neutron beam fluence and all test events (e.g., reprogramming, detected upsets, and functional failure). A beam counter recorded fluence data from the dosimeter with an added time stamp in two second intervals. This provides a detailed log of fluence and flux of the neutron beam. In a similar fashion, a time stamp was recorded every time the device was reprogrammed and every time a functional failure was detected. This gives a detailed log of clean starts to failure events. By aligning the beam counter and the test run logs, the total fluence exposure per failure event can be obtained. This information is used later to plot the reliability of the designs over fluence exposure as another means of analyzing the data (see Section VI).

A. Device Data

The neutron cross section for each FPGA device as a whole was also measured as part of this experiment. Table V shows the total fluence expose and the number of detected configuration upsets for each FPGA.[1] Each double-adjacent upset and multibit upset reported by the Stratix V internal scrubbing are counted as two upsets. They make up 1.6% of all upset events in the Stratix V. Because three Kintex 7 FPGAs were tested simultaneously, the total amount of fluence exposure is approximately three times greater than that of the Stratix V. The cross section of the entire device is calculated by dividing the total number of detected upsets by the total fluence. The cross section of a single CRAM cell is approximated by dividing the device cross section by the total number of CRAM bits. The number of CRAM bits for each device was estimated using reports from their respective vendor tools and the structure of configuration memory (i.e., number of frames and bits per frame). It is believed that the actual value is within 10 million of these estimates. The actual number of CRAM bits is unknown, and error may exist due to incorrect estimate (e.g., being off by 10-million bits changes the estimated single-bit cross section by approximately 1×10^{-15} cm^2).

B. Design Data

The total neutron fluence exposure of each test and the number of observed failures are reported in Table IV. The design cross section is determined by dividing the number of observed failures by the total neutron fluence. The 95% confidence intervals are calculated using a common methodology [27]. Because the sensitive cross section of a design does not depend on the total number of CRAM bits contained in the FPGA, the design cross section can be used as a one-to-one comparison metric across architectures. Improvement is a cross section ratio between the unmitigated and mitigated design versions.

Scrubbing logs from both FPGAs show that multiple upsets were detected and corrected every scrub cycle. In scrub cycles with at least 1 upset, the Kintex 7 scrubber reported 1 to 77 upsets averaging 3.1, and the Stratix V scrubber reported 1 to 37 upsets averaging 4.7 with as many as 20 uncorrected upsets. Because the Stratix V scrubber uses a combination of internal and external scrubbing, it is difficult to determine if upsets were allowed to accumulate. Having more than one upset present in the device may compromise the TMR mitigation scheme and allow a failure to occur [31]. Increasing the scrub rate would lower the number of upsets that coexist between scrub cycles, but single particle strikes that cause multibit upsets would still be a concern. This accumulation of upsets most likely increases the reported cross section in Table IV of the mitigated designs with TMR and scrubbing.

VI. RELIABILITY MODEL FIT

The reliability of a design is the probability that the design is still operating correctly as a function of time or fluence exposure. The cross section reflects the mean fluence-to-failure, but the reliability of a design reflects the probability of correct operation at a given point in fluence exposure. This is valuable information, because some mitigation schemes, such as TMR without scrubbing, may have a larger cross section even though they have higher reliability early on in the mission [32].

Given a sufficient number of fluence-to-failure samples, it is possible to plot the reliability of a system as a function of fluence exposure [33]. This section presents the collected neutron radiation data as a plot of reliability verses fluence exposure and discusses how this analysis is performed. In addition to plotting the reliability, classic reliability models that match the SEU mitigation schemes are fitted to the data based on the cross section of the respective designs.

The reliability of a design given a set of fluence-to-failure samples is

$$R(\Phi) = \frac{n(\Phi)}{N}$$

where Φ is the fluence exposure since start of operation, $n(\Phi)$ is the number of *survivors* (i.e., samples still operating correctly) as a function of Φ, and N is the total number of samples in the set. This is adapted from a derivation for the reliability of a system as a function of time given a set of time-to-failure samples in [33]. By observation, the reliability of a design is directly related to the percentage of survivors in a sample set at a given point in fluence exposure.

In this experiment, the set of fluence-to-failure samples for a given design and version is derived from the neutron beam count log and the test run log. As explained in Section V, these

[1]To the best of our knowledge, there is no publicly available measurement of neutron cross section of a single CRAM bit in an Intel Stratix V FPGA

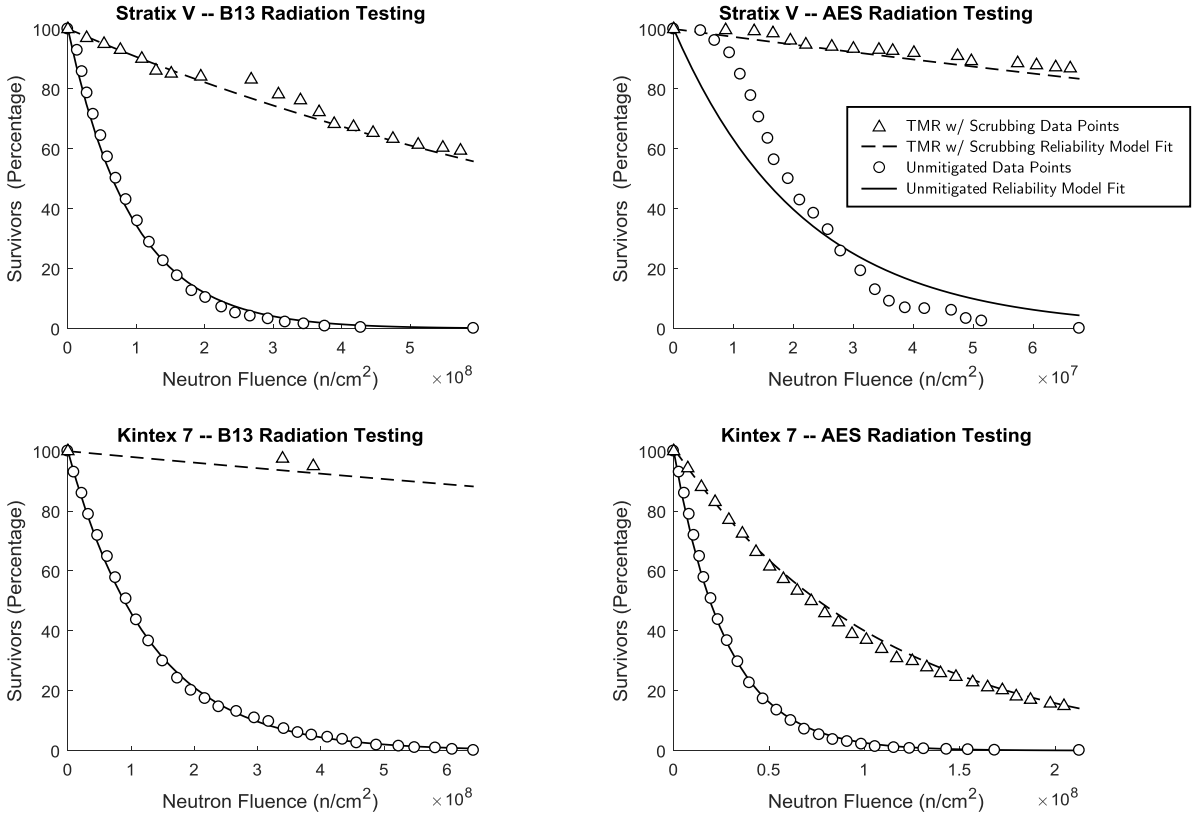

Fig. 9. Reliability plots of the neutron test results with reliability models fitted to the cross section of the unmitigated and mitigated design.

logs contained time-stamped information on fluence exposure and failure events during a test run. Using the time stamps, the fluence-to-failure was determined for each observed failure in the experiment (i.e., fluence exposure of FPGA between a full reconfiguration and a subsequent functional failure). For each event, the fluence-to-failure is known.

Fig. 9 shows the plotted reliability graphs of all of the test designs. The circles represent the unmitigated data points, and the triangles represent the TMR with scrubbing data points. These points are generated as the percentage of surviving samples at a given fluence exposure. In the beginning, all samples are operating correctly. As fluence exposure increases, fewer samples continue to survive.

Reliability models have been developed for simplex systems, TMR systems, and TMR systems with repair [32]. These models are fitted to neutron radiation data points in Fig. 9 by adjusting parameters in the mean-time-to-failure (MTTF) to match the cross section of the design. The MTTF of a simplex system and a TMR system with a repair system is

$$\text{MTTF}_{\text{simplex}} = \frac{1}{\lambda}, \quad \text{MTTF}_{\text{TMR}w/\text{repair}} = \frac{5 + \frac{\mu}{\lambda}}{6\lambda}.$$

For this application, the MTTF is viewed as an average neutron fluence exposure to failure. The failure rate λ is substituted with the cross section of the unmitigated design. The repair rate μ is solved for by setting the MTTF of the TMR system with repair to the inverse of mitigated cross section and substituting λ with the unmitigated cross section. With these parameters adjusted, the respective reliability curves are plotted as overlays on the graphs in Fig. 9. The solid line is the unmitigated simplex model, and the dashed line is the mitigated TMR with repair (i.e., CRAM scrubbing) model.

VII. Discussion

The primary goal of this paper is to compare the dynamic SEU sensitivity of FPGA designs across different SRAM-based FPGA architectures. Comparison of unmitigated results compares the architectural impact on dynamic SEU sensitivity. Comparison of mitigated results reflects the benefits of TMR and scrubbing across the different designs and architectures. A secondary goal of this paper is to compare the low-level, single-bit cross section of the devices.

The unmitigated cross section of the designs was similar across FPGA architectures. This cross section reflects the dynamic SEU sensitivity of the unmitigated designs across architectures, which is affected by logic synthesis, technology mapping, placement, and routing in addition to the architecture of the target FPGA. Each design was synthesized and implemented separately using tools from Intel and Xilinx for the Stratix V and the Kintex 7, respectively. Despite the independent implementations, the unmitigated cross section of a design on the Stratix V is within a factor of 1.4× to that of the same design on the Kintex 7 (i.e., $\sigma_{\text{Stratix V}}/\sigma_{\text{Kintex 7}}$).

TMR and CRAM scrubbing mitigation techniques benefited both designs on both targeted FPGAs. Applying TMR and CRAM scrubbing to designs reduced the dynamic SEU sensitivity of the designs by a factor of 4× to 54×. TMR favored the AES design on the Stratix V and the B13 design on the Kintex 7. There was a greater relative improvement in cross section for the TMR'd version of the AES design on the Stratix V than on the Kintex 7, and the actual cross section

on the Stratix V is also less than that of the Kintex 7. The opposite is true for the TMR'd version of the B13 design. This suggests that TMR benefits are design and device-dependent.

At the low-level device sensitivity, the single-bit cross sections between the two devices are within a factor of 2× of each other. The neutron cross section of a single CRAM cell on the Kintex 7 matches the measurement reported by Xilinx [34]. As expected, the low-level device sensitivity between the two FPGAs is very similar.

VIII. Conclusion

Unmitigated and mitigated versions of two benchmark designs were tested for SEU sensitivity on similar FPGAs with different architectures. Neutron radiation testing shows that TMR and scrubbing benefited both designs on both FPGAs. Similarities were found between the two FPGAs in the dynamic SEU sensitivity of the unmitigated version of each design and in the low-level single-bit CRAM cross section. The SEU mitigated versions of each design favored opposite FPGAs suggesting that benefits from TMR are both the design structure and the FPGA architecture-dependent. Single point failures within the TMR implementation, such as cross-domain errors for TMR on a single FPGA [31], require further study and research and will be addressed in future work. The process used in this paper to compare mitigation techniques across architectures could be applied to other FPGAs and benchmark designs. It was found that the FPGAs tested in this paper are fairly comparable to each other for dynamic SEU sensitivity of a particular design mapped to the FPGA and that TMR and scrubbing reduced the designs' dynamic SEU sensitivity by a factor of 4–54× in neutron radiation testing.

References

[1] R. C. Baumann, "Soft errors in advanced semiconductor devices—Part I: The three radiation sources," *IEEE Trans. Device Mater. Rel.*, vol. 1, no. 1, pp. 17–22, Mar. 2001.

[2] M. Bellato, M. Ceschia, M. Menichelli, A. Papi, J. Wyss, and A. Paccagnella, "Ion beam testing of SRAM-based FPGA's," in *Proc. 6th Eur. Conf. Radiat. Effects Compon. Syst.*, Sep. 2001, pp. 474–480.

[3] A. Lesea, S. Drimer, J. J. Fabula, C. Carmichael, and P. Alfke, "The rosetta experiment: Atmospheric soft error rate testing in differing technology FPGAs," *IEEE Trans. Device Mater. Rel.*, vol. 5, no. 3, pp. 317–328, Sep. 2005.

[4] M. Ceschia *et al.*, "Identification and classification of single-event upsets in the configuration memory of SRAM-based FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 50, no. 6, pp. 2088–2094, Dec. 2003.

[5] L. Sterpone and M. Violante, "Analysis of the robustness of the TMR architecture in SRAM-based FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 52, no. 5, pp. 1545–1549, Oct. 2005.

[6] M. Alderighi, F. Casini, S. D'Angelo, M. Mancini, S. Pastore, and G. R. Sechi, "Evaluation of single event upset mitigation schemes for SRAM based FPGAs using the FLIPPER fault injection platform," in *Proc. 22nd IEEE Int. Symp. Defect Fault-Tolerance VLSI Syst.*, Sep. 2007, pp. 105–113.

[7] K. S. Morgan, D. L. McMurtrey, B. H. Pratt, and M. J. Wirthlin, "A comparison of TMR with alternative fault-tolerant design techniques for FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 54, no. 6, pp. 2065–2072, Dec. 2007.

[8] P. Adell, G. Allen, G. Swift, and S. McClure, "Assessing and mitigating radiation effects in Xilinx SRAM FPGAs," in *Proc. Eur. Conf. Radiat. Effects Compon. Syst. (RADECS)*, Sep. 2008, pp. 418–424.

[9] J. M. Johnson and M. J. Wirthlin, "Voter insertion algorithms for FPGA designs using triple modular redundancy," in *Proc. 18th Annu. ACM/SIGDA Int. Symp. Field Programm. Gate Arrays (FPGA)*, 2010, pp. 249–258.

[10] A. M. Keller and M. J. Wirthlin, "Benefits of complementary SEU mitigation for the LEON3 soft processor on SRAM-based FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 64, no. 1, pp. 519–528, Jan. 2017.

[11] H. M. Quinn *et al.*, "A test methodology for determining space readiness of Xilinx SRAM-based FPGA devices and designs," *IEEE Trans. Instrum. Meas.*, vol. 58, no. 10, pp. 3380–3395, Oct. 2009.

[12] H. Quinn *et al.*, "Using benchmarks for radiation testing of microprocessors and FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 62, no. 6, pp. 2547–2554, Dec. 2015.

[13] Intel Corp. *Stratix V Device Overview*. Accessed: Sep. 26, 2017. [Online]. Available: https://www.altera.com/en_US/pdfs/literature/hb/stratix-v/stx5_51001.pdf

[14] Xilinx Inc. *7 Series FPGAs Data Sheet: Overview*. Accessed: Sep. 26, 2017. [Online]. Available: https://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf

[15] M. *Chang*, "Device architecture," in *Reconfigurable Computing*, S. Hauck and A. Dehon, Eds. Burlington, MA, USA: Morgan Kaufmann, 2008, ch. 1.

[16] Xilinx Inc. *7 Series FPGAs CLB User Guide*. Accessed: Sep. 6, 2017. [Online]. Available: https://www.xilinx.com/support/documentation/user_guides/ug474_7Series_CLB.pdf

[17] Intel Corp. *Stratix V Device Handbook*. Accessed: Oct. 31, 2017. [Online]. Available: https://www.altera.com/en_US/pdfs/literature/hb/stratix-v/stx5_core.pdf

[18] *Intel Quartus Prime Standard Edition Handbook*, vol. 2, Intel Corp., Mountain View, CA, USA, May 2017. Accessed: Sep. 26, 2017. [Online]. Available: https://www.altera.com/enUS/pdfs/literature/hb/qts/qts-qps-5v2.pdf

[19] *DE5-Net FPGA Development Kit*. Accessed: Sep. 8, 2017. [Online]. Available: http://de5-net.terasic.com

[20] *Xilinx Kintex-7 FPGA KC705 Evaluation Kit*. Accessed: Aug. 30, 2017. [Online]. Available: https://www.xilinx.com/products/boards-and-kits/ek-k7-kc705-g.html

[21] F. Corno, M. S. Reorda, and G. Squillero, "RT-level ITC'99 benchmarks and first ATPG results," *IEEE Des. Test. Comput.*, vol. 17, no. 3, pp. 44–53, Jul. 2000.

[22] *AES::Overview::OpenCores*. Accessed: Oct. 8, 2017. [Online]. Available: https://opencores.org/project,tiny_aes

[23] H. R. Zarandi and S. G. Miremadi, "Dependability evaluation of Altera FPGA-based embedded systems subjected to SEUs," *Microelectron. Rel.*, vol. 47, nos. 2–3, pp. 461–470, Feb. 2007.

[24] Brigham Young Univ., Provo, UT, USA. (Sep. 2009). *BYU EDIF Tools*. Accessed: Sep. 26, 2017. [Online]. Available: http://byuediftools.sourceforge.net

[25] J. Johnson *et al.*, "Using duplication with compare for on-line error detection in FPGA-based designs," in *Proc. IEEE Aerosp. Conf.*, Mar. 2008, pp. 2322–2332.

[26] *Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices*, Standard JESD89A, JEDEC Solid State Technology Association, 2006. [Online]. Available: https://www.jedec.org/sites/default/files/docs/JESD89A.pdf

[27] H. Quinn, "Challenges in testing complex systems," *IEEE Trans. Nucl. Sci.*, vol. 61, no. 2, pp. 766–786, Apr. 2014.

[28] H. M. Quinn, D. A. Black, W. H. Robinson, and S. P. Buchner, "Fault simulation and emulation tools to augment radiation-hardness assurance testing," *IEEE Trans. Nucl. Sci.*, vol. 60, no. 3, pp. 2119–2142, Jun. 2013.

[29] P. Ramachandran, P. Kudva, J. Kellington, J. Schumann, and P. Sanda, "Statistical Fault Injection," in *Proc. IEEE Int. Conf. Depend. Syst. Netw. FTCS DCC (DSN)*, Jun. 2008, pp. 122–127.

[30] A. M. Keller, "Using on-chip error detection to estimate FPGA design sensitivity to configuration upsets," M.S. thesis, Dept. Elect. Comput. Eng., Provo, UT, USA, 2017. [Online]. Available: http://scholarsarchive.byu.edu/etd/6302/

[31] H. Quinn, K. Morgan, P. Graham, J. Krone, M. Caffrey, and K. Lundgreen, "Domain crossing errors: Limitations on single device triple-modular redundancy circuits in Xilinx FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 54, no. 6, pp. 2037–2043, Dec. 2007.

[32] D. Siewiorek and R. Swarz, *Reliable Computer Systems*, 3rd ed. Natick, MA, USA: A K Peters, 1998.

[33] M. L. Shooman, *Reliability of Computer Systems and Networks: Fault Tolerance, Analysis, and Design*. Hoboken, NJ, USA: Wiley, 2002.

[34] Xilinx Inc. *Device Reliability Report*. Accessed: Sep. 22, 2017. [Online]. Available: https://www.xilinx.com/support/documentation/user_guides/ug116.pdf