Using Partial Duplication With Compare to Detect Radiation-Induced Failure in a Commercial FPGA-Based Networking System

Andrew Keller, Jared Anderson, and Michael Wirthlin
Department Electrical and Computer Engineering
Brigham Young University
Provo, USA
andrewmkeller@byu.edu

Shi-Jie Wen, Rita Fung, and Conner Chambers
Cisco Systems, Inc.
San Jose, USA

***Abstract*—Duplication with compare, a circuit-level fault-detection technique, is used in this study in a partial manner to detect radiation-induced failures in a commercial FPGA-based networking system. A novel approach is taken to overcome challenges presented by multiple clock domains, the use of third-party IP, and the collection of error detection signals dispersed throughout the design. Novel fault injection techniques are also used to evaluate critical regions of the target design. Accelerated neutron radiation testing was performed to evaluate the effectiveness of the applied technique. One design version was able to detect 45% of all failures with the proposed technique applied to 29% of the circuit components within the design. Another design version was able to detect 31% of all failures with the proposed technique applied to only 8% of circuit components.**

***Index Terms*--electronic design automation, fault detection, radiation effects , redundancy, reliability.**

I. Introduction

Terrestrial radiation can cause high-performance networking systems to fail [1]. The likelihood of radiation-induced failures in a *single* device is very low, but failures occur more often in large-scale deployments [2]. SRAM-based field programmable gate arrays (FPGAs) are commonly used in high performance networking products. They are susceptible (with low likelihood in terrestrial environments) to radiation-induced upsets, or single event upsets (SEUs). An SEU in the configuration memory (CRAM) of an FPGA can directly alter functionality and lead to failure. SEUs that cause undetected, persistent functional failure are of major concern [1], (e.g., persistent traffic loss on any connection without system awareness). Extensive research has been conducted on the effects of CRAM SEUs on SRAM-based FPGAs to better understand and address these failures [3], [4].

Duplication with compare (DWC) [5] is the technique employed in this study to detect failures caused by SEUs. DWC detects failures by comparing the outputs of two identical circuits. If a discrepancy is found, an error is reported. This circuit-level fault-detection scheme is depicted in Fig. 1. The detection logic is also duplicated to filter out false detections that results from SEUs in detection logic. In [5], DWC applied to all components in a circuit was able to accurately detect 99.9% of all SEU-induced failure events in SRAM-based FPGA designs.

This work was supported by ChipIR at the ISIS neutron source of the Rutherford Appleton Laboratory (UK) under proposal 1900120.

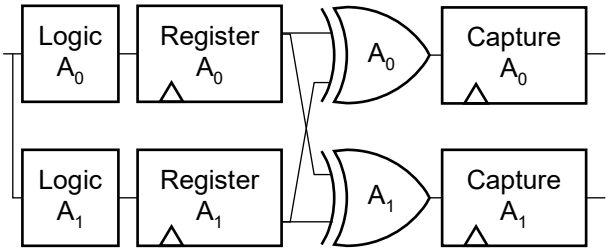

Fig. 1. Duplication with Compare using Redundant Detection Logic.

DWC is difficult to apply to larger more complex FPGA designs. Several factors contribute to the challenge of application including timing closure, resource utilization, multiple clock domains, and the use of third-party IP with involved constraints. This work demonstrates a novel method of applying DWC to portions of a complex commercial FPGA design. This technique, referred to as partial DWC, is applied after product development and significantly improves system awareness of failures that are otherwise undetectable.

The proposed partial DWC involves evaluating critical regions of an SRAM-based FPGA design and applying duplication to the evaluated regions. Random fault injection into targeted regions of the design is used to evaluate the sensitivity of sub-modules in the design. DWC is applied in an automated fashion to the most sensitive regions using a custom-built electronic design automation (EDA) tool.

Two partial DWC design versions were implemented on the commercial device and were evaluated in an accelerated neutron test. The first design applied DWC to three critical regions, covering 29% of components in the design (29% of all lookup-table, registers, or other primitive components were duplicated). The second design focused on applying DWC to sensitive circuit structures within the three critical regions and only covered 8% of the design. A neutron test was conducted to determine the effectiveness of partial DWC to detect system failures. The neutron test showed that the designs with partial DWC were able to successfully detect a large portion of silent errors while only duplicating small portions of the design. The first partial DWC design was able to detect 45% of persistent silent network disruptions and the second partial DWC design was able to detect 31% of persistent silent network disruptions.

978-1-7281-3199-3/20/$31.00 ©2020 IEEE

II. Single Event Upsets and FPGA-Based Networking

SEUs can disrupt the flow of traffic in FPGA-based networking systems. SRAM-based FPGAs are used in many high-end networking products to process large amounts of network traffic. This study examines the impact of SEUs on a commercial FPGA-based networking switch. The objective of a networking switch is to forward all incoming traffic to its intended destination. When an SEU occurs in the CRAM of an FPGA it can alter the device functionality and prevent the proper flow of traffic.

Generally, a network switch architecture on an FPGA consists of multiple independent streams of data, buffers, and control logic for arbitration based on packet information. Each of these constructs is implemented and interfaced with using FPGA resources. FPGA resources include registers, lookup tables, memory blocks, arithmetic units, interconnects and high-speed serial IO ports. CRAM bits are associated with these resources, and the values held by those bits control resource behavior and functionality.

One of the unique challenges of SEUs in CRAM is that while the SEU is present it directly alters the underlying circuitry while the design is active. Unlike SEUs in ECC protected memory, there is no opportunity to correct the upsets before it effects the behavior of the design. As soon as the SEU occurs, the circuit is broken, and undesirable behavior can result. For example, SEUs can instantly alter logic equations, disconnect or short connections between components, and corrupt the values of user memories [3]. Many FPGAs provide internal capability to scrub or repair these upsets, but during the short time that the upset is present it can still adversely affect the design. This challenge presents a need for additional circuit-level fault-detection in systems that require high reliability.

Many high-level protocols are put in place to detect and respond to undesirable behavior, but even these protocols can fail to detect the most hazardous of SEU-induced failure modes—persistent silent network disruption. This failure mode is referred to as network disruption or failure throughout the paper and is characterized by a loss of traffic flow that goes undetected by the system and remains present until the system is manually rebooted. System-level redundancy with switchover capability is used to minimize the impact of persistent silent network disruption. This study applies DWC to portions of the FPGA design to improve the system's ability to detect this failure mode.

The severity of high impact failure modes [1] and their increased occurrence in large-scale deployments [2] justifies the application of additional fault-detection techniques. This holds true even though the likelihood of failure in a *single* device is very low. For example, a device with a failure rate of 150 FIT (failures in time per billion hours of operation) at sea level would experience one failure every seven-hundred sixty-one years on average, which may be tolerable. However, the likelihood of failure in a large-scale deployment may be too high [2]. A mass deployment of ten-thousand instances of the same device at a higher elevation [6] would experience a failure event once a week on average. The impact severity and increased frequency of a failure mode in large-scale deployments justifies to application of additional fault-detection techniques.

III. Network System and Test Setup

The networking system used in this study is a commercial campus backbone switch. It is typically used to link smaller networks together. Fig. 2 provides a high-level layout of its components. The system consists of a controller board, an integrated network board and four open bays where modular network boards (often referred to as line cards) can be added to increase the systems network capacity.

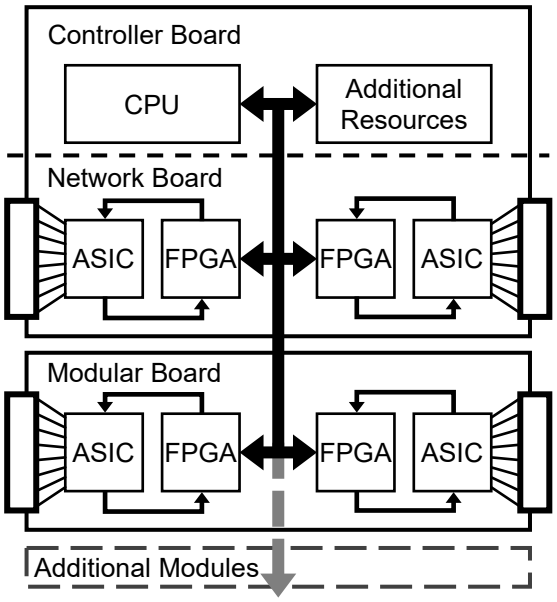

Fig. 2. Campus Network Backbone Switch System Layout.

Each network board has two banks of 8 ports that are paired with an ASIC and an FPGA for processing and transferring network data. All sets of connections are connected to the system's backplane and the entire system is governed by the CPU on the controller board. The FPGA design targeted in this study resides in a Virtex-7 FPGA (XC7VX330T) on a modular network board in the system.

A test infrastructure was developed for this study to provide interesting stimulus and observe failure events. The test infrastructure assists in accomplishing three things. First, the failure rate of SEU-induced network disruption can be estimated through radiation testing or fault injection (simulated corruption of CRAM) using the test infrastructure. Second, critical regions of the design can be evaluated. Finally, the effectiveness of the applied fault-detection technique can be evaluated.

A diagram of the test infrastructure is shown in Fig. 3. A single modular network board of the commercial network system is connected to a network traffic generator. A random stream of data is presented to the network switch by both ports on the traffic generator. The network switch is configured to redirect all incoming traffic through all of the ports in a loopback fashion such that traffic entering the first port will travel through all ports and return to the traffic generator out of the last port and the same in reverse (from the last to the first port). This stimulus ensures that data flows through key resources in the FPGA design and

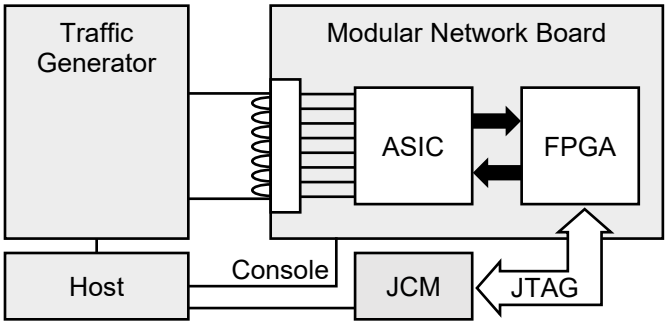

Fig. 3. Test Infrastructure for Fault Injection and Radiation Testing.

makes it possible to observe network disruptions. The FPGA on the modular extension is connected to a custom JTAG configuration manager (JCM) [11], which enables observation of upsets in CRAM and the simulated corruption of CRAM through fault injection. The JCM, traffic generator, and the network switch console terminal are all connected to a host computer. The host computer orchestrates the flow of experiments on the system.

Fault injection is used in this study to validate the test infrastructure prior to radiation testing, to estimate the failure rate of SEU-induced network disruption, and to identify critical regions in the designs. Fault injection is a common reliability testing technique [8]. It is the purposeful corruption of CRAM to emulate SEUs. It is performed in this study via partial reconfiguration by reading out a frame of CRAM, inverting a single bit, and writing the corrupted frame back into the CRAM of the FPGA. Fig. 4 shows an image of the test setup used for fault injection.

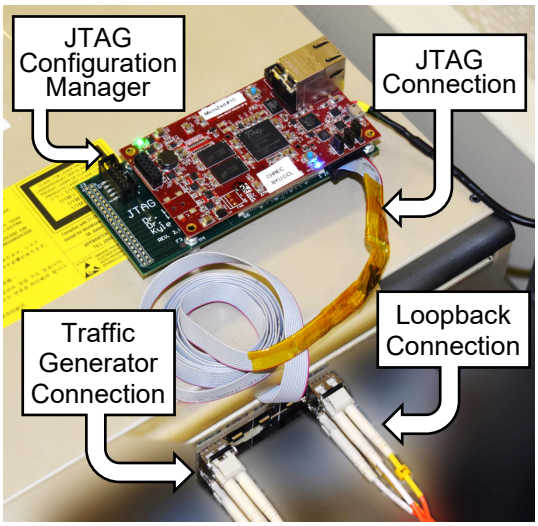

Fig. 4. Image of the Test Infrastructure Setup Used for Fault Injection.

A simple test flow is used in this study (see Fig. 5). First, the system is brought into a working state with traffic flowing correctly through all ports. Then a CRAM bit is purposefully corrupted if fault injection is being used, or the FPGA is exposed to the accelerated neutron beam if testing under radiation. The flow of network traffic is periodically monitored. Once a second, the flow of network traffic is checked. Under fault injection, if the flow is still good, then the fault is repaired, a new CRAM bit is corrupted, and the test continues. Under radiation testing, if the flow is still good, the test continues to the next check. If ever the flow of data drops below 90% capacity for an extended period of time (3 seconds) without recovery, the system is rebooted, and a persistent silent network disruption is recorded. Other failure modes detected by the network system are not considered persistent silent network disruptions and are noticed only to bring the system back into a working state.

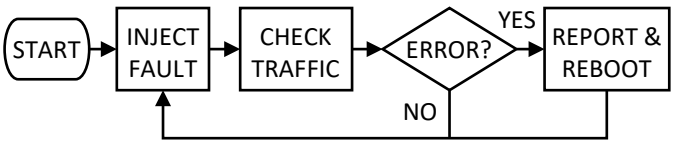

Fig. 5. Campus Network Backbone Switch System Layout.

IV. Critical Region Evaluation via Fault Injection

It is anticipated that some regions in a design may be more prone to SEU-induced failure than others. SEUs can happen anywhere in the design, but if an upset occurs in a particular region, a resultant failure may be more likely. Potentially, the failure rate of the design can be unevenly distributed across components in a design. For example, half of all failures could result from upsets within one-quarter of the design. To maximize the error detection coverage of partial DWC, high risk regions should be prioritized when selecting areas for duplication. By targeting critical regions for duplication, significant error coverage can be achieved with limited overhead.

A fault injection test was developed to accurately estimate the proportional failure rate of a design region. This test uses the test infrastructure setup for random fault injection within a targeted region of the design. To test only one region, random fault injection is performed on the essential bits that correspond to that region. Essential bits are a subset of the CRAM bits which are classified as associated with the circuitry of the design. A list of all the essentials bits for a design can be created during bitstream generation using vendor tools. For this study Xilinx's Vivado 2016.2 was used.

The essential bits corresponding to a targeted region are identified by removing unwanted components from the placed and routed design and running bitstream generation again to create a new list of essential bits. The intersection between the original essential bits list (with the whole design placed and routed) and the new essential bits list (with only the targeted region placed and routed) is taken to obtain the desired bits. Taking the intersection excludes minor inaccuracies due to changes in peripheral routing.

Only a fraction of the essential bits actually affect the functionality of the design. If an SEU in a particular CRAM bit affects the functionality of the implemented design, the bit is classified as a critical bit [9]. Critical bits are identified via fault injection; if a failure results from injecting a fault into a particular bit, the bit is considered critical. All critical bits are considered essential, but not all essential bits are considered critical. By randomly sampling essential bits via fault injection, the number of critical bits in a region can be estimated without exhaustively sampling each essential bit in the region.

Three regions were marked for their potential as critical regions. These regions are the Packet Reader (PR), the Traffic Manager (TM), and Interlaken sub-modules of the design. Table 1 shows the distribution of essential bits across the whole design and the selected regions. There are approximately 79.3 million CRAM bits in the target device. Random fault injection was performed within the essential bits each targeted region to estimate the number of critical bits with each region. The table shows the total number of injected faults, the number of observed failures and the corresponding estimated sensitivity rate. From the sensitivity rate and number of essential bits for the evaluated region, the number of critical bits for that region is estimated.

Table 1. Essential Bits and Random Fault Results Within a Target Region.

Region	Essential Bits	Faults Injected	Failures	Sensitivity	Critical Bits
Whole Design	27.1M	29624	360	1.2%	325.2K
PR	1.7M	3628	104	2.9%	49.3K
TM	2.1M	23402	467	2.0%	42.0K
Inter	4.9M	19627	435	2.2%	107.8K

The results of regional fault injection are shown in Table 2. 95% confidence intervals are provided on the percent of critical bits based on the statistics of population sampling. The PR sub-module makes up 6% of the design based on essential bits and relative resource utilization, yet it contributes to 14% of the devices total estimated critical bits. This region disproportionately contributes to the critical bits (a small region with a large contribution). As shown in Table 2, the essential bits corresponding to the PR region where 2.36× more likely to result in failure if upset than the average essential bit in the device. Tests on the Traffic Manager and Interlaken concluded in similar results. On the other hand, upsets outside of these selected regions are 1.70× less likely to result in failure than the average essential bit in the design.

Table 2. Distribution of Critical Bits Within Sub-Regions of the Design.

Region	Percent of Overall Design	Percent of Critical Bits	Sensitively compared to whole design
PR	6.1%	14.4±4.7%	2.36x
TM	7.7%	12.6±2.7%	1.63x
Inter	18%	32.9±7.1%	1.83x
All 3	31.8%	59.9%±14.5%	1.88x
Other	68.2%	40.1±14.5%	0.59x

Analysis of these findings show that the sensitivity of the device is not evenly distributed. Upsets in some regions of the design are more likely to result in failures than others. Partial DWC takes advantage of this by targeting highly sensitive modules. By applying DWC to all three regions mentioned previously, 59.9% of failures could potentially be detected while duplicating only 31.8% of the design. This information is what was used to guide the selection of components for DWC in this study.

V. Partial DWC Implementation

Partial DWC was applied in an automated fashion using a custom-built EDA tool. The implementation flow is shown in Fig. 6. First, the original HDL source code undergoes logic synthesis. This produces a detailed list of design components and connectivity between components as a netlist. The netlist is input to the custom EDA tool. The tool analyzes the design, selects components to apply DWC to under user guidance, and then implements DWC on the selected components. The revised netlist is then imported into vendor tools to be mapped, placed, and routed. A resultant bitstream can then be used to load the partial DWC design onto the target FPGA.

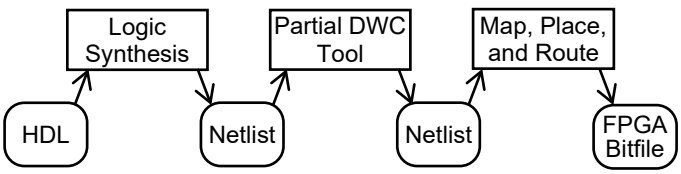

Fig. 6. Partial DWC Insertion Flow.

A. Challenges

While DWC is a promising fault-detection technique, there are a number of challenges that arise when it comes to applying this technique on a large commercial design. These challenges included, but are not limited to: timing closure, resource utilization, clock domain crossings, use of third-party IP, and the collection of dispersed error detection signals. Several novel approaches were taken to address these challenges and successfully implement partial DWC on the target design.

Subsections of the design were selected for DWC to curtail its impact on timing and resource utilization. DWC does not add logic along the critical path but it does increase resource utilization and add routing congestion to the design. This makes it harder to meet timing constraints. Resource limitations also prevent the application of DWC. In this study, the targeted baseline design begins with high resource utilization (see Table 3). There are not enough resources to duplicate user memory blocks (BRAM) and it would be difficult to place, route, and meet timing on the design if every register and lookup table (LUT) were duplicated.

Another issue that could prevent the successful application of DWC is that of clock domain crossings and synchronization between redundant circuits [10]. DWC compares signals on a clock cycle by clock cycle basis. If the redundant circuits become even one clock cycle offset from each other, the ability to detect errors becomes compromised. If a synchronizer between clock domains is duplicated, then the outputs of the redundant synchronizers can become out of sync. This in turn causes the remainder of the redundant circuits to fall out of sync with each other and results in a false positive. To address this issue, two steps were taken. First, the connectivity between components was analyzed to identify clock domain crossings. Then, any synchronizers (chains of registers driven by other clock domains), user memory blocks driven by multiple clock domains, and blackboxes (encrypted third party IP) were excluded from DWC.

It was found in this study that the use of third-party IP carried with it a set of constraints dependent on the hierarchical organization of the implemented design. Removing hierarchical boundaries through flattening simplifies the application of DWC on the design, but it had other unintended side effects including the complication of constraints. To address this issue, DWC was applied across hierarchy by replicating any necessary ports among hierarchical entities in the netlist and connecting redundant components together through the replicated ports. As a result, a minimum number of constraint changes were needed so that the constraints be correctly applied to the design.

A final challenge in applying DWC to a complex system identified in this study is the collection of error detection signals dispersed throughout the design. Error detectors are needed anywhere in the design where a signal transitions from duplicated logic to unduplicated logic. When components are excluded from DWC, connectivity between selected components becomes discontinuous and small clusters of replicated logic form. These smaller groups of logic considerably increase the number of detection voters needed. The collection of thousands of error detection signals presents challenges in routing congestion because so many signals spread through the design that need to be reduced down to a single pair of signals to indicate that an error has been detected. To address this issue, only clusters greater than one-hundred components were selected for DWC and an asynchronous reduction network was implemented. Detection events were captured in their proper clock domain and then the error signal was propagated through an asynchronous reduction network to an edge detector for reporting the detected failure. Other implementations of detection signal collection resulted in intolerable routing congestion or timing issues.

B. Implementations

This study applies DWC in a partial manner to detect network disruptions that are persistent and undetected by the system. DWC is applied to areas of the target design that have been evaluated through targeted fault injection as more prone to radiation-induced network disruption. Two variants of the design were generated. The first variant applied DWC to the three critical regions evaluated through targeted random fault injection. This variant covered 29% of components in the design and required more than twenty-six hundred pairs of error detectors. The second variant applied DWC to components within the three critical regions that were identified as more sensitive through graph analysis as logic between strongly connected components (SCC) [11]. This resulted in a design with 8% DWC coverage. The number of needed detection pairs was reduced to less than seventeen hundred for this design version. Table 3 shows the resource utilization for each design version. The number of BRAMs, global clock buffers (BUFGs), registers, LUTs, and IO are included along with their percent utilization (percentage of total device resources utilized). The DWC Coverage row indicates the percentage of the original design components (registers, lookup tables, other primitive components) that were covered by DWC. Detector Pairs is the number detector pairs inserted into the design for partial DWC.

Table 3. Virtex 7 330 T Resource Utilization.

Virtex 7 330T	Baseline	Partial DWC PR/TM/INTER	Partial DWC Between SCC
Slices	40,826 (80%)	49,016 (96%)	44,814 (88%)
Registers	136,766 (34%)	180,516 (44%)	152,106 (37%)
LUTs	99,165 (49%)	134,639 (66%)	113,278 (56%)
BRAMs	457.5 (61%)	569 (76%)	477 (64%)
BUFGs	30 (94%)	30 (94%)	30 (94%)
IOs	622 (89%)	622 (89%)	622 (89%)
DWC Coverage	0%	29%	8%
Detector Pairs	0	2,627	1,687

VI. Neutron Radiation Test and Results

Neutron radiation testing of the baseline design and DWC variants was conducted at the ChipIR experiment of the ISIS neutron source of the Rutherford Appleton Laboratory in the United Kingdom [12] in March of 2019. The commercial network switch was aligned perpendicular to the neutron beam aperture such that one of the Virtex 7 330 T FPGAs on a modular network board was directly over the neutron beam flight path. This setup is shown in Fig. 7. A two-inch collimator was used.

For this experiment, the beam shutter was only opened after the network system was in a working state, with traffic flowing correctly, and the beam shutter was closed once a failure was detected to allow the system time to reboot between failures without influencing results. A detailed log of the neutron fluence and device events was used to record the total fluence exposure of each design version.

Tables 4 and 5 display the results of the neutron radiation test. The rows in Table 4 are defined as follows: SEUs are the number of SEUs detected by the JCM, total failures are any persistent loss of traffic on any port, detected failures are the number of persistent traffic loss events detected by partial DWC, accuracy reflects the percentage of failures detected by partial DWC, and false detection reflects the percentage of failure detection events

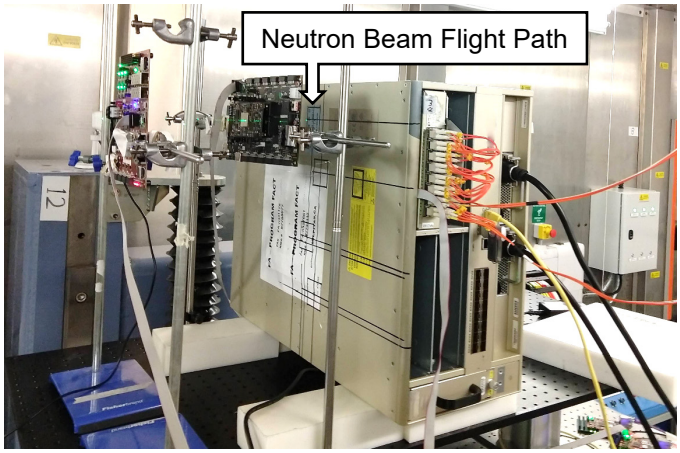

Fig. 7. Accelerated Neutron Beam Test Setup at ChipIR.

that do not correspond to an actual failure event. The partial DWC designs detected almost one-third to less than one-half of all failure events. This is significant considering that a relatively small amount of DWC was applied to the system.

It is important to note that false detection greater than 50% is expected of a DWC fault-detection scheme. This is because one on the redundant copies does not influence the behavior of the design. It is only used for comparison purposes. Thus, errors in the secondary copy trigger a failure detection even though these errors would have no influence on the primary design. At least half of all detection events should then be false positives.

Table 4. Partial DWC Accuracy in Accelerated Neutron Testing.

Design	Baseline	Partial DWC PR/TM/INTER	Partial DWC Between SCC
SEUs	459	675	1024
Total Failures	11	20	32
Detected Failures	0	9	10
Accuracy	0%	45%	31%
False Detection	0%	61%	58%

Table 5 shows the number of undetected failures for each design with the corresponding cross section and FIT rate for undetected failures. The total number of undetected failures divided by neutron fluence yields the cross section. The cross section is converted to FIT using the New York City reference flux of 13 n cm^{-2} h^{-1} [7]. 95% confidence intervals are provided for both the neutron cross section and the corresponding FIT rate. The partial DWC design that covered the PR/TM/INTER regions reduced the FIT rate of undetected failures to 106 FIT, a 1.4× improvement. The second partial DWC design that covered logic between SCC resulted in a measured 132 FIT rate of undetected failures.

Table 5. Accelerated neutron testing results.

Design	Baseline	Partial DWC PR/TM/INTER	Partial DWC Between SCC
Undetected Failures	11	11	22
Fluence	9.37E+8 n/cm^2	1.35E+9 n/cm^2	2.17E+9 n/cm^2
Cross Section (95% conf.)	1.17E-8 cm^2 (5.7E-9, 2.1E-8)	8.17E-9 cm^2 (3.3E-9, 1.3E-8)	1.01E-8 cm^2 (5.9E-9, 1.4E-8)
FIT (95% conf.)	152 (74, 273)	106 (43, 169)	132 (77, 187)

VII. Conclusion

This study applied duplication with compare to a complex FPGA design to increase the failure detection rate. Several challenges arose in the application of DWC to the design. These challenges were addressed by a novel application approach.

Critical region evaluation prior to radiation testing identified that almost two-thirds of persistent silent network disruptions originate from CRAM upsets in the packet reader, traffic manager, and Interlaken submodules. These components make up approximately 32% of the design. DWC on these submodules (excluding several components) covered 29% of the original design. This detection scheme was able to detect 45% of otherwise undetectable failures. Similarly, DWC of only 8% was able to detect 31% of errors.

Partial DWC in this study was shown to significantly improve system awareness of radiation-induced failures in a complex commercial FPGA design for a networking application. Likely, with further improvement of these techniques, even greater awareness may be obtainable while needing fewer resources. This could potentially be achieved if the DWC selection were refined to include more critical components and fewer less critical components.

References

[1] A. Silburt *et al.*, "Specification and verification of soft error performance in reliable internet core routers," in *IEEE Transactions on Nuclear Science*, vol. 55, no. 4, pp. 2389-2398, Aug. 2008.

[2] H. Quinn and P. Graham, "Terrestrial-based radiation upsets: a cautionary tale," *13th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'05)*, Napa, CA, USA, 2005, pp. 193-202.

[3] Battezzati, Niccolò, Luca Sterpone, and Massimo Violante. "Reconfigurable field programmable gate arrays: Failure modes and analysis." Reconfigurable Field Programmable Gate Arrays for Mission-Critical Applications. Springer, New York, NY, 2011. 37-83.

[4] Quinn, Heather, et al. "An introduction to radiation-induced failure modes and related mitigation methods for Xilinx SRAM FPGAs." ERSA. 2008.

[5] J. Johnson *et al.*, "Using Duplication with Compare for On-line Error Detection in FPGA-based Designs," *2008 IEEE Aerospace Conference*, Big Sky, MT, 2008, pp. 2322-2332.

[6] *Measurement and reporting of alpha particle and terrestrial cosmic ray-induced soft errors in semiconductor devices*, JEDEC Solid State Technology Association Std. 89A, 2006. [Online]. Available: https://www.jedec.org/sites/default/files/docs/JESD89A.pdf

[7] A. Gruwell *et al.*, "High-speed FPGA configuration and testing through JTAG," in *2016 IEEE AUTOTESTCON*, 2016, pp. 1–8.

[8] H Quinn *et al.*, "Fault Simulation and Emulation Tools to Augment Radiation-Hardness Assurance Testing," in *IEEE Transactions on Nuclear Science*, vol. 60, no. 3, pp 2119–2142, 2013.

[9] Robert Le, "Soft error mitigation using prioritized essential bits," Application Note, Xilinx, 2012.

[10] Y. Li *et al.*, "Synchronization techniques for crossing multiple clock domains in FPGA-based TMR Circuits," *IEEE Trans. Nucl. Sci.*, vol. 57, no. 6, pp. 3506–3514, Dec. 2010.

[11] B. Pratt *et al.*, "Fine-Grain SEU Mitigation for FPGAs Using Partial TMR," in IEEE Transactions on Nuclear Science, vol. 55, no. 4, pp. 2274-2280, Aug. 2008.

[12] M. Wirthlin *et al.*, "Validation of Partial Duplication With Compare for FPGA Systems," STFC ISIS Neutron and Muon Source, 2019, Available online: https://doi.org/10.5286/ISIS.E.RB1900120