Partial TMR for Improving the Soft Error Reliability of SRAM-Based FPGA Designs

Andrew M. Keller, *Student Member, IEEE*, and Michael J. Wirthlin, *Senior Member, IEEE*

***Abstract*— Triple modular redundancy (TMR) is a single-event upset (SEU)-mitigation technique that uses three circuit copies to mask a failure in any one copy. It improves the soft error reliability of designs implemented on SRAM-based field-programmable gate arrays (FPGAs) by masking the effects of upsets in the configuration memory. Although TMR is most effective when applied to an entire FPGA design, a reduction in the sensitive cross section of an FPGA design can be obtained by applying TMR selectively. This article explores several approaches for selecting components to triplicate. The benefit is a reduction in the neutron cross section for any output error as a percentage compared to that of a non-triplicated design. The cost is the percentage of components triplicated. The goal is to maximize the benefit–cost ratio. Twenty-five different selections are tested on a benchmark design. Some selections increase the cross section; others decrease the cross section significantly.**

***Index Terms*— Fault injection, field programmable gate arrays (FPGAs), fault tolerance, integrated circuit reliability, neutron radiation effects, radiation hardening (electronics), redundancy, single-event upsets (SEUs), triple modular redundancy (TMR).**

I. Introduction

SRAM-BASED field-programmable gate arrays (FPGAs) contain a large amount of state that is susceptible to radiation-induced corruption. Atomic particles that pass through the device may induce a single-event upset (SEU), which is a soft error that alters values stored in memory cells [1]. Configuration memory (CRAM) in an SRAM-based FPGA stores the configuration of components. Additional state stores the values of memory elements used within an active design. When CRAM or additional memory values become corrupted, the proper functionality of a design may be jeopardized.

Triple modular redundancy (TMR) is an SEU mitigation technique that is often employed in SRAM-based FPGA designs to improve soft error reliability. TMR uses multiple identical copies and majority voters to mask a failure within a single circuit copy (see Fig. 1). As long as two or more copies and the corresponding voter are error-free, the output of the design will also be error-free. This technique has been shown to effectively improve the soft error reliability of digital circuits implemented on SRAM-based FPGAs [2]–[4].

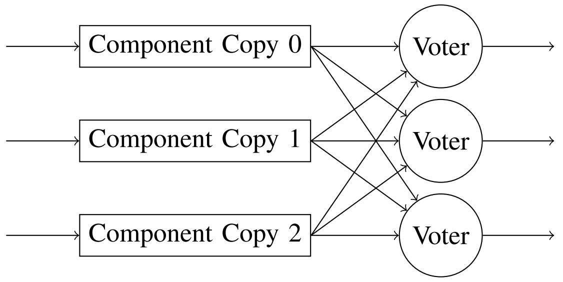

Fig. 1. Spatial TMR with triplicated voters.

In many situations, TMR cannot be applied to every circuit component in an SRAM-based FPGA design. Resource limitations, tight timing constraints, the use of encrypted third-party IP, and other factors limit the amount of TMR that can be applied to a design implemented on an SRAM-based FPGA. In these situations, TMR may often still be applied to a subset of the components that make up a design. Applying TMR to only a subset of components within a design is known as selective [5] or partial TMR [6]. Under partial TMR, some level of soft error reliability benefit may still be obtained.

The effectiveness of partial TMR is determined by its benefit–cost ratio. In this article, the benefit of partial TMR is a reduction in the neutron cross section for any observable output error. This reduction benefit is measured as a percentage decrease compared to the neutron cross section of a design without any TMR. A negative benefit represents an increase in cross section. The cost metric in this article is defined as the percentage of components included in partial TMR. The benefit–cost ratio divides the benefit (i.e., neutron cross section reduction percentage) by the cost (i.e., percentage of components triplicated) to normalize the benefit gained from the application of partial TMR.

The effectiveness of partial TMR is greatly influenced by the subset of design components that are selected for TMR. Some selections are likely to be more effective than others. Effectiveness is influenced by a number of different factors. These factors include the number of non-triplicated voters inserted into the design to support the selection, the number of triplicated connections between components, and the relationships between selected components. This article explores several different selection approaches that stress different factors of effectiveness.

Manuscript received September 8, 2020; revised February 7, 2021 and March 24, 2021; accepted March 30, 2021. Date of publication April 5, 2021; date of current version May 20, 2021. This work was supported in part by the Utah National Aeronautics and Space Administration (NASA) Space Grant Consortium, in part by the Industry/University Cooperative Research Centers (I/UCRC) Program of the National Science Foundation (NSF) under Grant 1738550, and in part by the Los Alamos Neutron Science Center (LANSCE) under Proposal NS-2019-8294-A.

The authors are with NSF Center for Space, High-Performance, and Resilient Computing (SHREC), Department of Electrical and Computer Engineering, Brigham Young University, Provo, UT 84602 USA (e-mail: andrewmkeller@byu.edu; wirthlin@byu.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNS.2021.3070856.

Digital Object Identifier 10.1109/TNS.2021.3070856

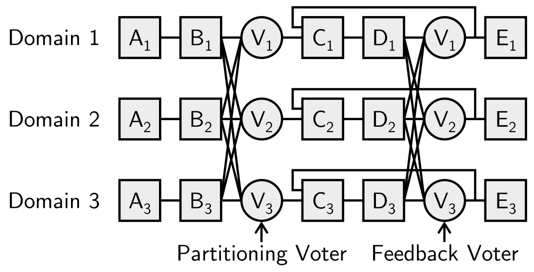

Fig. 2. Diagram of full TMR with partitioning and feedback voters.

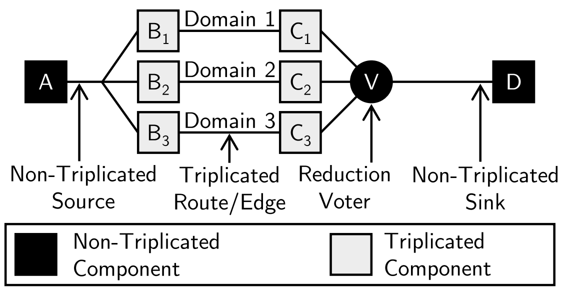

Fig. 3. Diagram of partial TMR with a reduction voter.

Twenty-five different selections are tested on a benchmark design [2] under neutron irradiation [1] and random fault injection [7]. The tested partial TMR selection approaches vary in the number of components selected for TMR and the area of the circuit selected for TMR. The neutron cross section for any observable output error of the design without any TMR is used as a baseline reference point. Changes in cross section range from a 51% increase to a 99% decrease. Benefit–cost ratios range from a 4% increase to a 2% decrease in neutron cross section per component triplicated. Thus, the selection of components for TMR is observed to greatly influence the effectiveness of partial TMR.

II. Partial TMR

In a design that is fully triplicated, each component has triplicate counterparts. An example of a fully triplicated design is shown in Fig. 2. There are three copies of each component (A–E), one copy for each TMR domain. If one of the triplicate counterparts should fail (e.g., A_1), the failure can be masked or hidden by voting on the outputs of all three copies (e.g., A_1–A_3) or their propagated signals. Voters themselves may also be triplicated whenever used for partitioning [8] or self-synchronization [9]. Partitioning allows for errors to exist in multiple TMR domains without defeating TMR so long as the errors are in different partitions [8]. Self-synchronization repairs errors that occur within a single TMR domain inside a feedback loop [9].

SRAM-based FPGAs implement a design using a set of primitive components and connections. These resources are configurable. They consist of items such as lookup tables (LUTs), registers, memory blocks, arithmetic units, clock managers, I/O pads and buffers, high-speed transceivers, and so on. These components form a basis from which any design can be implemented. Numerous connection resources surround components. These configurable connection resources are used to complete connections or routes between instanced components.

Implementing a fully triplicated design in a single FPGA device is challenging. Full TMR requires at more than three times as many resources as a non-triplicated design [9]. Consuming more resources requires more power. Using more resources and inserting voters negatively impacts the propagation time of signals in the device, which makes it harder to meet design constraints in implementation.

In situations where full TMR is not feasible, components within a design can often still be triplicated selectively. Selectively triplicating components means that some design components are triplicated, while others are not. The application of selective or partial TMR allows resources that would otherwise be unused to go toward improving soft error reliability.

When partial TMR is applied to a design, the objective is generally to maximize its benefit–cost ratio. This is done by decreasing the neutron cross section as much as possible for any observable output error while triplicating only a certain portion of components in the original design. The benefit of partial TMR is a reduction in neutron cross section for any observable output error. The cost of partial TMR is the portion of design components that are triplicated.

A simplified diagram of partial TMR is presented in Fig. 3. In this diagram, two of the design components have been triplicated (B and C), while the remaining two components (A and D) have not been triplicated. Additional connections and a reduction voter have been added to support the triple redundancy. Triplicated voters are not included in this study. The output signal of component A drives a component that has been triplicated. Since component A is not triplicated, its output signal is a non-triplicated source. The triplicated output signal of component C drives a reduction voter. The reduction voter transitions the triplicated output signal to a simplex signal that can then be used to drive component D, a non-triplicated component. Since component D is not triplicated, its input signal is a non-triplicated sink.

A route is a collection of edges that all share the same signal source. A route represents a physical connection between a single driver (i.e., a source) and one or more sinks (i.e., terminals driven by a source). A route is able to fan-out from a single source to multiple sinks, whereas an edge includes only a single source–sink pair. Multiple edges that share a common source pertain to the same route. There is one edge for each fan-out of a route.

Edges between triplicated components are also themselves triplicated. An SEU-induced error in any copy of a triplicated edge can occur without defeating TMR so long as only a single copy is in error. In this way, triplicated edges and components are protected from SEU-induced errors. Since multiple edges pertain to the same route, it is possible for a route to be partially triplicated. This occurs when a triplicated source drives triplicated and non-triplicated sinks.

Choosing which components to triplicate and which components not to triplicate in a design is a graph partitioning problem. For any given design with n number of components, there are 2^n possible selections for TMR ranging from the exclusion of all components from TMR to the inclusion of all components in TMR. It is not clear which subsets of components are the most beneficial to triplicate.

This article explores different ways in which components can be selected for TMR automatically without user intervention. These partial TMR selection approaches follow a set of rules that govern the selection of components for TMR. The effectiveness of each approach is measured and compared.

III. Proposed Methodology

In this article, a number of different partial TMR selection approaches are applied to a benchmark design. The resulting variations of the design are tested to determine their neutron cross section and fault injection sensitivity for the occurrence of any output error. Design variants are placed into a specialized test harness that accelerates data collection. The resulting benefit is normalized by the amount of partial TMR applied to determine the effectiveness of the partial TMR selection.

A. Metrics

This article is focuses on mitigating the likelihood of any output error caused by an SEU. This likelihood is measured in two ways. First, it is measured as a neutron cross section. If a high-energy neutron were to path through this hypothetical area [10], an output error would occur. The smaller the area, the less likely an SEU-induced output error is to occur. Second, the likelihood of an output error is measured as a sensitivity to randomly injected faults. This is a percentage of randomly injected faults that result in the occurrence of an observable output error. The smaller the percentage, the less likely observable output errors are to occur. The later measurement serves only to augment the first [7].

The cross section is determined by dividing the total number of observed events by the total measured fluence of exposure [11]. Observed events are the occurrence of any observable output error. Periods from first observance of an output error to a return to normal behavior following a repair or reset are counted as a single event. Ninety-five percent confidence intervals are approximated using conventional methods [11].

Fault injection sensitivity is measured as the percentage of randomly injected faults that result in an observed error event. This percentage follows the maximum likelihood estimator of the binomial distribution. Ninety-five percent confidence intervals are taken using the normal approximation as in [12].

Reduction in neutron cross section and fault injection sensitivity is measured in terms of percentage compared to the cross section and sensitivity of the baseline version of a design respectively as follows:

$$\text{Reduction} = 1 - \frac{\text{variant}}{\text{baseline}}.$$

The neutron cross section and fault injection sensitivity of a design without any TMR serves as a baseline reference point used for comparison purposes. If the cross section or sensitivity resulting from the application of partial TMR is larger than that of the baseline design, a negative reduction percentage is obtained. A negative reduction percentage reflects an increase in neutron cross section or fault injection sensitivity. A negative reduction reflects a negative benefit, which is actually a loss in soft error reliability. Ninety-five percent confidence intervals on reduction percentages are determined using interval arithmetic.

The benefit–cost ratio or *return* is determined by dividing the reduction percentage by the percentage of components included in the partial TMR. A smaller ratio means that less benefit is gained per component triplicated, and a larger ratio means that more benefit is gained per component triplicated. This article demonstrates that some partial TMR selection approaches yield a higher benefit–cost ratio than others.

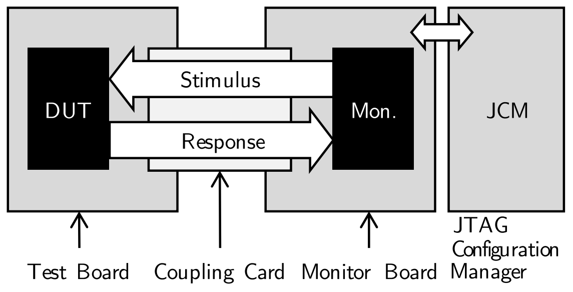

Fig. 4. Test setup.

B. Test Platform

This article uses a test platform that consists of two FPGA development boards paired together by a coupling card [12] as shown in Fig. 4. Both boards are Nexys Video boards with an Artix-7 200T FPGA (XC7A200TSBG484-1). A common clock on the coupling card is provided to both boards. The FPGA design loaded onto the monitor board provides stimulus to the test board and monitors the response of the test board. The design under test (DUT) is loaded onto the test board. A custom JTAG configuration manager (JCM) [13] orchestrates the test and performs CRAM read and write operations.

For radiation testing, the monitor board is placed outside the neutron beam so that its functionality is affected minimally by radiation. For fault injection testing, faults are only injected into the FPGA on the test board. This allows the monitor board to provide stimulus to the test board and compare the response of the DUT against a source of expected golden output values with minimal interference from external sources.

C. Test Flow

The test flow used is shown in Fig. 5. After initialization, three steps are taken successively that repeat until the test is terminated. Initially, the test is brought into a working state, with the DUT producing correct output vectors for a provided set of input vectors. The first step in the flow after initialization is to allow for upsets in the DUT. Upsets occur through either neutron radiation exposure or through fault injection. The second step is to check for errors by reading status registers on the monitor board. The third step is to repair or reset the DUT. During this step, all of the upsets in CRAM are restored to their proper value by the JCM and the DUT is rechecked for

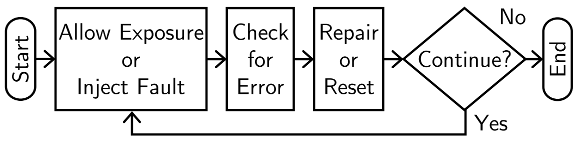

Fig. 5. Test flow.

TABLE I
ITC'99 B13 BENCHMARK RESOURCE UTILIZATION

Resource	FF	LUT-1	LUT-2	LUT-3	LUT-4	LUT-5	LUT-6	Total
Count	56	1	7	12	13	7	4	100

erroneous output. If the DUT remains in error, the DUT is reset by reprogramming the FPGA on the test board.

The same test flow is used for neutron radiation testing and for fault injection testing. During neutron radiation testing, neutron exposure commences with the opening of the beam shutter and continues throughout the duration of the test. Subsequent repetitions of the allowable exposure step merely indicate the passage of time. During fault injection testing, a new fault is introduced each time the inject fault step is executed. In both modes of testing, the monitor board continuously presents the DUT with a set of test vectors and stores the observation of any output errors in a status register. This status register is reset after each check for errors.

D. Test Design

The design selected for this study is a digital circuit from the ITC'99 benchmark suite [14] known as the "B13." It has been included in several studies [2], [4], [15], [16]. Its use in this article is intended to allow for comparison to other work [2]. It consists of a set of interdependent finite-state machines. Implemented on an SRAM-based FPGA, this design consists of LUTs, registers (FFs), and supporting connections. Table I shows a breakdown of the primitive resources used by the B13. This design is simple and provides rich opportunity to explore the reliability impact of various partial TMR selection approaches.

Results obtained from studying this design are insightful, yet caution is advised in their generalization toward all digital circuits implemented on SRAM-based FPGAs. While the B13 contains feedback and feed-forward logic (finite-state machines, counters, pipeline registers, etc.), which are common among digital designs, it is a smaller circuit consisting of only registers and LUTs. Safe generalization of these results to all digital circuits requires additional study. This article is limited to a single benchmark design due to testing constraints and its exploration of many selection variants.

A total of 25 different design variants were tested. Each tested design variant varies in the amount of partial TMR applied and in its partial TMR selection approach. A baseline version of the design without any TMR applied to it was included for use as a comparison reference point. Each of the design variants, their selection amount, and approach are discussed in Section VI.

To accelerate data collection, 256 instances of the same design variant were tested together on the same FPGA such that an output failure in any single copy would trigger a failure event in the monitoring circuit (located on the monitor development board outside the beam or influence of fault injection). The output IO pins and supporting comparison logic are only triplicated in the "Full TMR" design variant. All other design variants include non-triplicated IO and non-triplicated comparison logic surrounding the instances of the DUT.

IV. NEUTRON RADIATION TEST

Neutron radiation testing was chosen for this study because it allows the results to more closely reflect the effectiveness of the evaluated partial TMR selection approaches in a terrestrial environment. Neutron testing is an important part of evaluating the soft error characteristics of FPGAs in terrestrial environments [1]. It is important in this study because the evaluated partial TMR selection approaches may be used in large-scale deployments of commercial FPGA designs in terrestrial environments [17].

Neutron radiation testing for this article was conducted in October of 2019 at the Los Alamos Neutron Science Center (LANSCE) (see Fig. 6). LANSCE provides a spallation neutron source with a neutron energy spectrum that is very similar to a scaled ground spectrum [1]. In this article, the neutron source provided by LANSCE was used to measure the neutron cross section of all of the design variants for any output error. From these measurements, the effectiveness of each partial TMR selection can be compared.

The neutron radiation test setup used for this article is displayed in Fig. 4. A stack of five test platforms were aligned perpendicular to the neutron beam aperture such that the 2-in collimated beam would pass directly through the FPGAs on the test boards. The distances from the tungsten target to the fission chamber and to each test board were used to appropriately attenuate the measured neutron flux for each board in the experiment. The attenuation caused by other boards in the flight path is assumed to be negligible. Five boards were used to accelerate data collection so that statistically significant data could be obtained.

The sum of fluence observed across the five test boards in the beam for the tested designs was 1.16×10^{12} n cm^{-2}. The high-energy (greater than 10 MeV) neutron flux at the fission ion chamber was measured to be 1.11×10^{6} n cm^{2}. Most of the 25 tested design variants were exposed to a neutron fluence of approximately 3.2×10^{10} n cm^{-2} with the DUT running continuously. The baseline design (without any TMR applied) and a few other design variants were exposed for a higher level of fluence (approx. 8×10^{10} n cm^{-2}) to improve 95% confidence intervals. Ninety-five percent confidence level reflects, to a degree, the fluence of exposure.

V. FAULT INJECTION TEST

Random fault injection testing is included in this article to augment the neutron radiation test results [7]. In this article, fault injection emulates SEUs in the target device by purposefully writing corrupt values to CRAM at random locations and on random clock cycles. Corrupt values are written to CRAM via JTAG memory access using the JCM.

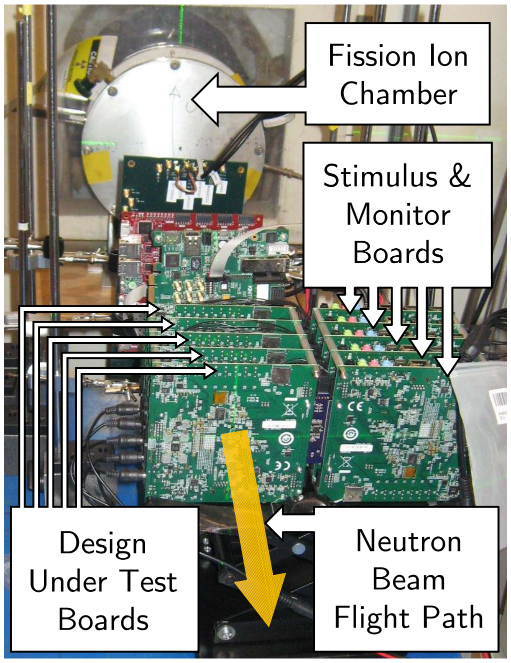

Fig. 6. Neutron radiation test setup.

Injected faults are similarly removed with a subsequent write operation to scrub away the injected corruption. Through random fault injection, 100 000 output errors were observed over approximately 11 million randomly injected faults spread over all 25 tested design variants. On average, approximately 400 000 random faults were tested on each design variant.

The data collected via random fault injection is in statistical agreement with the data collected via neutron radiation testing. Measured values of reliability improvement between the two modes of testing fall within each other's 95% confidence intervals. For some design variants, fault injection data provide more conclusive results. Fault injection data are provided in this article as an additional means of verification.

VI. Results

Several partial TMR selection approaches are explored in this article. Partial TMR selection approaches follow a set of rules to select components for TMR. Components that are selected for TMR get triplicated. Components that are not selected remain non-triplicated. The partial TMR selection approaches explored in this article range in their complexity.

Some selection approaches significantly reduce the cross section or sensitivity of the tested design, while others increase the cross section or sensitivity, making the reliability worse. Some selection modes provided a larger benefit–cost ratio return than others. Table II presents design statistics and test results for each tested design variant. Test results are included for neutron radiation testing and fault injection testing. Related design variants are grouped together.

The first major column in Table II contains design statistics. Design statistics include several attributes. First, the amount of TMR applied or coverage (Cov.) is listed as a percentage of components triplicated under the given selection. Second, the number of triplicated edges (Edg.) is included. The third statistic is the number of reduction voters (Vot.) required to support the selection. The fourth statistic is the number of weakly connected components (WCCs) among the selection. A WCC represents a connected cluster of components. It is thought that efficient selections triplicate more edges while requiring fewer reduction voters.

The second major column in Table II contains neutron radiation testing results. The measured neutron cross section for any output error is given with 95% confidence intervals. The percentage of cross section reduction (Red.) for a given selection is determined by comparing the design variant cross sections against the cross section measurement of the baseline design (i.e., without any TMR). The benefit–cost ratio or return is the quotient of the reduction percentage and coverage percentage. The return indicates the effectiveness of the selection (reduction per percent replicated). The 95% confidence intervals on reduction and return are determined through interval arithmetic.

The final major column in Table II contains fault injection testing results. Fault injection data are presented in terms of sensitivity (Sens.) or the percentage of randomly injected faults that result in an observable output errors. Ninety-five percent confidence intervals are included. The reduction and return metrics in this column carry the same meaning as they do for neutron radiation testing. They apply to the measured sensitivity. Ninety-five percent confidence intervals for these metrics are also calculated using interval arithmetic.

A. Baseline

The first evaluated test variant is the baseline design without any TMR applied to it. Evaluating the cross section and sensitivity of a design without any TMR provides a reference from which all other selections can be compared. Comparing the cross section and sensitivity of design variants against the baseline demonstrates improvement, degradation, or no change. The cross section and sensitivity of design variants may be better or worse than that of the baseline design or there may be no statistical difference based on a 95% confidence interval.

The baseline version of the design is expected to use the least amount of resources. It provides a lower bound on resource utilization and a middle ground for a benefit–cost ratio. No resources are triplicated and no reduction voters are needed. Other selections will consume more resources, but their benefit–ratios can range widely.

B. Full TMR

In this implementation of the B13, all of the components of the B13 were selected for TMR *including* its IO interfaces and the comparison logic that determines if any one of the 256 instances of the design produces an output that disagrees with the other copies. This is the highest level of TMR coverage that can be applied to a design implemented on a single device.

Data for this design variant is taken from [4]. In [4], the same test platform with a similar test design and setup was used in to explore low-level implementation strategies

TABLE II
DESIGN STATISTICS AND TEST RESULTS

Design Statistics					Neutron Radiation			Random Fault Injection		
Design	Cov.	Edg.	Vot.	WCC	Cross Section (cm^2)	Red. (%)	Return	Sens. (%)	Red. (%)	Return
Baseline	0%	0	0	0	$2.6 \pm 0.3 \times 10^{-9}$	–	–	1.06 ± 0.03	–	–
Full TMR	100%	234	9	1	$2.0 \pm 0.8 \times 10^{-11}$	99 ± 1	0.99 ± 0.01	$.002 \pm 0.01$	98 ± 1	0.98 ± 0.01
All Components	100%	234	9	1	$4.1 \pm 1.3 \times 10^{-10}$	84 ± 5	0.84 ± 0.05	0.14 ± 0.01	86 ± 1	0.86 ± 0.01
All LUTs	44%	17	37	28	$1.7 \pm 0.4 \times 10^{-9}$	34 ± 14	0.77 ± 0.32	0.80 ± 0.03	25 ± 5	0.56 ± 0.11
All FFs	56%	24	54	0	$3.9 \pm 1.0 \times 10^{-9}$	-51 ± 37	-0.92 ± 0.66	1.66 ± 0.04	-56 ± 8	-1.01 ± 0.15
Random 9%	9%	3	9	6	$3.5 \pm 0.7 \times 10^{-9}$	-36 ± 25	-3.96 ± 2.78	1.33 ± 0.03	-26 ± 7	-2.86 ± 0.79
Random 20%	20%	9	18	12	$2.6 \pm 0.6 \times 10^{-9}$	-2 ± 22	-0.09 ± 1.08	1.39 ± 0.03	-31 ± 7	-1.57 ± 0.36
Random 38%	38%	32	28	13	$2.7 \pm 0.6 \times 10^{-9}$	-3 ± 22	-0.07 ± 0.57	1.29 ± 0.03	-22 ± 7	-0.58 ± 0.18
Random 50%	50%	60	37	11	$2.5 \pm 0.5 \times 10^{-9}$	5 ± 21	0.09 ± 0.42	1.28 ± 0.03	-21 ± 7	-0.41 ± 0.14
Random 75%	75%	134	39	2	$2.6 \pm 0.6 \times 10^{-9}$	0 ± 21	0.00 ± 0.28	0.82 ± 0.03	22 ± 5	0.30 ± 0.06
Max Edg. / Min Vot. 9%	9%	20	3	1	$2.1 \pm 0.5 \times 10^{-9}$	21 ± 18	2.32 ± 2.00	1.06 ± 0.03	0 ± 6	0.04 ± 0.66
Max Edg. / Min Vot. 20%	20%	60	3	1	$1.8 \pm 0.4 \times 10^{-9}$	32 ± 17	1.58 ± 0.84	0.90 ± 0.03	15 ± 5	0.74 ± 0.26
Max Edg. / Min Vot. 38%	38%	101	2	1	$2.0 \pm 0.5 \times 10^{-9}$	25 ± 18	0.65 ± 0.46	0.71 ± 0.02	33 ± 4	0.86 ± 0.12
Max Edg. / Min Vot. 50%	50%	135	7	1	$1.6 \pm 0.4 \times 10^{-9}$	40 ± 14	0.81 ± 0.28	0.66 ± 0.02	38 ± 4	0.75 ± 0.08
Max Edg. / Min Vot. 75%	75%	191	4	1	$7.6 \pm 2.3 \times 10^{-10}$	71 ± 9	0.94 ± 0.12	0.40 ± 0.02	63 ± 3	0.83 ± 0.04
SCC Largest	69%	163	24	1	$1.4 \pm 0.3 \times 10^{-9}$	46 ± 10	0.67 ± 0.14	0.53 ± 0.02	50 ± 4	0.72 ± 0.05
SCC Output	31%	40	9	7	$2.4 \pm 0.5 \times 10^{-9}$	9 ± 18	0.28 ± 0.58	1.15 ± 0.03	-9 ± 6	-0.28 ± 0.20
TF Level 1	64%	149	18	4	$1.4 \pm 0.3 \times 10^{-9}$	47 ± 12	0.73 ± 0.19	0.65 ± 0.02	38 ± 4	0.60 ± 0.06
TF Level 2	28%	40	14	1	$2.4 \pm 0.4 \times 10^{-9}$	7 ± 17	0.26 ± 0.60	1.08 ± 0.03	-2 ± 6	-0.05 ± 0.16
TF Largest v1	32%	83	14	1	$2.1 \pm 0.4 \times 10^{-9}$	20 ± 17	0.64 ± 0.53	0.84 ± 0.03	20 ± 5	0.64 ± 0.16
TF Largest v2	34%	87	15	1	$2.0 \pm 0.4 \times 10^{-9}$	23 ± 17	0.67 ± 0.50	0.80 ± 0.03	24 ± 5	0.71 ± 0.14
TF 2nd Largest	12%	25	5	1	$2.1 \pm 0.4 \times 10^{-9}$	21 ± 17	1.74 ± 1.42	0.83 ± 0.03	22 ± 5	1.80 ± 0.43
TF Counter	27%	70	5	1	$1.9 \pm 0.4 \times 10^{-9}$	26 ± 14	0.95 ± 0.53	0.85 ± 0.03	20 ± 5	0.73 ± 0.19
TF In-Between	9%	9	6	1	$2.7 \pm 0.5 \times 10^{-9}$	-4 ± 20	-0.40 ± 2.22	1.09 ± 0.03	-3 ± 6	-0.36 ± 0.66
TF Feed Forward	13%	12	1	1	$2.5 \pm 0.5 \times 10^{-9}$	4 ± 19	0.27 ± 1.45	1.17 ± 0.03	-10 ± 6	-0.78 ± 0.49

for improving the effectiveness of full TMR. The full TMR cross section and fault injection sensitivity is taken from [4] without any additional mitigation techniques applied. The full TMR design variant provides an upper bound on the amount of cross section or sensitivity reduction that can be obtained (approximately a 99% reduction in both cases).

C. All Components

In this selection, all of the components in the tested design are triplicated *except* for I/O ports and the logic that compares the output of instances. There are approximately 27 000 components in the baseline design without TMR. This includes all of the used IO ports on the device, the comparison logic, and the components that make up the instances of the B13. The "All Components" selection triplicates the components that make up the B13 instances, which are approximately 25 000 components. Approximately 7% of components in the test design are purposefully excluded from TMR.

In many partial TMR applications, I/O and some additional logic may be fixed and cannot be altered. The "All Components" selection demonstrates the impact of such exclusions. This is related to the significant impact that single-point failures have on the effectiveness of TMR [18]. Reduction in neutron cross section and fault injection sensitivity in this selection is diminished from the 99% reduction that was available from full TMR to approximately 85% reduction.

This selection sets an upper bound on the amount of benefit that can be obtained for the remaining partial TMR selections. All of the remaining selection approaches applied to the B13 exclude I/O and comparison logic from TMR. This selection includes more components in triple redundancy than any of the remaining selections in Table II.

D. All LUTs

This selection triplicates all of the LUTs in the B13 design. LUTs are logic elements that produce a specific output value for a given set of input values. They typically have more input signals than output signals. AN LUT with n-inputs requires at least 2^n CRAM bits to implement, one for each possible binary input combination. Triplicating an LUT also triplicates at least some portions of the route that connect to the LUT.

Replicating only LUTs reduced the cross section and sensitivity by approximately 30%. It provides a positive return. This selection mode required more reduction voters than the number of edges it protected with TMR. These results suggest that more benefit was gained in triplicating only LUTs than was lost by inserting reduction voters.

E. All FFs

Only triplicating FFs and not triplicating LUTs is referred to as local TMR (LTMR) [19]. It is suggested in [19] that this form of TMR will likely perform poorly in SRAM-based FPGAs due to the susceptibility of connections and LUTs in the device to radiation-induced corruptions. FFs require far fewer raw resources to implement than LUTs. Triplicating registers decrease the cross section of the design, but adding reduction voters increases the cross section by adding non-triplicated components in the design. This can make the overall cross section worse. In order for partial TMR to be beneficial for mitigating any output error, the amount of benefit gained by including components in triple redundancy must outweigh the loss of benefit incurred by the addition of reduction voters.

Triplicating FFs only *increased* the neutron cross section and sensitivity by approximately 30%–50%. The benefit–cost ratio return for this selection is negative, meaning that the cross section and sensitivity of the design is better off without triplicating the FFs than with triplicating them by themselves. This selection mode has nearly the same protected edge to reduction ratio as the previous selection. Based on these results, it appears that the benefit gained by triplicating the registers is outweighed by the insertion of reduction voters. These results affirm the suggestion in [19] that LTMR should not be applied to SRAM-based FPGAs.

F. Random

In this selection, components are chosen at random for inclusion in partial TMR. It is anticipated that random selections will likely yield mediocre to subpar performance due to the amount of unrelated components replicated and amount of reduction voters needed to support the replication. When related logic is triplicated together, it likely requires fewer reduction voters than triplicating scattered components.

For this article, five different levels of coverage were explored. Coverage levels were 9%, 20%, 38%, 50%, and 75%. Testing different levels of coverage provides a sense of benefit gained under varying constraints. All levels of coverage for random selection provided little to no benefit in cross section or sensitivity reduction. The smallest level of random coverage increased the cross section and sensitivity by 30%. This is the most negative return of any selection, a very large increase for a very small coverage. The cross sections of higher levels of coverage largely overlapped that of the baseline design, suggesting no change in cross section. Fault injection results suggest an increase in sensitivity for random selection. These results suggest that methodical selection in partial TMR is necessary in order for it to provide benefit and that randomly scattered clusters of triplicated logic is not helpful overall.

G. Maximize Triplicated Edges, Minimize Reduction Voters

This selection strives to triplicate as many edges as possible (i.e., single-bit connections between components) while using fewer reduction voters. It has been found in several studies [20], [21] that more than half of all utilized CRAM bits pertain to routing configuration and that a majority of SEU-induced failures result from SEUs in routing CRAM bits. Triplicating as many connections as possible may disproportionately improve the overall reliability of the design based on the number of components triplicated.

This selection approach solved a set of constraints to minimize the cost of selection based on a cost function. The cost function was set to be the sum of required reduction voters less the sum of triplicated edges. A directed graph of connectivity was generated from the B13 design. Each vertex in the graph represented a component in the design. Each edge in the graph represented a connection in the design. Constraints were set so that a component could be either included or excluded from TMR. Subsequent constraints were set to indicate the need for a reduction of voters and the inclusion of an edge in TMR. An optimizer was used to identify a selection with the minimum cost for the given levels of allowable coverage.

Five different coverage levels are included in this article for maximizing protected routes and minimizing reduction voters. The coverage levels are the same as they are for random selection. These selections require far few reduction voters and triplicate far more edges than the random selection for comparable coverage levels. All of the components in test selections are connected to each other and reduction voters are only needed on the peripheral of the selection.

All of the selections prove beneficial in reducing the neutron cross section. The selection with 9% coverage had the highest benefit–cost ratio return of any selection for neutron cross section benefit; however, fault injection results were much less optimistic for this selection. Reduction for neutron cross section and fault injection sensitivity improve as more components are included in TMR.

H. Feedback Based

This selection approach chooses components for TMR based on feedback relationships between components. Feedback occurs when the future output of a component is dependent on its current output. Previous research has exploited feedback relationships to mitigate the occurrence of persistent errors in SRAM-based FPGAs [6], [22]. This study explores using feedback relationships to guide partial TMR selection for mitigating the occurrence of any output error.

Two main types of feedback relationships are explored. The first type is associated with strongly connected components (SCCs). A directed graph is said to be strongly connected if and only if every vertex is reachable from every other vertex. An SCC is a maximal subgraph that is strongly connected. The second type of feedback is tight feedback (TF). TF occurs when a signal's next state depends on its current or previous state within only a few clock cycles. TF can occur as nested feedback within a larger feedback structure. Fig. 7 presents a simple example of tight nested feedback. There is TF from FF_1 back to itself nested inside a larger, more loose feedback structure. Nested feedback decomposition (identifying feedback nesting within a circuit) can help identify groups of related logic components.

The final two sets of selection approaches are based on feedback analysis of the test design. The analysis performed begins with a connectivity graph, which is a mathematical representation of the test design where each vertex in the graph

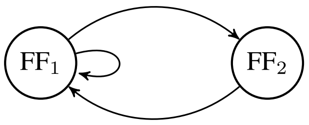

Fig. 7. Nested feedback.

represents a component and each directional edge represents a connection between two components. This graph is decomposed to aid the different selection approaches.

The SCC-based selections use an SCC decomposition of the connectivity graph where each SCC in the graph is replaced with a single representative vertex. SCCs were identified using Tarjan's SCC algorithm [23]. The "SCC Largest" selection applies TMR to the SCC in the graph with the greatest number of subcomponents. The "SCC Output" selection triplicates scattered logic driven by the largest SCC.

The TF-based selections use a decomposition of the connectivity graph that is designed to create a hierarchical organization of nested feedback. This decomposition first eliminates non-sequential vertices by replacing them with edges between associated sequential vertices. This creates a sequential connectivity graph. This graph is then further decomposed using a modified depth-limited breadth-first search to identify groups of components within TF (components in a closed walk with a minimal number of edges). Each group of components is replaced with a single representative vertex. The decomposition continues until all TF has been folded into a hierarchical organization of nested feedback. The details and variants of this decomposition are beyond the scope of this article. For the purpose of this article, the decomposition provides an alternative view of feedback within the test design.

Several different selections were made based on TF relationships. Fig. 8 depicts the "TF Counter" selection. This nested feedback group contains all of the sequential registers and combinational logic (LUTs) associated with a 7-bit counter used in the test design. Each bit is contained in its own feedback group with a return distance of one (i.e., there exists a self-containing closed walk with a single edge). Each bit within the larger feedback group can reach at least one other bit with a return distance of two. The "TF Level 1" selection contains all sequential registers with a return distance of one and associated combinational logic (i.e., LUTs that drive and are driven by the selected registers). The "TF Level 2" selection contains registers and associated combinational logic with a return distance of two. The "TF Largest v1" selection contains a sizable feedback group, and the "TF Largest v2" selection contains the same feedback group with the addition of a nearby register and LUT. The "TF 2nd Largest" selection contains the next smallest sizable feedback group. The "TF In-Between" selection contains sequential registers and associated combinational logic that are in between TF groups. Finally, the "TF Feed Forward" selection contains any components not contained in a feedback group that are downstream from TF (i.e., driven by components that are themselves contained in TF). These selections were made to explore a variety of different selection approaches based on TF.

Promising results were found among the selections based on feedback. The "SCC Largest" selection provided a positive

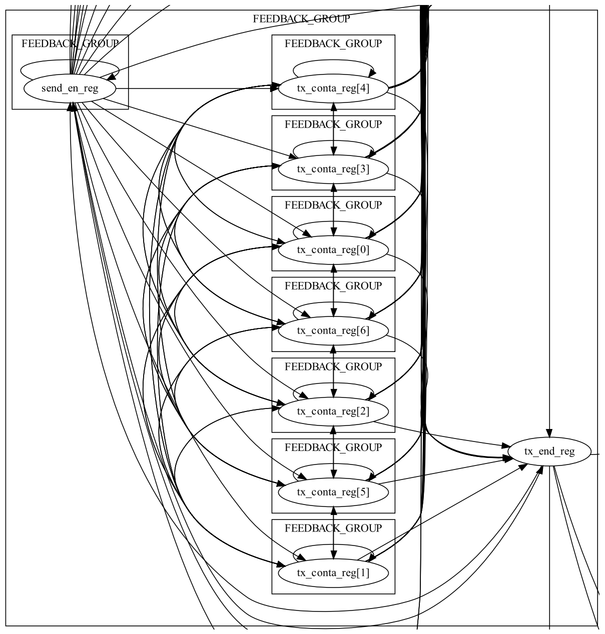

Fig. 8. TF counter selection.

return and a significant reduction. The largest "Max Edg./Min Vot." selection still provided better results at a marginal increase in coverage. The "SCC Output" selection yielded subpar results. The TF set of selections yield the best results when the selections include a single cluster of logic containing TF (as opposed to logic in between TF, etc.). Selections that include large clusters of feedback and selections that include clusters of TF show promise for reducing the overall likelihood of failure in a design.

VII. Related Work

Many selection approaches have been proposed for partial TMR [5], [6], [24]–[26]. A mode of selection is a set of rules by which a portion of the design is selected for triple redundancy. In general, the objective of a selection mode is to maximize provided benefit while maintaining some predefined constraints. The desired benefit ranges from minimizing persistent errors through exploiting feedback relationships [6], to limiting the percentage of clock cycles in which the outputs of a design are in error through iterative analysis of attribute-based selections [5], to minimizing the likelihood of error propagation based on logic masking probabilities [24], [25], to minimizing the arithmetic severity of SEUs through most significant bit (MSB) inclusion [26], and so on. This study seeks to minimize the occurrence of any observable SEU-induced output error using previously untested selection approaches.

VIII. Conclusion

Partial TMR, or the application of TMR to a subset of design components, can provide some of the reliable benefits of TMR at a reduced cost. This is helpful for situation where applying TMR to all design components may not be feasible. This study explores the application of partial TMR for improving the overall soft error reliability of an SRAM-based FPGA design. Twenty-five different design variants were tested, each with differing amounts of partial TMR and selection methodologies.

It was found that the selection of components for partial TMR significantly impacted its effectiveness. Replicating all components except I/O reduced the cross section by 85%. Replicating all combinational logic improved the cross section with a high return (34% reduction for 44% coverage) despite having a large number of independent TMR regions and inserted voters. Only replicating registers increased the cross section by 50%. Random selection proved to be counter-productive. Maximizing the number of triplicated routes and minimizing the number of reduction voters clustered component selection and performed well. Triplicating larger clusters of related logic tended to perform the best at improving the overall soft error reliability of the design. One selection, which included a single group of closely related feedback logic, reduced the neutron cross section by 21% while triplicating only 12% of the components in the design.

REFERENCES

[1] *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

[2] 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.

[3] A. Manuzzato, S. Gerardin, A. Paccagnella, L. Sterpone, and M. Violante, "Effectiveness of TMR-based techniques to mitigate alpha-induced SEU accumulation in commercial SRAM-based FPGAs," in *Proc. 9th Eur. Conf. Radiat. Effects Compon. Syst.*, Sep. 2007, pp. 98–104.

[4] M. Cannon, A. Keller, and M. Wirthlin, "Improving the effectiveness of TMR designs on FPGAs with SEU-aware incremental placement," in *Proc. IEEE 26th Annu. Int. Symp. Field-Program. Custom Comput. Mach. (FCCM)*, Apr. 2018, pp. 141–148.

[5] O. Ruano, J. A. Maestro, and P. Reviriego, "A methodology for automatic insertion of selective TMR in digital circuits affected by SEUs," *IEEE Trans. Nucl. Sci.*, vol. 56, no. 4, pp. 2091–2102, Aug. 2009.

[6] B. Pratt, M. Caffrey, J. F. Carroll, P. Graham, K. Morgan, and M. Wirthlin, "Fine-grain SEU mitigation for FPGAs using partial TMR," *IEEE Trans. Nucl. Sci.*, vol. 55, no. 4, pp. 2274–2280, Aug. 2008.

[7] 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.

[8] W. Xin, "Partitioning triple modular redundancy for single event upset mitigation in FPGA," in *Proc. Int. Conf. E-Product E-Service E-Entertainment*, Nov. 2010, pp. 4213–4216.

[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 Program. Gate Arrays (FPGA)*, 2010, pp. 249–258.

[10] N. Battezzati *et al.*, "Failure modes and analysis," in *Programmable Gate Arrays for Mission-Critical Applications*. New York, NY, USA: Springer, 2011, pp. 37–83.

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

[12] C. Thurlow, H. Rowberry, and M. Wirthlin, "TURTLE: A low-cost fault injection platform for SRAM-based FPGAs," in *Proc. Int. Conf. ReConFigurable Comput. FPGAs (ReConFig)*, Dec. 2019, pp. 238–245.

[13] A. Gruwell, P. Zabriskie, and M. Wirthlin, "High-speed FPGA configuration and testing through JTAG," in *Proc. IEEE AUTOTESTCON*, Sep. 2016, pp. 218–225.

[14] 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.

[15] A. J. Sanchez-Clemente, L. Entrena, and M. Garcia-Valderas, "Partial TMR in FPGAs using approximate logic circuits," *IEEE Trans. Nucl. Sci.*, vol. 63, no. 4, pp. 2233–2240, Aug. 2016.

[16] N. Battezzati *et al.*, "Putting mitigation techniques at work," in *Reconfigurable Field Programmable Gate Arrays for Mission-Critical Applications*. New York, NY, USA: Springer, 2011, pp. 187–204.

[17] H. Quinn and P. Graham, "Terrestrial-based radiation upsets: A cautionary tale," in *Proc. 13th Annu. IEEE Symp. Field-Program. Custom Comput. Mach. (FCCM)*, Apr. 2005, pp. 193–202.

[18] M. J. Cannon, A. M. Keller, C. A. Thurlow, A. Perez-Celis, and M. J. Wirthlin, "Improving the reliability of TMR with nontriplicated I/O on SRAM FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 67, no. 1, pp. 312–320, Jan. 2020.

[19] M. Berg. (2019). *FPGA Mitigation Strategies for Critical Applications*. [Online]. Available: https://ntrs.nasa.gov/citations/20190033455

[20] 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.

[21] L. Sterpone and M. Violante, "A new partial reconfiguration-based fault-injection system to evaluate SEU effects in SRAM-based FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 54, no. 4, pp. 965–970, Aug. 2007.

[22] K. Morgan, M. Caffrey, P. Graham, E. Johnson, B. Pratt, and M. Wirthlin, "SEU-induced persistent error propagation in FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 52, no. 6, pp. 2438–2445, Dec. 2005.

[23] R. Tarjan, "Depth-first search and linear graph algorithms," *SIAM J. Comput.*, vol. 1, no. 2, pp. 146–160, Jun. 1972.

[24] S. Baloch, T. Arslan, and A. Stoica, "Probability based partial triple modular redundancy technique for reconfigurable architectures," in *Proc. IEEE Aerosp. Conf.*, Mar. 2006, pp. 2946–2952.

[25] P. K. Samudrala, J. Ramos, and S. Katkoori, "Selective triple modular redundancy (STMR) based single-event upset (SEU) tolerant synthesis for FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 51, no. 5, pp. 2957–2969, Oct. 2004.

[26] B. Pratt, M. Fuller, M. Rice, and M. Wirthlin, "Reduced-precision redundancy for reliable FPGA communications systems in high-radiation environments," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 49, no. 1, pp. 369–380, Jan. 2013.