Improving the Reliability of TMR With Nontriplicated I/O on SRAM FPGAs

Matthew J. Cannon, *Student Member, IEEE*, Andrew M. Keller, *Student Member, IEEE*, Corbin A. Thurlow, Andrés Pérez-Celis, *Student Member, IEEE*, and Michael J. Wirthlin, *Senior Member, IEEE*

***Abstract*— Triple modular redundancy (TMR) with repair is a commonly employed mitigation strategy used on SRAM field-programmable gate arrays (FPGAs) to reduce the effects of ionizing radiation and improve a circuit's sensitive cross section. This article examines TMR circuits, where the I/O ports of the circuit have not been triplicated, but the internal circuitry has. Such circuits introduce single-point failures (SPFs) into the circuit that limit the neutron cross-sectional improvement offered by TMR to only 3× for the b13 benchmark circuit used in this article. This article proposes two different mitigation techniques to address SPFs, which alter the placement and routing of the circuit. These mitigation techniques reduce the neutron cross section by 26× over the unmitigated circuit while minimally affecting the circuit's maximum clock frequency and resource utilization.**

***Index Terms*— Configuration scrubbing, fault injection, field-programmable gate arrays (FPGAs), radiation testing, single-point failure (SPF), single-event effects (SEEs), single-event upset (SEU), triple modular redundancy (TMR).**

I. Introduction

SRAM field-programmable gate arrays (FPGAs) are integrated circuits (IC) that can implement any digital logic function, given enough resources. An FPGA consists of lookup tables (LUTs), flip-flops (FFs), block memories (BRAMs), and other special resources (digital signal processors (DSPs), multigigabit transceivers (MGTs), and so on) coupled with a large configurable routing network to programmatically connect all of these resources. The FPGA also consists of a large memory (SRAM cells for an SRAM FPGA) called the configuration memory (CRAM), which defines the operation of all of the resources and the routing network on the device [1].

FPGAs are being increasingly considered for use in many harsh environments, such as in space, high-energy physics experiments, and high-altitude environments. While in these environments, FPGAs are exposed to ionizing radiation and subject to numerous types of single-event effects (SEEs), such as single-event upsets (SEUs), single-event transients (SETs), single-event latchup (SEL), and single-event function interrupts (SEFIs) [2], [3]. SRAM FPGAs are particularly sensitive to SEUs, as SEUs typically occur in the large CRAM (approaching 1 Gb on the newest devices) potentially changing the circuit's implementation [4], [5]. SEUs within the CRAM could flip an LUT value which could change the circuits functionality or occur in the routing network and connect/disconnect physical wires on the device.

Triple modular redundancy (TMR) with repair is a commonly employed strategy to mitigate against the effects of SEUs on FPGAs [6]. In TMR, the circuit is replicated three times and voters are inserted on the outputs so that only the majority logic value, as voted on by the three redundant circuits, is propagated from the device. By only propagating the majority value, any failures limited to one of the redundant circuits are masked. Applying TMR with repair to an entire circuit (including triplicating all the I/O ports) has shown to greatly increase the mean time to failure (MTTF) of the circuit by 50–100× [7], [8] and, in this article, is shown to improve the neutron cross section by 86×.

However, [7], [8], and other works study the impact of TMR with repair when the circuit has been completely triplicated. When the circuit is completely triplicated, all I/O ports are triplicated along with all of the internal resources, including any BRAMs, MGTs, or DSPs that are used by the circuit. In many applications, such triplication may not be possible. Whenever an unmitigated circuit utilizes more than 1/3 of any one resource, that resource cannot be triplicated.

Even when a circuit cannot be completely triplicated, the designer may still want to improve a circuit's radiation sensitivity [9]. When TMR is applied in these scenarios, the designer introduces single-point failures (SPFs) into the circuit that will limit how much TMR can improve a circuit's radiation sensitivity. SPF occurs in the regions of the circuit that are untriplicated, as failures in these regions will not be masked by a majority voter.

Not triplicating the I/O ports (common-IO) is common for FPGA circuits, as FPGAs are generally employed in a system to interface with many devices or because the device interfacing with the FPGA does not have I/O triplication.[1]

Manuscript received July 5, 2019; revised September 28, 2019 and November 20, 2019; accepted November 22, 2019. Date of publication December 4, 2019; date of current version January 29, 2020. This work was supported in part by the I/UCRC Program of the National Science Foundation under Grant 1738550, in part by the Utah NASA Space Grant Consortium (UNSGC), and in part by the Los Alamos Neutron Science Center (LANSCE) under Grant NS-2017-7574-A, Grant NS-2017-7574-F, and Grant NS-2018-7895-A.

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

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

Digital Object Identifier 10.1109/TNS.2019.2956473

[1] When the I/O on an FPGA is triplicated, the I/O device connecting to those ports must be able to read/write the triplicated I/O.

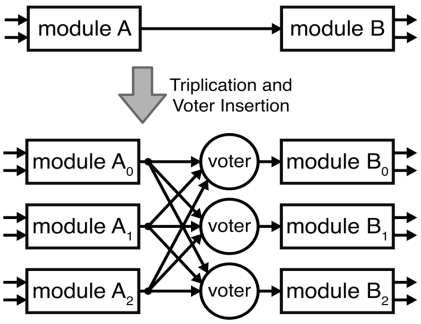

Fig. 1. Triple modular redundancy.

In the common-IO version of TMR for the b13 circuit studied in this article, the neutron cross section improved only 3× over the unmitigated version, which was much less than 86× cross-sectional improvement observed for the triplicated I/O (trip-IO) TMR version of the circuit.

The goal of this article is to develop and validate mitigation techniques for SPF to reduce the sensitive cross section in TMR circuits, where the I/O ports are not triplicated. This article develops two separate mitigation techniques called split-clock and split-IO. These techniques mitigate SPF by altering the placement and routing of the circuit. By applying these techniques to a Xilinx Artix-7 200T FPGA, the neutron cross section improves by 26× over the unmitigated design, about an order of magnitude improvement over the base common-IO design.

II. Background

TMR can be used to triplicate the circuitry within the FPGA, as shown in Fig. 1. When a failure occurs in one of the TMR domains, it is limited to that domain as the redundant copies will successfully mask the failure. TMR is usually applied to the circuit using automated tools by directly modifying the circuit's netlist. The tool we use for this article is called the BL-TMR tool [10].

To increase a circuit's MTTF, a repair element is needed in addition to TMR. Repairing a system refers to bringing the broken module into a correct operating state after it has failed and then resynchronizing it with the other modules. Depending on the repair speed relative to the failure rate, the TMR with a repair system can greatly improve the reliability and MTTF of a circuit.

There are two components to implementing repair for an FPGA circuit. The first step is to restore upset CRAM bits to their original value to "repair" the original circuit functionality. This is done through configuration scrubbing, where a scrubber continually corrects the original bitstream to its proper value [11]. There are a variety of ways to implement configuration scrubbing, including blind scrubbing with a golden copy, readback scrubbing with an external scrubbing circuit, or the use of an error-correcting code (ECC).

The second component is to provide repair for the dynamic memory elements in the circuit, such as the FFs and BRAMs, called resynchronization. For memory that is written every cycle (such as FFs), TMR voters can be added along the feedback paths, which will automatically resynchronize the memory as soon as the CRAM is scrubbed, called feedback TMR [12]. ECCs can also be used for larger memories [13].

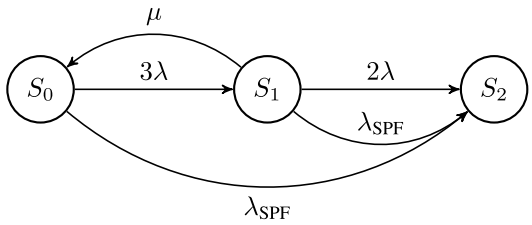

Fig. 2. TMR with an SPF reliability model.

A. Related Work

Despite adding TMR with repair to SRAM FPGAs, upsets within some single CRAM bits have been shown to cause circuit failures even when the design is completely triplicated [14]–[17]. These bits that affect the operation of more than one TMR domain are referred to as common-mode failures (CMFs). In these previous works, the authors improve TMR by altering the placement and routing. Sterpone and Violante [14] proposed the reliability-oriented place and route algorithm (RoRA). This group measured their improvement to be on the order of 350–650× during fault injection. The same group created the VERI-Place tool, which also includes an estimator to predict a circuit's sensitivity to SEUs [18]–[20]. Cannon *et al.* [16], [17] proposed mitigation techniques called striping and PCMF (used in this article) and showed a neutron cross-sectional improvement up to 350×.

The main difference between this article and previous works is that this article seeks to mitigate SPF in TMR systems with nontriplicated I/O ports. The goal of this article is to make TMR with repair improve the cross section as much as possible, despite the limitation imposed by SPF. Because SPF will always be part of the system, the improvements measured in this article will never be as good as the improvements seen when the circuit is completely triplicated.

B. Reliability Modeling TMR With SPF

Mathematical models are often used to represent the reliability of systems and potential fault-tolerant techniques. Markov chains are useful because reliability metrics such as reliability as a function of time and the MTTF can be derived [21]. The Markov chain for a TMR with repair system and SPF can be modeled, as shown in Fig. 2. This is the same chain that was used in [17]. In the model, λ represents the failure rate of one of the redundant TMR circuits, μ represents the circuit repair rate, and λ_{SPF} represents the SPF rate.

The MTTF and maximum improvement can be calculated using the model [21]

$$\text{MTTF} = \frac{5\lambda + \lambda_{\text{SPF}} + \mu}{6\lambda^2 + 5\lambda\lambda_{\text{SPF}} + \lambda_{\text{SPF}}^2 + \mu\lambda_{\text{SPF}}}.$$

Assuming a repair rate that is much greater than the redundant circuit failure rate (i.e., $\mu \gg \lambda$) yields

$$\text{MTTF} = \frac{1}{\lambda_{\text{SPF}}}.$$

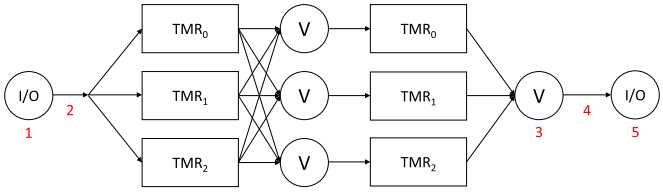

Fig. 3. Example of the TMR FPGA circuit with SPF in the I/O.

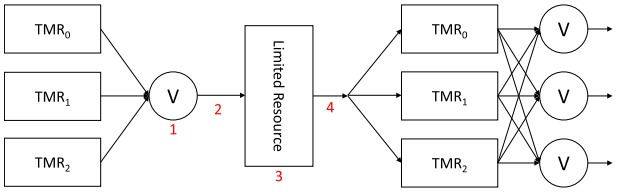

Fig. 4. Example of the TMR FPGA circuit with SPF in a limited resource.

Thus, the MTTF of this system is limited by its SPF rate. By reducing the cross section of the SPF regions of the circuit, the reliability and MTTF of the system can be improved.

III. Single Point Failures

For the purpose of this article, an SPF is defined to be an upset in any one bit in the CRAM that causes TMR failure, where that one bit corresponds to an untriplicated component/resource in the circuit. By the definition of SPF, SPF can only occur in a TMR circuit that is not completely triplicated. Thus, the only way to completely remove SPF is to completely triplicate the circuit. Even though the presence of SPF places a limit on the total cross-sectional improvement TMR can offer, it can still be partially mitigated. SPF mitigation techniques can be developed by understanding how SPF occurs.

Fig. 3 shows the example of a TMR circuit with untriplicated I/O pins on an FPGA. There are five locations in this diagram, where an SPF can occur, labeled 1–5. Two of the five locations are the I/O themselves (#1 and #5). Two other locations occur on the routes leading to/from the I/O (#2 and #4). The net coming from the input I/O is a single route before it eventually splits to feed each individual TMR domain. Then, when going to the output I/O, there is a single route from the reduction voter to the output. The final location is the single reduction voter (#5). Because there is only a single output, the output of the three TMR domains needs to be reduced to a single net for the device output. This is done through the use of a single voter.

SPF can also occur when other resources in the circuit are not triplicated, such as when these designs use a single resource. Such resources could be the MGTs, DSPs, or BSCAN. Like I/O, there are relatively a few of these resources on the FPGA compared with the relatively large number of LUTs and FFs. If the untriplicated design uses more than 1/3 of these resources, then they all may not be triplicated. An example of such a circuit is shown in Fig. 4. There are four locations in the limited resource circuit, where SPF can occur: in the single voter (#1), in the route leading to the resource (#2), in the resource (#3), and in the route leading from the resource (#4).

Fault injection [22], [23] can be used to test a circuit's sensitivity to single-bit upsets (SBUs) and each bit that caused a failure can be analyzed to determine what low-level architecture was affected. There are several types of failures.

- *Routing Failure:* A failure within the routing network. This can occur either from the input pad or going to the output pad from the reduction voter. A routing failure can further be split into a nonclock routing failure or a clock routing failure. It is useful to make this distinction on an FPGA, as the routing for nonclock and clock signals is distinct and uses separate resources.
- *Voting Failure:* A failure that occurs within the reduction voter that is used to drive a single signal output of the chip. This occurs in the LUT that is used for the voting logic.
- *CMF:* A CMF as explained in [17].
- *Other Failure:* Any other form of failure that is unknown.

Using these failure types, all the failures from the fault-injection results for the common-IO TMR version can be classified and are presented in Table I (the results from the fault injection test can be found in Section V).

These results show that routing failures, both nonclock and clock, are by far the largest cause of SPF. This is because input nets can have a large fan-out, making them more susceptible. It is possible to map the location of failures onto the physical design implementation to visualize what part of the circuit is being affected by the failure. Fig. 5, for example, shows an input pin that has a large fan-out and red dots that show the location of several failures observed during fault injection on the b13 circuit. As the figure shows, this particular route is quite sensitive. There are many more nets similar to this one in the circuit, which make the circuit quite sensitive. Similarly, output nets are also sensitive to SPF, as shown in the figure.

TABLE I
SPF in Common-IO TMR Circuits

Circuit	Routing Failures: Non-Clock Failures	Routing Failures: Clock Failures	Voter Failures	CMF	Other Failures	Total
b13	1,941	491	0	103	256	2,791
md5	65	197	0	30	83	375
sha3	52	13	0	3	28	96
aes	51	13	0	3	38	105
Total	2,109	714	0	139	405	3,367

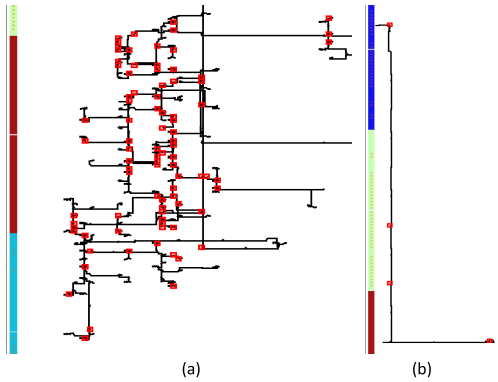

Fig. 5. Data nets in the b13 circuit (black lines) with failures found during fault injection mapped on top (red squares) showing the sensitivity of these nets. (a) High fan-out input pin net. (b) Reduction output net.

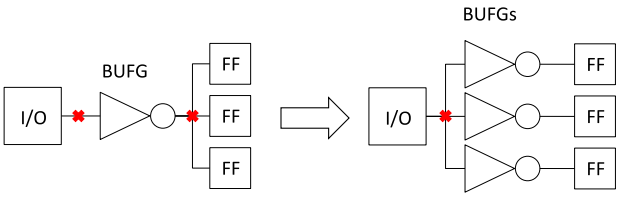

Fig. 6. Triplicating clocks.

Voter failures occur much less frequently, because there are fewer configuration bits that control the voting logic. Since voters take three input signals (from the three domains), it would be implemented as an LUT3 or a three-input LUT. Only eight bits are needed to implement an LUT3, meaning there are eight potential sensitive bits.

IV. Mitigation Strategies

In an FPGA circuit, each I/O pin can either be a clock net or a nonclock net. Each type of net needs to be treated differently due to the different resources used to route them through the device. We developed two techniques to mitigate SPF: split clock and split-IO. These techniques are all implemented as tool command language (TCL) scripts. TCL can be used in the vendor's CAD tool for low-level manipulation of a circuit.

A. Split Clock

An input pin for a clock is first routed through a clock buffer, as shown in Fig. 6. Buffers are used to drive the clock signal onto the clock-specific routing network and to minimize the skew across the clock network. There are different clock buffers available on the device, which are used to drive different regions on the device. The most common type of buffer is a global buffer (BUFG). In a typical design, the clock would be routed through a BUFG, or similar buffer, and then routed to each cell in the design.

In a common-IO TMR design, there would only be one clock signal for each clock in the original design. To internally triplicate each clock, they can each be routed to three buffers, one for each domain. Then, the output of each buffer is routed to the cells associated with that particular clock and TMR domain. This mitigation technique is shown in Fig. 6.

As shown in the figure, there are a few locations where SPF can occur in the original design, denoted by the red "x"s. The first location is from the input pin to the buffer and the second location is from the buffer to the cells of each domain. While the figure only shows one red "x," it is likely that each of these nets uses multiple wire segments, which would correspond to multiple configuration bits. After buffer triplication, there is only one location where SPF can occur, on the net from the input pin to each of the buffers. The possibility for SPF has been removed from the design implementation after the buffers.

B. Split-IO

Similar to split clocks, nonclock input pins can also be split, as shown in Fig. 7. The net routes from the input pin through any number of wire segments before it splits to feed each TMR domain. As the figure shows, SPF can occur before the net splits denoted with the red "x"s. SPF can be mitigated by splitting the net early so that SPF is limited to occurring near the input pin.

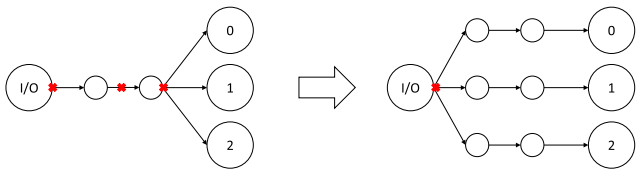

Fig. 7. Split I/O technique.

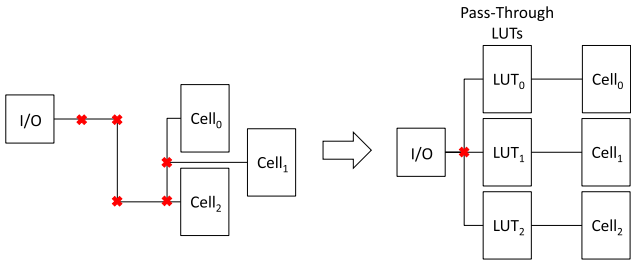

Fig. 8. Pass-through LUTs.

Splitting the nonclock nets must be performed differently than splitting the clock nets. Unlike clock nets which pass through a buffer before being routed to the cells, nonclock nets are routed directly from the input pin to each corresponding cell. To mitigate SPF, the net should be split as early as possible. Just how early the net is able to split is dependent on the device architecture as there may be local routing that needs to be performed before the net can be routed onto the general routing network.

There are two steps to implementing the split-IO technique within the vendor tool. The first step is to introduce a pseudobuffer into the netlist. This pseudobuffer can be inserted for each separate domain to force the net to split. This pseudobuffer is implemented as a pass-through LUT, which is a single-input LUT, whose function is to copy the logic value of the input wire onto the output wire. The split-IO technique using pass-through LUTs is shown in Fig. 8. The second step to the split-IO technique is to constrain these pseudobuffer LUTs to be placed in the CLB tile closest to the input pin.

Implementing the split-IO technique can impact device utilization in two ways. First, there is a need for three additional LUTs to serve as the pseudobuffers. This would increase the number of LUTs by three times the number of logic input pins in use for a given circuit. Second, this would increase the routing utilization and congestion. Without the pseudobuffers, the tool is free to split the net as close to the sink cells as it would like. With the pseudobuffers in place, the tool must route three copies of the net to the sink cells. The increase in routing congestion would be dependent on the number of sink cells for the net and how close they are placed to the pseudobuffers on the device.

C. Output Placement

Mitigating SPF for an output net is more difficult to perform. This is because the net is coming from the redundant circuitry

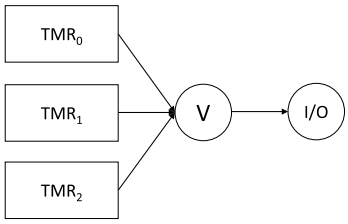

Fig. 9. Untriplicated output net.

and needs to be reduced to a single output wire. TMR ceases at the reduction voter, which is used to propagate the majority value from the TMR domains to the output pin, as shown in Fig. 9. Anywhere from the reduction voter to the output pin could fail from SPF. The easiest strategy is to ensure that the path from the reduction voter to the output pin is as small as possible. This is done by placing the reduction voter in the CLB tile closest to the output pin, just like the pseudobuffers were placed in the CLB tile closest to the input pin. No additional cells are needed to implement this technique. For the results, output placement will be combined with split-IO.

D. Early Split (ES)

All these techniques, split clock, split-IO, and output placement, can be used together. When used together, they are referred to as ES. When strictly considering SBUs, these mitigation techniques would be most effective when used together. However, certain missions/environments may dictate otherwise. For example, there may be environments where parts of the FPGA may be more susceptible to failure than other parts. Or, there may be enough resources to implement split-IO, but there may not be enough resources to implement split clock.

E. Combined With CMF Mitigation Techniques

Both PCMF and striping mitigation techniques developed in [17] can be used in tandem with the SPF mitigation techniques to increase the reliability of common-IO TMR. To use the PCMF technique, the clock *must* be triplicated since the failure mechanism involves multiple clocks. While striping can be used without any SPF techniques, doing so has little practical sense, as the failure cross section will usually be dominated by SPF. Before applying any CMF technique, the most effective SPF techniques should first be applied.

V. Reliability Testing and Analysis

Four different benchmark circuits were used to evaluate the techniques described in the previous section. The first of the circuit, the b13, design comes from the ITC'99 benchmark suite and is a simple finite-state machine that interfaces with a weather station. We chose the relatively small b13 circuit because this circuit is commonly used as a benchmark by a number of researchers investigating the effects of radiation on FPGA designs [20]. Ideally, results from this article can be compared to others if the same benchmark circuit is used. The circuit is replicated 256× because it is so small. If only a single instance was used, then we would have much less failure data (and thus worse statistics). In addition, the common effects of I/O and other global issues can be spread across multiple b13 instances rather than a single instance giving a more accurate per instance result. It was selected as the only circuit tested in radiation due to the limited availability of beam time.

Three other circuits were also used for this article: the md5, sha3, and aes128. These other circuits are different types of hashing and encryption algorithms. They have also been replicated to increase the circuit utilization. All the circuits and all the TMR variations of the circuits were clocked at 50 MHz.

In total, 12 different TMR variations were tested for each circuit.

- *Unmitigated:* The original circuit with no mitigation techniques applied to it.
- *Common-IO:* The original circuit with TMR applied to it but without any of the input or output pins being triplicated. This design is further subdivided into one-voter and three-voter designs, where only one voter is used for each partition or where three voters are used for every partition.
- *Split-IO:* The common-IO three-voter circuit with the split-IO mitigation technique and output placement technique applied.
- *Split Clock:* The common-IO three-voter circuit with the split-clock mitigation technique applied.
- *Split-Clock-PCMF:* The split-clock circuit with PCMF also being applied.
- *ES:* The common-IO three-voter circuit with the split-IO, split clock, and output placement techniques applied.
- *ES-PCMF:* The ES circuit with PCMF also being applied.
- *Trip-IO:* The original circuit with TMR applied to it, where all the input and output pins are triplicated. Like the common-IO circuits, these are also subdivided into one-voter and three-voter designs.
- *PCMF:* The trip-IO circuit with PCMF applied [17].
- *Striped:* The trip-IO circuit with striping applied [17].

The last two TMR variations are CMF mitigation strategies specifically for trip-IO TMR circuits. These are included to show the tradeoffs involved when choosing between common-IO and trip-IO TMR circuits. Routing was not able to complete for the striping technique for the md5 and sha3 circuits.

A. Performance Analysis

Five metrics were chosen to study the implementation impacts of each of the techniques. The first metric is the maximum achievable frequency for the circuit. This metric helps measure how each technique impacts the ability to meet timing. The second metric is the number of routing nodes. The number of routing nodes is the number of electrical nodes on the FPGA device that are driven by nets in the circuit. This metric directly relates to power consumption and routing congestion, which impact routeability. The third, fourth, and

TABLE II
MEAN IMPLEMENTATION METRICS

Metric/Technique	fmax (MHz)	# nodes	# cells	# sites	# tiles
Unmitigated	1×	1×	1×	1×	1×
Common-IO (1-Voter)	0.77×	3.48×	3.16×	3.75×	3.62×
Common-IO (3-Voter)	0.73×	3.66×	3.44×	3.73×	3.56×
Split-IO	0.80×	3.69×	3.44×	3.67×	3.55×
Split-clock	0.73×	3.70×	3.44×	3.83×	3.68×
Split-clock-PCMF	0.73×	3.76×	3.44×	3.83×	3.69×
ES	0.73×	3.68×	3.44×	3.67×	3.55×
ES-PCMF	0.73×	3.78×	3.44×	3.76×	3.68×
Trip-IO (1-Voter)	0.79×	3.62×	3.17×	3.86×	3.74×
Trip-IO (3-Voter)	0.77×	3.89×	3.44×	3.87×	3.75×
PCMF	0.76×	3.95×	3.44×	3.87×	3.77×
Striped	0.78×	4.20×	3.51×	3.46×	3.24×

TABLE III
BASE CIRCUIT UTILIZATION

Resource/Circuit	LUT	LUTRAM	FF	I/O
B13	11,460	0	13,056	37
MD5	25,598	2,063	31,354	37
SHA3	12,550	2	5,688	37
AES128	32,529	0	12,699	37

fifth metrics are all area measurements. They measure the number of cells, sites, and tiles used by the circuit.

We developed a TCL script to find the maximum frequency. Once the maximum frequency is found, the number of routing nodes, cells, sites, and tiles is reported for the maximum frequency implementation. The geomean change (over the unmitigated design) across all circuits can be found in Table II, while the baseline utilization (the number of LUTs, LUTRAMs, FFs, and I/O) is shown in Table III. For the geomean change, a higher change is better for the maximum frequency, while a lower change is better for all of the other metrics.

As Table II shows, the geomean maximum clock frequency is not affected by the proposed SPF mitigation techniques (over the common-IO three-voter design). Of course, the individual circuit results do vary. The md5 circuit was negatively affected, while the other circuits were positively affected in the maximum clock frequency. The other metrics were also slightly affected by the SPF mitigation techniques. This table also shows that the SPF mitigation techniques have a minimal impact on the circuit size. This means that if the designer is already willing to pay the price for common-IO TMR (three voters), then applying the SPF mitigation techniques has minimum additional impact on the circuit.

B. Experimental Setup

For the reliability testing (fault injection and neutron radiation testing), this article uses a golden copy with the device under test (DUT) structure for testing. In this setup, there are two different chips, the golden copy and the DUT, both running the same copy of the circuit. The only difference between the two chips is that the DUT is exposed to errors, whether that be through fault injection or radiation testing, while the golden copy is not subject to any errors. The boards are run in lockstep so that failures in the DUT can be detected in real time. All detection logic is implemented on the golden copy, so that all failures occur from the circuit on the DUT.

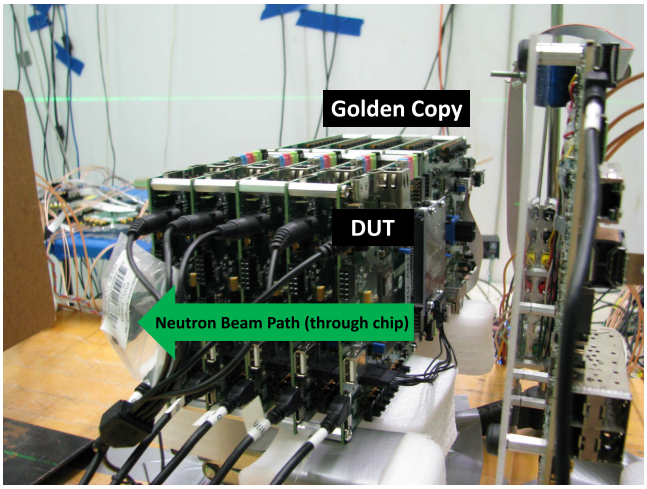

Fig. 10. TURTLE setup.

A setup called the TURTLE is used for this article. The boards contain an Artix-7 200T FPGA chip, shown in Fig. 10. In the TURTLE setup, the golden copy and DUT are implemented on separate boards, the Nexys Video Artix-7 boards available from Digilent. The boards are connected via an FMC coupler card, which handles all communications between the boards. Configuration and scrubbing are handled through the JTAG port using the JTAG configuration manager (JCM), which was designed to handle these tasks [24].

The TURTLE setup is designed to be stacked on top of each other to aid in data collection. Fig. 10 shows the standard five-board stack that is used for the majority of the tests. This allows data to be collected approximately 5× faster than normal, as five fault-injection tests can be running simultaneously and five experiments will run simultaneously during neutron radiation testing. The collection of more data allows for more event observations, which tightens the confidence intervals.

C. Fault-Injection Results

All the circuits and TMR variations were tested using fault injection [22], [25]. During our fault-injection experiments, faults were randomly injected into any bit from the type 0 logic frames of the device. This fault was allowed to propagate for 1 ms and then scrubbed, and if no failure was detected, fault injection continued by selecting a new random bit.

To understand the relative value of each technique, the total number of injections and detected failures is reported in Table IV. This table also reports the geometric mean for the techniques sensitivities and their improvement. The goal of the geometric means is to provide an idea for which techniques are usually more effective for any general circuit. The specific results for each benchmark circuit and mitigation technique are shown in Fig. 11.

The main takeaway from this table and plot is that the SPF mitigation techniques do reduce the circuit's sensitivity to SBUs. In addition to the main takeaway, there are a few general trends from the plot and table.

- The ES-PCMF mitigation technique offered the most improvement for common-IO TMR circuits.

TABLE IV
FAULT-INJECTION MEAN RESULTS

	Metric/ Technique	Number of Injections	Number of Failures	Sensitivity	Improvement
Mean Results	**Unmitigated**	$6,045,542$	$330,450$	3.80×10^{-2}	$1\times$
	Common-IO (1-Voter)	$6,037,318$	$16,967$	1.56×10^{-3}	$24.3\times$
	Common-IO (3-Voter)	$6,132,724$	$3,367$	3.16×10^{-4}	$121\times$
	Split-IO	$8,000,000$	$5,447$	8.45×10^{-5}	$450\times$
	Split-clock	$6,538,320$	778	1.13×10^{-4}	$336\times$
	Split-clock-PCMF	$6,336,939$	486	8.45×10^{-5}	$450\times$
	ES	$9,255,262$	313	2.27×10^{-5}	$1,675\times$
	ES-PCMF	$18,000,000$	377	1.63×10^{-5}	$2,340\times$
	Trip-IO (1-Voter)	$16,000,000$	$50,836$	4.83×10^{-4}	$78.8\times$
	Trip-IO (3-Voter)	$28,000,000$	225	2.09×10^{-6}	$18,235\times$
	PCMF	$21,541,693$	49	9.92×10^{-7}	$38,341\times$
	Striped	$4,000,000$	0	5.00×10^{-7}	$57,443\times$

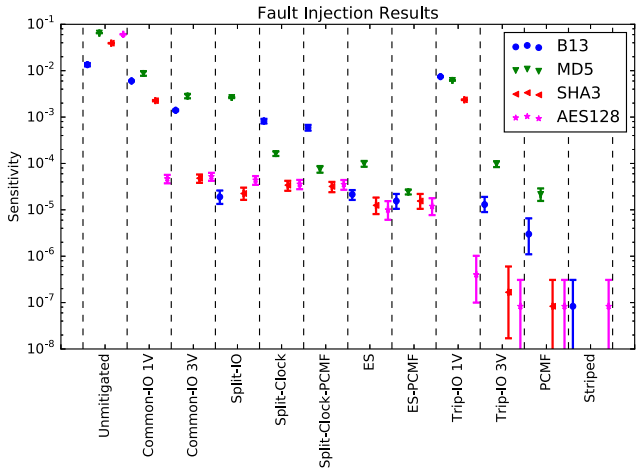

Fig. 11. Fault-injection sensitivity of designs/circuits.

- The split-IO and split-clock-PCMF mitigation techniques offer about the same improvement for common-IO TMR circuits.

Since the fault-injection results for the mitigation techniques are promising, the next step is to validate these mitigation techniques through radiation testing.

D. Radiation Testing Results

Radiation testing can be used to measure the response of a circuit (such as an FPGA) in the presence of ionizing radiation [26], [27]. Neutron radiation testing took place at the Los Alamos Neutron Science Center (LANSCE) in Ice House I and II [28]. The beam at LANSCE is a wide-energy neutron beam that approximates the terrestrial environment, but at much higher flux. Neutrons with an energy greater than 10 MeV are counted toward the fluence.

For the radiation test, the TURTLE was setup incident to the beam and run at room temperature. Additionally, the beam was collimated to 2 in. so that only the DUT FPGA and a few surrounding components were exposed to the full beam flux. TMR was applied to the golden copy circuit, since the golden copy device is close to the beam. Additionally, when a failure is detected, the golden copy is scrubbed so a posttest analysis can filter out any failures that can be attributed to the golden copy (no failures on the golden copy were observed for this

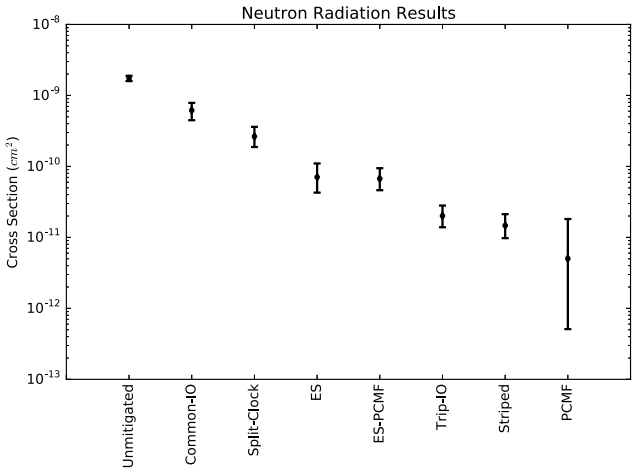

Fig. 12. Neutron cross section of designs.

experiment). This helps to ensure that observed failures are due to the DUT.

The results from the LANSCE neutron tests are shown in Table V and are graphically plotted in Fig. 12. Since beam time is limited, only the b13 circuit and only some of the mitigation techniques were tested.

The major takeaway from beam testing is that the mitigation techniques improve the cross section of the circuit. The improvements in radiation testing are not as high as they were in fault injection, but this is because SBUs are not the only cause of circuit failure. In addition, there are several other takeaways from the data.

- The common-IO circuit only showed a $3\times$ improvement over no mitigation.
- The ES-PCMF was the best mitigation technique for common-IO circuits with an improvement of $26\times$.
- The trip-IO circuit performed better than all the mitigation techniques for common-IO circuits.
- PCMF performed better than striping in radiation testing (as was explored in [17]).

E. Radiation Test Analysis

To further understand the failures observed during radiation testing, the radiation test logs can be used to perform a "replay" using fault injection. During each iteration, the JCM logs any faults in the configuration memory and whether or not the DUT failed. This process repeats until the end of the radiation test.

Using the information provided by the logs, each scrub cycle where a failure occurred can be "replayed." A radiation replay is a form of fault injection, but instead of a single bit being injected during each iteration, the upset bits logged during a scrub cycle are all injected during each iteration. If a failure occurs during the replay, then the failure in that scrub cycle was caused by SEUs in the configuration memory. When a failure does not occur during the replay, then the failure was caused by other radiation effects.

If the replay determines that the cause of failure was due to SEUs in the configuration memory, then fault injection can be

TABLE V
NEUTRON RADIATION TESTING RESULTS

TMR Type	Fluence (n/cm^2)	Number of Failures	Cross-Section (cm^2)	+95% Confidence -95% Confidence	FIT Sea-Level	Improvement	Number of Failures (Filtered)	Improvement (Filtered)
Unmitigated	3.19×10^{11}	555	1.74×10^{-9}	1.89×10^{-9} 1.59×10^{-9}	2.26×10^{1}	1×	551	1×
Common-IO	8.61×10^{10}	51	5.9×10^{-10}	7.6×10^{-10} 4.3×10^{-10}	7.7×10^{0}	2.9×	48	3.1×
Split-clock	1.48×10^{11}	39	2.6×10^{-10}	3.6×10^{-10} 1.9×10^{-10}	3.4×10^{0}	6.6×	37	6.9×
ES	2.69×10^{11}	19	7.1×10^{-11}	1.1×10^{-10} 4.3×10^{-11}	9.2×10^{-1}	25×	16	29×
ES-PCMF	4.91×10^{11}	33	6.7×10^{-11}	9.4×10^{-11} 4.6×10^{-11}	8.7×10^{-1}	26×	27	31×
Trip-IO	1.69×10^{12}	34	2.0×10^{-11}	2.8×10^{-11} 1.4×10^{-11}	2.6×10^{-1}	86×	30	97×
PCMF	3.98×10^{11}	2	5×10^{-12}	2×10^{-11} 5×10^{-13}	7×10^{-2}	400×	1	700×
Striped	1.90×10^{12}	28	1.5×10^{-11}	2.1×10^{-11} 9.8×10^{-12}	1.9×10^{-1}	120×	22	150×

Note: FIT rates calculated using sea-level neutron flux level of 13 n/cm^{2}h according to JESD89A standard [29].

done at a finer granularity. When fault injection is performed at a finer granularity, every combination of upset bits can be injected to determine which subset caused the failure.

When multiple bits are determined to be the cause of a failure, then further analysis can be used to determine if the multiple bits are the result of an MCU or multiple SEUs. If an MCU is the cause of the failure, then the circuit failed from a *single* event. However, if multiple SEUs (SEU accumulation) are the cause of a failure, then the circuit failed from *multiple*, independent events.

Determining the difference between SEU accumulation and MCUs is difficult as logically adjacent bits are not necessarily physically adjacent on the device. Statistical methods can be employed to determine which upsets have a high probability of being an MCU. This is done by using the aggregate data across the entire test to observe common upset patterns. Due to the random nature of the test, most upset patterns should only occur a few times, and thus, patterns that occur many times are highly unlikely to be random. These patterns can be used to identify MCUs for every scrub cycle. This is still an active area of research, but a tool built on these principles has been employed for this article [30].

The results of the radiation replay are presented in Table VI. This table shows how many failures can be attributed to SBUs, MCUs, and SEU accumulation or other for each TMR design. Other upsets were likely caused by errors in unmonitored areas of the device, such as the user memory. As expected, applying the SPF and CMF mitigation techniques reduces the impact of SBUs with the striped and PCMF designs showing no failures due to SBUs.

The cross sections from the radiation test can be updated to filter out failures that were caused by SEU accumulation. SEU accumulation is much more likely to happen in the accelerated testing environment because upsets are happening at an accelerated pace. This increases the likelihood of observing multiple independent events in the same scrub cycle. In a deployed environment where the repair rate easily outpaces the bit upset rate, SEU accumulation is much less likely. The filtered results and their improvements are included in Table V. Using the filtered results, the common-IO circuit shows a 3.1× cross-sectional improvement and the ES-PCMF technique shows a 31× cross-sectional improvement.

TABLE VI
CLASSIFICATION OF FAILURES FROM RADIATION TEST

Design/ Failure Type	SBU	MCU	Accumulation	Not Repeatable
Unmitigated	202 (36.4%)	4 (.7%)	4 (.7%)	345 (62.2%)
Common-IO	20 (39.2%)	7 (13.7%)	3 (5.9%)	21 (41.2%)
Split-Clock	9 (23.2%)	7 (17.9%)	2 (5.1%)	21 (53.8%)
ES	1 (5.3%)	6 (31.6%)	3 (15.8%)	9 (47.4%)
ES-PCMF	3 (9.1%)	7 (21.2%)	6 (18.2%)	17 (51.5%)
Trip-IO	2 (5.9%)	15 (44.1%)	4 (11.8%)	13 (38.2%)
Striped	0 (0.0%)	18 (64.3%)	6 (21.4%)	4 (14.3%)
PCMF	0 (0.0%)	1 (50.0%)	1 (50.0%)	0 (0.0%)

VI. CONCLUSION AND FUTURE WORK

Mitigating SPFs for TMR circuits on FPGAs has shown significant improvement. Neutron testing has shown that the SPF mitigation techniques can provide up to a 26× reduction in the sensitive cross section over the unmitigated design, which translates to about an order of magnitude improvement over TMR without triplicated I/O. These techniques will be beneficial to the design engineer weighing the tradeoffs between circuit utilization, power, and reliability.

These techniques have been shown for the Xilinx 7-Series generation of FPGAs. We are currently planning future work to use these techniques on other generations, such as the Xilinx UltraScale FPGAs, and to test these mitigation techniques on other benchmark circuits. Future mitigation strategies will explore ways to address the effects of MCUs. We would

also like to explore the possibility of using custom placement and/or routing to perform some of these mitigation techniques.

REFERENCES

[1] M. L. Chang, "Device architecture," in *Reconfigurable Computing: The Theory and Practice of FPGA-Based Computation*, vol. 1, S. Hauck and A. Dehon, Eds. Amsterdam, The Netherlands: Elsevier, 2008, ch. 1, pp. 16–18.

[2] R. Katz *et al.*, "Radiation effects on current field programmable technologies," *IEEE Trans. Nucl. Sci.*, vol. 44, no. 6, pp. 1945–1956, Dec. 1997.

[3] M. Wirthlin, "FPGAs operating in a radiation environment: Lessons learned from FPGAs in space," *J. Instrum.*, vol. 8, no. 2, Feb. 2013, Art. no. C02020, doi: 10.1088/1748-0221/8/02/c02020.

[4] P. Graham, M. Caffrey, J. Zimmerman, P. Sundararajan, and E. Johnson, "Consequences and categories of SRAM FPGA configuration SEUs," in *Proc. Int. Conf. Mil. Aerosp. Program. Log. Devices*, 2003, pp. 1–10.

[5] 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.

[6] 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.

[7] 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.

[8] M. J. Wirthlin, A. M. Keller, C. McCloskey, P. Ridd, D. Lee, and J. Draper, "SEU mitigation and validation of the LEON3 soft processor using triple modular redundancy for space processing," in *Proc. ACM/SIGDA Int. Symp. Field-Program. Gate Arrays*, New York, NY, USA, 2016, pp. 205–214.

[9] F. Lima, L. Carro, and R. Reis, "Designing fault tolerant systems into SRAM-based FPGAs," in *Proc. 40th Annu. Des. Automat. Conf.*, New York, NY, USA, 2003, pp. 650–655, doi: 10.1145/775832.775997.

[10] B. Pratt, M. Caffrey, P. Graham, K. Morgan, and M. Wirthlin, "Improving FPGA design robustness with partial TMR," in *Proc. IEEE Int. Rel. Phys. Symp.*, San Jose, CA, USA, Mar. 2006, pp. 226–232, doi: 10.1109/RELPHY.2006.251221.

[11] J. Heiner, B. Sellers, M. Wirthlin, and J. Kalb, "FPGA partial reconfiguration via configuration scrubbing," in *Proc. Int. Conf. Field Program. Log. Appl.*, Aug./Sep. 2009, pp. 99–104.

[12] 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*, New York, NY, USA, 2010, pp. 249–258.

[13] N. Rollins, M. Fuller, and M. J. Wirthlin, "A comparison of fault-tolerant memories in SRAM-based FPGAs," in *Proc. Aerosp. Conf.*, Mar. 2010, pp. 1–12.

[14] L. Sterpone and M. Violante, "A new reliability-oriented place and route algorithm for SRAM-based FPGAs," *IEEE Trans. Comput.*, vol. 55, no. 6, pp. 732–744, Jun. 2006.

[15] 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.

[16] 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./May 2018, pp. 141–148.

[17] M. J. Cannon, A. M. Keller, H. C. Rowberry, C. A. Thurlow, A. Pérez-Celis, and M. J. Wirthlin, "Strategies for removing common mode failures from TMR designs deployed on SRAM FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 66, no. 1, pp. 207–215, Jan. 2019.

[18] L. Sterpone *et al.*, "Experimental validation of a tool for predicting the effects of soft errors in SRAM-based FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 54, no. 6, pp. 2576–2583, Dec. 2007.

[19] M. Desogus, L. Sterpone, and D. M. Codinachs, "Validation of a tool for estimating the effects of soft-errors on modern SRAM-based FPGAs," in *Proc. IEEE 20th Int. On-Line Test. Symp. (IOLTS)*, Castell-Platja d'Aro, Girona, Jul. 2014, pp. 111–115.

[20] 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.

[21] R. S. Swarz and D. P. Siewiorek, "Evaluation criteria," in *Reliable Computer Systems: Design and Evaluation*, 3rd ed. Natick, MA, USA: AK Peters, 1998, ch. 5, pp. 334–336.

[22] J. A. Clark and D. K. Pradhan, "Fault injection: A method for validating computer-system dependability," *Computer*, vol. 28, no. 6, pp. 47–56, Jun. 1995.

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

[24] A. Gruwell, P. Zabriskie, and M. Wirthlin, "High-speed FPGA configuration and testing through JTAG," in *Proc. IEEE AUTOTESTCON*, Sep. 2016, pp. 1–8.

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

[26] D. S. Lee, M. Wirthlin, G. Swift, and A. C. Le, "Single-event characterization of the 28 nm Xilinx Kintex-7 field-programmable gate array under heavy ion irradiation," in *Proc. IEEE Radiat. Effects Data Workshop (REDW)*, Jul. 2014, pp. 1–5.

[27] D. S. Lee, "Single-event characterization of the 20 nm Xilinx Kintex UltraScale field-programmable gate array under heavy ion irradiation," in *Proc. IEEE Radiat. Effects Data Workshop (REDW)*, Jul. 2015, pp. 1–6.

[28] P. W. Lisowski, C. D. Bowman, G. J. Russell, and S. A.Wender, "The Los Alamos national laboratory spallation neutron sources," *Nucl. Sci. Eng.*, vol. 106, no. 2, pp. 208–218, 1990.

[29] *Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray Induced Soft Errors in Semiconductor Devices*, JEDEC Standard JESD89A, JEDEC solid state technology association, Oct. 2006.

[30] M. Wirthlin, D. Lee, G. Swift, and H. Quinn, "A method and case study on identifying physically adjacent multiple-cell upsets using 28-nm, interleaved and SECDED-protected arrays," *IEEE Trans. Nucl. Sci.*, vol. 61, no. 6, pp. 3080–3087, Dec. 2014.