Statistical Method to Extract Radiation-Induced Multiple-Cell Upsets in SRAM-Based FPGAs

Andrés Pérez-Celis, *Student Member, IEEE*, and Michael J. Wirthlin, *Senior Member, IEEE*

***Abstract*— Radiation-induced multiple-cell upsets (MCUs) are a concern because they can overcome the protection of error correction code and triplicated designs. Extracting MCU data from radiation tests is helpful to perform more accurate fault injection tests, where MCUs could be simulated with the injection of bits based on the MCUs shapes, sizes, and frequencies. This article presents a statistical method to extract MCU shapes and frequencies from components with no information regarding their physical layout. The proposed method can be used to extract MCU information from BRAM and CRAM alike. The results show the MCU data for three families of Xilinx field-programmable gate arrays (FPGAs).**

***Index Terms*— Field-programmable gate arrays (FPGAs), multiple-bit upset (MBU), multiple-cell upset (MCU), radiation testing, single-event effects (SEEs).**

I. Introduction

ELECTRONIC circuits are susceptible to radiation-induced effects known as single-event effects (SEEs) [1]. These events occur when a particle strikes the circuit transferring some of its energy. The transferred energy can be sufficient to cause a change in the state of a memory element [2]. This change of state is known as a single-event upset (SEU) and can cause a variety of problems in electronic circuits.

As transistors shrink, the possibility that an SEU upsets multiple cells increases [3]. This could be a result of charge collection in adjacent transistors that are physically closer to each other [4]. With transistors decreasing in size, a charged particle is more likely to generate charge in an area that will allow more than one transistor to collect some of the charges. Other characteristics that cause more than one cell to upset with the interaction of a single particle are the fabrication process [5] and the indirect charge collection in the transistor wells [6].

SEEs that affect more than one cell are known as multiple-cell upsets (MCUs). MCUs are a concern for error correction codes (ECCs) since they affect several bits of a word overcoming the ECC protection [7]. They also affect triplicated designs—MCUs have been shown to be the leading cause of failures exhibited on circuits protected with triplicated hardware redundancy [8]. Evaluating the impact that MCUs have on an application is needed.

Extracting MCU data from accelerated radiation tests can be used to fulfill this goal. Namely, MCU data can be used to perform more accurate fault injection tests, where MCUs could be simulated by injecting bits based on the MCUs shapes, sizes, and frequencies. Thus, MCU data can be used to evaluate memory devices, such as field-programmable gate arrays (FPGAs), with fault injection tests. These tests will closely simulate the results of an accelerated radiation test.

This article presents an improved method to extract the MCUs distribution of SRAM FPGAs from radiation data with no information about the physical layout. This method extends the previous work in [9] with three specific contributions. First, this article introduces an improved technique for extracting MCUs with prefiltering data, providing a larger search window, and uses a statistical approach for identifying MCU patterns. Second, this improved technique has been used to extract MCU data from a neutron radiation test data with three different FPGA families: the Xilinx 7-Series, (28 nm), the UltraScale (22 nm), and UltraScale+ (16 nm). Third, this technique was applied to measure MCUs within the configuration memory (CRAM) of all three FPGA families and the block memory (BRAM) of the 7-Series FPGA.

II. Background

A static radiation test for devices containing a dense array of a static memory (such as an SRAM or an FPGA) typically involves two primary steps. First, the device under test (DUT) is irradiated to induce SEUs within the memory array. Second, the radiation beam is stopped, and the contents of the memory state are read back from the test apparatus. The contents of the memory are compared against the known memory pattern to determine how many memory cells (n) were upset during the radiation phase. The sensitive cross section of the memories can then be estimated by dividing the number of upset memory cells by the total radiation fluence (number of particles per cm^2) applied during the test

$$\sigma = \frac{n}{\text{fluence}}. \tag{1}$$

In addition to computing the memory cell cross section, static radiation tests typically collect and record a list of specific memory cells that have been upset during a test run. This list

Manuscript received October 22, 2019; revised November 18, 2019; accepted November 18, 2019. Date of publication November 21, 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 and in part by the Los Alamos Neutron Science Center under 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) (e-mail: pcelis@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.2019.2955006

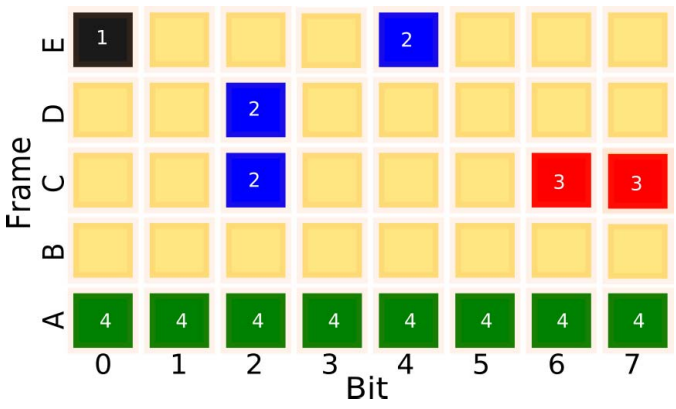

Fig. 1. Examples of the types of upsets experienced in an SRAM FPGA. The black cell is an SCU, the blue cells show an MCU, the two red cells are a CSEU, and the green cells represent a micro-SEFI.

of upset cells can be analyzed to obtain more insight into the behavior of the device in a radiation field. For example, this list can be analyzed to measure the radiation sensitivity of different memory cell types within the device, identify the device-specific failure modes, or extract the location-specific behavior within the device.

The list of upset cells from a radiation test can also be used to extract multicell upsets. If the physical layout of the device is known, MCUs can be easily extracted from this list by grouping upset memory cells that are physically adjacent to the device. If the physical layout of the device is not known, MCUs extraction is more difficult and involves statistical and pattern analysis of the upset data [7], [9]. The primary goal of the MCU extraction process described in this article is to analyze a list of individual cell upsets that occur within a radiation test experiment and decide which of them, if any, upset together as a part of a single-event multicell upset.

Extracting MCUs from radiation test data cannot be done by simply finding cells that are "logically" adjacent. Memory cells are uniquely identified through the use of a logical address that specifies the row, column, frame, bit, or other device-specific characteristic manner of organizing the memory cells. Although a logical address provides a unique identifier for each cell in the device, this logical address does not necessarily provide any insight into the physical location of the cell or its relationship to other logically adjacent neighbors.

For the Xilinx FPGAs used in this study, the logical address of a configuration memory cell (CRAM) and block memory cell (BRAM) is specified by its frame number and bit number within the given frame. The number of bits per frame varies between devices from different families.[1] Fig. 1 demonstrates a figurative 2-D logical address space of the configuration memory of a Xilinx FPGA with frames indicated as rows and bits as the columns.

Most of the cells that upset in a radiation test are single-cell upsets (SCUs). These upsets involve a single particle causing a single cell to upset (represented by the black cell labeled "1" in Fig. 1). These upsets involve one and only one cell and have no impact on other cells in the device. Some events in the radiation test will upset multiple cells from a single particle and are called multicell upsets or MCUs. An example of an MCU event is shown by the blue cells labeled "2" in Fig. 1. This example demonstrates that the bits associated with this single-event MCU are not necessarily logically adjacent.

Fig. 1 demonstrates two other upset types that make it more difficult to infer MCUs from the radiation test data. The first upset type is called coincident SEUs (CSEUs) and occurs when two independent SCUs are found in logically adjacent or nearby locations (represented by the red cells labeled "3" in Fig. 1). While CSEUs are rare, the probability of occurrence increases significantly as the number of upset cells in the radiation test data increases. The presence of CSEUs will negatively affect the quality of the MCU inference.

The second type of upsets is called micro-Single-Event Functional Interrupts (micro-SEFIs). A micro-SEFI is an SEU within a single bit that indirectly causes multiple-related cells to change their value. For example, upsetting a register that controls the reset signal of a multibit register will result in many cells appearing in the radiation test upset data. An example of a micro-SEFI is represented by the eight green cells labeled "4" in Fig. 1. Micro-SEFI events have been observed in SRAM FPGAs in the form of BRAM column upsets [10] and full LUT contents upsets [11]. The presence of micro-SEFIs within the radiation test will skew the radiation test data by introducing false upset patterns. Failing to identify and disregard micro-SEFIs could lead to inaccurate identification of MCUs.

III. Overview of MCU Extraction Process

The method proposed in this article to extract MCUs from a list of unique upset bits involves identifying common patterns on the positional difference of upsets. These common patterns are then combined to identify statistically likely MCUs from radiation test data. This method consists of the following steps.

1) Gather SRAM upset data.
2) Filter contamination on the data.
3) Compute the offsets between upsets and generate a histogram of offsets.
4) Select the most-common offsets.
5) Reconstruct MCUs based on most-common offsets.

Each step of this method is described in more detail within the remainder of this section.

A. Data Collection

The first step of our statistical method is to collect upset data from static designs. During this step, we continuously perform a readback of the device and compare it to a golden copy to identify upsets on the configuration memory (CRAM) and BRAM cells. Then, upsets on the CRAM are scrubbed, while upsets on BRAM are updated in the golden copy to avoid reporting duplicated upsets. The time it takes to perform a readback, a comparison with the golden copy, and the correction of the upsets is known as a scrub cycle.

[1]The Xilinx 7-Series FPGA has 3232 b/frame, the Ultrascale FPGA has 3936 b/frame, and the Ultrascale+ has 2976 b/frame.

TABLE I
UPSET INFORMATION FROM LANSCE NEUTRON TESTING

Upset ID	Scrub cycle	Frame	Word	Bit
1	13	0x0C8F36	5	16
2	13	0x1040693	44	23
3	13	0x11807C4	67	22
1	14	0x11409A0	10	16
1	15	0x1140067	61	25
2	15	0x1180B3C	70	15

The collected data show the logical addresses where the upsets occurred. Table I shows an example of the data collected from a neutron test performed at the Los Alamos Neutron Science Center (LANSCE) in December 2018. The data are comprised of the frame number, the word, the bit, the current scrub cycle, and an ID for each upset within a scrub cycle. Having the data separated by the scrub cycle makes it easier to manage since an MCU cannot span across multiple scrub cycles because an MCU is caused by a single particle that upsets the content of more than one memory cell.

It is important to ensure that the fewest number of upsets occur during each scrub cycle [12]. As the number of upsets per scrub cycle increases, the probability of having CSEUs also increases. Conversely, the probability of a CSEU decreases as the size of the configuration memory increases. We can compute the probability of a CSEU occurring in a scrub cycle based on the average number of upsets per scrub cycle, the size of the memory array, and the size of the CSEU. The expression to compute the probability of a 2-b CSEU is

$$P(\text{CSEU}_2) = \frac{2}{N-1} \times \binom{u}{2} \tag{2}$$

where $P(\text{CSEU}_2)$ is the probability of at least one given CSEU of size two, N is the number of cells in the memory array, which could be BRAM or CRAM, and u is the average number of upsets per scrub cycle.

For example, the probability of having a given size-2 CSEU $[P(\text{CSEU}_2)]$ on the CRAM of the Artix-7 200t device is 3.4×10^{-8}, assuming that on average the device upset two times every nonempty scrub cycle, which lasts 0.25 s, and considering that the Artix-7 200t has 59 145 600 bits[2] on its configuration memory (CRAM). $P(\text{CSEU}_2)$ can be seen as the probability that u upsets will generate a size-2 CSEU in a grid with N cells. The same approach can be used to compute the probability of size-2 CSEU for BRAM upsets.

B. Data Filtering

Wirthlin *et al.* [9] did not describe any type of filtering applied to their data; however, the process of filtering data is needed to remove contamination caused by micro-SEFIs. In the presence of a micro-SEFI, readbacks can show hundreds or thousands of upsets per cycle. These scrub cycles have little information on MCUs and should be disregarded in the remainder of the process.

[2]The number of CRAM bits was obtained by performing a readback of each type-0 and type-1 frame. A total of 18 308 type 0 frames were identified. Out of those, eight were dummy frames. Finally, the remaining number of frames (18 300) was multiplied by 3232, which is the number of bits per frame.

TABLE II
PROBABILITY OF EVENTS IN A SCRUB CYCLE WITH $\lambda = 2$

x	$P(x)$
0	0.135
1	0.270
2	0.270
3	0.180
4	0.090

Since beam testing is a random process, any memory location has the same probability of having an upset. Experiments of such nature follow a Poisson distribution [13]. The probability of an event that follows the Poisson distribution is:

$$p(x) = \frac{\lambda^x e^{-\lambda}}{x!} \tag{3}$$

where x is an integer of the number of events, $P(x)$ is the probability of exactly x events happening, and λ is the mean of the distribution or expected number of events.

Our approach to filtering the data comes from computing the number of expected upsets per scrub cycle. This number can be computed multiplying the number of upsets per second by the seconds in a scrub cycle. Then, the expected number of upsets per scrub cycle is used as the mean of a Poisson distributed process. The Poisson distribution shows the probability of having x number of upsets on a scrub cycle.

Consider the following example, where an FPGA experiences two upsets per scrub cycle, and the probabilities for x events happening in a scrub cycle are shown in Table II. As the expected number of upsets increases, the probability of such a number occurring decreases. This is important because it gives us statistical support to identify anomalies, including micro-SEFIs, on scrub cycles that report a large number of upsets.

Using this calculation, it is possible to identify scrub cycles corresponding to outliers, i.e., scrub cycles with an improbable number of upsets. If the data are not filtered, SEFIs could hide the actual patterns of MCUs. Using this filtering process, we found that the XCZU9EG device experienced several micro-SEFIs in the configuration memory that would have disrupted the identification of MCUs (refer to Section VI for more details). The cross section for these events is reported in this article.

C. Computation of Offsets

The offset computation begins with the translation of the upset information into a coordinate system. The proposed method looks into the whole memory array, instead of only computing offsets in a restricted window as performed in [9]. Our proposed coordinate system consists of (x, y), where x is the frame number in decimal and y is computed as $32 \times \text{word} + \text{bit}$. The method then iterates through each upset computing the positional difference, i.e., offset, for each pair of upsets within the same scrub cycle. Table III shows an example with neutron data for the XCKU040 device.

TABLE III
UPSET INFORMATION EXTRACTED FROM LANSCE NEUTRON TESTING FOR THE XCKU040

Upset	Scrub ID	Frame	Word	Bit	(x,y)	Offsets
1	1	0x060980	25	23	(395648,823)	(1,-1)
2	1	0x060981	25	22	(395649,822)	(-1,1)
1	2	0x044823	115	22	(280611,3702)	NA
1	3	0x0429A9	115	31	(272809,3711)	(1,0)
2	3	0x0429AA	115	31	(272810,3711)	(-1,0)

Each offset is added to a histogram containing the frequency of each offset. To avoid duplicates, we only consider offsets with a positive x value. The resulting histogram is used in the next step.

D. Selection of Most Common Offsets

The histogram contains the frequency of each offset that occurs during the test. The most common offsets (MCOs) are chosen for the MCU reconstruction based on their frequency. The more times an offset happens the more likely that it is the effect of a single particle. In the past, the MCOs to reconstruct MCUs have been chosen arbitrarily [9], [7]. For this step, our method uses the Poisson statistics to choose the MCOs.

Relying on the nature of radiation testing following a Poisson distribution, offsets should have the same frequency in the histogram. Offsets that appear at a higher frequency are flagged as MCOs. Those MCOs comprise the adjacency model.

The MCOs are chosen based on the probability that a given offset should not appear repeatedly due to the random nature of beam testing. Using the average number of upsets per nonempty scrub cycles and the Poisson statistics, we compute the probability that a scrub cycle has two or more upsets. Since this probability is high, we know that is highly likely that any shape we only see once has been randomly generated and it is not an MCU. Given that we see a shape once, the probability of seeing it again is modeled by (2).

E. Reconstruction of MCUs

The reconstruction is an iterative process that goes through each scrub cycle and groups any two bits that have the same offset as any of the MCOs. Consider the example in Fig. 2, where a scrub cycle experienced five upsets. Also, consider that the adjacency model has only two MCOs: $(0, -1)$ and $(1, 0)$. Fig. 2(a) shows the data in the scrub cycle. The algorithm takes the first MCO on the adjacency model, $(0, -1)$ for this case, and groups the bits into MCUs. The result is shown in Fig. 2(b). The algorithm goes through all the MCOs of the adjacency model and groups the upsets into MCUs. For this example, the resulting MCUs are shown in Fig. 2(c). Then, the algorithm continues with the next scrub cycle.

IV. EXPERIMENTAL SETUP

The proposed technique was used to extract MCU data and identify micro-SEFIs on data from multiple neutron tests. The CRAM data used include three FPGAs from Xilinx: Artix-7 (XC7A200T), Kintex Ultrascale (XCKU040), and an MPSOC featuring the Zynq Ultrascale+ (XCZU9EG). BRAM data used the XC7A200T device with all the BRAMs instantiated.

We tested the three FPGAs from Xilinx at the LANSCE in December 2018. Fig. 3 shows the KCU105 board setup inside the facility. Each board was connected to a JTAG Configuration Manager (JCM) [14]. The JCM was used to

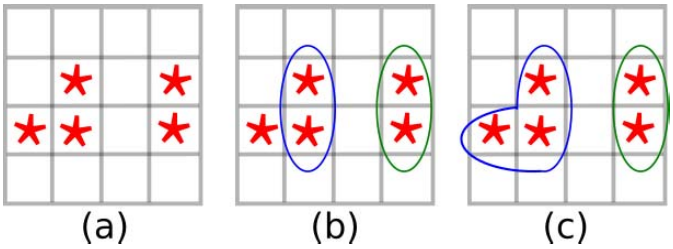

Fig. 2. Steps on the reconstruction of MCUs from upset data. (a) Five upsets that happened within a scrub cycle. (b) Grouping of MCUs after processing the first MCO $(0, -1)$. (c) Two MCUs after running through all MCOs.

TABLE IV
SUMMARY OF CRAM TESTING SETUP FOR THREE DIFFERENT DEVICES

	XCZU9EG	XCKU040	XC7A200T
Technology	FinFET16nm	Planar 20nm	Planar 28nm
Fluence (n/cm^2)	1.04×10^{11}	2.34×10^{11}	1.59×10^{11}
Bits in CRAM	142,693,248	102,800,448	59,145,600
Scrub style	JTAG	JTAG	SelectMAP
Scrub cycle time (s)	9	4	0.25
CRAM σ (cm^2/bit)	2.67×10^{-16}	2.55×10^{-15}	6.99×10^{-15}
Upsets per second	0.0313	0.215	0.340
Upset per scrub cycle	0.282	0.862	0.0850

TABLE V
SPECIFICATIONS FOR BRAM EXPERIMENT

Device	XC7A200T
BRAM Bits	59,145,600
BRAM σ (cm^2/bit)	6.32×10^{-15}
Fluence (n/cm^2)	3.47×10^{10}
Upset per scrub cycle	0.0175

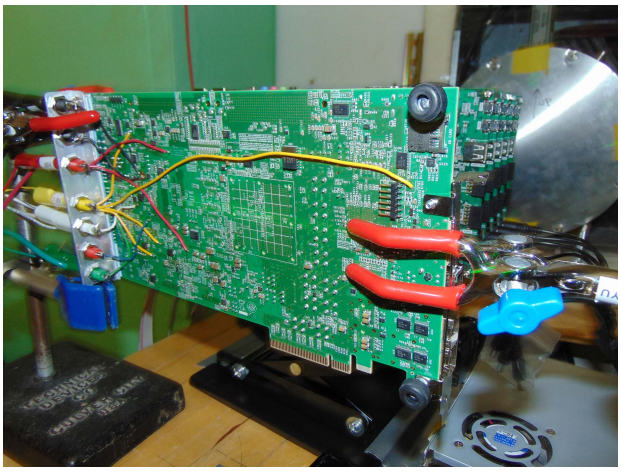

Fig. 3. XCKU040 aligned with the neutron beam.

TABLE VI
AVERAGE UPSETS PER NONZERO SCRUB CYCLE AND PROBABILITY OF CSEU FOR NEUTRON DATA GATHERED AT LANSCE

Device	Avg. upsets per non-zero scrub cycles	Cut-off value	Offsets with freq. of 1	Probability of 2+ Upsets	Probability of any one shape
XC7A200T	1.605	15	1422	0.4768	3.4×10^{-8}
XCKU040	1.658	15	26531	0.4937	1.9×10^{-8}
XCZU9EG	1.169	13	405	0.3263	1.4×10^{-8}

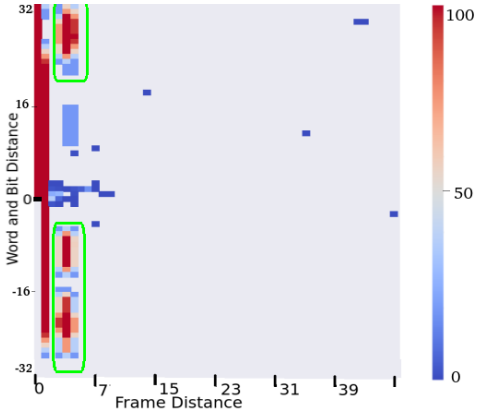

Fig. 4. Heat map of patterns for the XCZU9EG data before filtering. Blue color indicates few repetitions of the offset, while red color indicates that the offset happened more frequently.

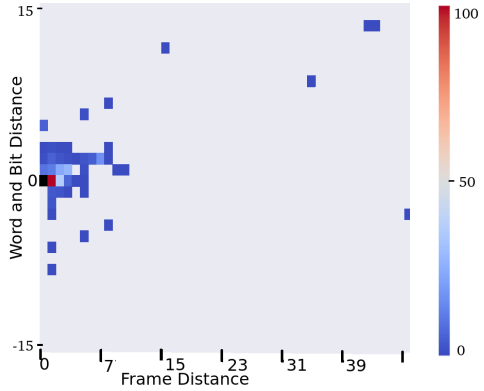

Fig. 5. Heat map of patterns for the XCZU9EG data after filtering. Blue color indicates few repetitions of the offset, while red color indicates that the offset happened more frequently.

configure the device, perform readbacks of the configuration memory and BRAMs, and scrub the configuration memory. The JCM will reconfigure the device in case an upset cannot be fixed. In addition, a timer power cycles the board every hour.

The JCM is capable of using JTAG and SMAP. On average, the JCM performed a readback and a scrub cycle every 0.25 s for the Artix-7 (with SMAP), 4 s for the XCKU040 (with JTAG), and 9 s for the XCZU9EG (with JTAG). The average flux was 8.22×10^5 n/cm^2/s. A summary of the testing setup, along with the CRAM data for the three families, is shown in Table IV. BRAM data for the seven-series device is shown in Table V.

V. CRAM RESULTS

The cutoff to filter contamination was computed using the average number of upsets for each nonzero scrub cycle as the mean of a Poisson distribution. The cutoff was set to the first value that yields a probability of 10^{-10} or less. The mean value used and the cutoff value are shown in the second and third columns of Table VI, respectively.

The filtering process is quite important to ensure accurate results. Consider Fig. 4 that shows an unfiltered heat map for radiation testing data on the XCZU9EG part. Each square on the heat map represents an upset that happened x frames and y bits away from the origin (depicted as a black square). The two long red lines on the left and the blocks of bits circled in green are contamination.

Fig. 5 shows the heat map after filtering the data. The heat map becomes much cleaner with all the red blocks of contamination gone. Thus, the heat map shows the patterns that are likely followed by MCUs. All the blocks that appeared before filtering the data jeopardize the identification of MCUs, as they introduce fake counts for the selection of the patterns.

After filtering the data, the offsets are computed. Then, the MCOs are chosen based on the probability that a given offset should not be seen frequently due to the random nature of beam testing. Using the average number of upsets per nonempty scrub cycles and Poisson statistics, we compute the probability that a scrub cycle has two or more upsets. Since this probability is high, it is highly likely that any shape that appeared once has been randomly generated and it is not an MCU. Given that we see a shape once, the probability of seeing it again is modeled by 2. The results for each device are shown in Table VI. The highest probability of seeing any one shape twice is 3.4×10^{-8} for the Artix-7 200t. Since the probability of having the same shape twice is low, all the

TABLE VII
NUMBER OF MCUS AND SCUS FOR EACH DEVICE

Device	Total Upsets	SCUs	MCUs
XC7A200T	65,935	30,186	11,567 (27.70%)
XCKU040	61,249	44,649	7,628 (14.59%)
XCZU9EG	3,971	3,508	208 (5.59%)

TABLE VIII
PERCENTAGES OF MCUS FOR EACH DEVICE

Device	MCU-2	MCU-3	MCU-4	MCU-5	MCU-6+
XC7A200T	66.3%	5.8%	5.6%	1.8%	20.4%
XCKU040	92.6%	3.8%	2.3%	0.6%	0.6%
XCZU9EG	85.1%	10.6%	2.4%	0.9%	0.9%

TABLE IX
COMPARISON BETWEEN THE PROPOSED TECHNIQUE AND THE TECHNIQUE IN [9]

Device	Technique	Total MCUs	% MCUs	MCU-2	MCU-3	MCU-4	MCU-5	MCU-6	MCU-7+
XC7A200T	Proposed	11567	27.7	66.31%	5.81%	5.62%	1.82%	18.26%	2.18%
	Previous	9800	18.5	72.43%	4.52%	5.06%	0.84%	1.01%	16.14%
XCKU040	Proposed	7628	14.59	92.61%	3.80%	2.37%	0.60%	0.17%	0.45%
	Previous	7508	14.12	95.27%	3.33%	0.84%	0.29%	0.12%	0.15 %
XCZU9EG	Proposed	208	5.59	85.10%	10.58%	2.40%	0.96%	0.48%	0.96%
	Previous	166	5	6.63%	24.10%	23.49%	11.45%	11.45%	22.89%

shapes with more than one occurrence are used for the MCU reconstruction.

To further stress the importance of filtering, we compute the MCOs for the unfiltered and filtered data shown in Figs. 4 and 5. The unfiltered data show that the four MCOs are (0, −1), (0, −2), (0, −3), and (0, −4). The filtered data show that the four MCOs are (1, 0), (1, −1), (3, −2), (1, 1). This is important because MCOs that should be disregarded can cause an increased number of MCUs or a merge of MCUs into bigger ones.

After choosing the MCOs, the MCUs are reconstructed with our proposed method. The number of MCUs and SEUs are shown in Table VII. For the XC7A200T device, 27.7% of events were MCUs, for the XCKU040 14.59% and for the XCZU9EG 5.59%.

Table VIII breaks down the MCUs based on their sizes, showing the percentages for each size. An interesting behavior was experienced by the XC7A200T device. An unexpected number of events of size greater than five were identified. Roughly 18% are 6-bit MCUs, and only 2% are larger. Although we are unable to determine the exact cause for this unexpected result, we think that a micro-SEFI affecting 6 b or the organization of the CRAM is possible explanations. However, more experiments are needed to uncover the reason behind this result. With the current data, we can say that from the decreasing behavior of the number of events with the size of the events, we would expect to have a percentage of 6-b MCUs within the range of the percentage of 5- and 7-b MCUs, and this is between 1.82% and 1.47%.

A. CRAM Result Comparison

This section compares the MCU distributions obtained with the proposed technique and the technique detailed in [9]. The results in Table IX show the number of identified MCUs, the percentage of events that were MCUs, and the distribution of those MCUs based on their size.

The method proposed in this article identifies a higher number of MCUs; this is mostly because it is not restricted by a window of 32 × 32 bits. Another factor is the usage of more MCOs. Having more MCOs increases the number of MCUs, especially 2-b MCUs because more shapes can be considered in the reconstruction of MCUs. However, it also changes the distribution because with more MCOs, some of the small-size MCUs identified with [9] become part of a larger MCU. In [9], and only the first four MCOs were used to generate the shapes.

Finally, the distributions change due to the contamination of micro-SEFI events. This is more prominent on the XCZU9EG results since the largest event found on the device was 320-b long, while in the other devices, it was only 36-b long. Without filtering the data, the MCOs changed, drastically modifying the distribution of MCUs. The unfiltered data show only 6.64% of 2-b MCUs and a high value of 22.89% for MCUs of seven or more bits.

TABLE X
SUMMARY OF THE BRAM ACQUIRED DATA

Device	XC7A200T
BRAM Bits	59,145,600
Fluence (n/cm^2)	3.47×10^{10}
Upsets	2951
Non-zero Scrub cycles	2307
Avg. upsets per scrub cycle	1.27
P(CSEU)	1.49×10^{-7}
Identified MCUs	207

TABLE XI
NUMBER OF DETECTED MCUS AND CORRESPONDING SCRUB CYCLES WITH EXACTLY n UPSETS

	MCU Size (n)			
	2	3	4	5
# of MCUs	180	16	10	1
Scrub cycles with n upsets	364 (49.4%)	75 (21.3%)	27 (37.0%)	6 (16.7%)

VI. BRAM RESULTS

All 365 BRAMs in the XC7A200T were instantiated for a static neutron test. A summary of the acquired data is presented in Table X.

Almost 3000 upsets were experienced in the BRAMs. All of those were used to perform the MCU data extraction. With the average upsets per nonzero scrub cycle, we computed the probability that an MCU identified with the proposed technique was a CSEU. The almost nonexistent probability (see Table X) shows that the identification of MCUs was successful.

Table XI shows, in the second row, the size of each of the 207 MCUs identified. The third row shows the number of scrub cycles with *n* upsets and a percentage of how many of those scrub cycles had an MCU. Some of the scrub cycles might have unidentified MCUs; however, there is not enough information since the shape of the upsets comprising the possible MCU only happened once. In the future, we will

develop a rigorous approach to determine when sufficient data to classify MCUs are acquired.

VII. Conclusion

MCUs are a concern for ECC-protected devices and for triplicated designs. This article shows a statistical technique that successfully identifies MCUs from radiation testing data. The method reduces the contamination of the data by identifying micro-SEFIs providing more accurate identification of MCUs. Likewise, more accurate results are achieved with the statistical identification of the MCOs used in the reconstruction of MCUS. Applying this method to radiation test enables the use of the extracted information to generate improved fault injection campaigns where MCUs could be injected either based on their distribution or at an accelerated pace. Moreover, researchers will have the option to control the size, shape, and frequency of the injected MCU.

As future work, we plan on investigating the extraction of MCUs using laser testing over a range of angles. Likewise, it will be interesting to explore the accuracy of our method for different duration of the scrub cycles. Finally, we will develop a more rigorous approach for determining when sufficient data are gathered to classify MCUs.

References

[1] 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.

[2] H. M. Quinn, P. Graham, K. Morgan, J. Krone, M. P. Caffrey, and M. J. Wirthlin, "An introduction to radiation-induced failure modes and related mitigation methods for Xilinx SRAM FPGAs," in *Proc. ERSA*, 2008, pp. 139–145.

[3] N. Seifert *et al.*, "Radiation-induced soft error rates of advanced CMOS bulk devices," in *Proc. IEEE Int. Rel. Phys. Symp.*, Mar. 2006, pp. 217–225.

[4] L. T. Clark and S. Shambhulingaiah, "Methodical design approaches to radiation effects analysis and mitigation in flip-flop circuits," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI*, Jul. 2014, pp. 595–600.

[5] P. E. Dodd, F. W. Sexton, and P. S. Winokur, "Three-dimensional simulation of charge collection and multiple-bit upset in Si devices," *IEEE Trans. Nucl. Sci.*, vol. 41, no. 6, pp. 2005–2017, Dec. 1994.

[6] J. D. Black *et al.*, "Characterizing SRAM single event upset in terms of single and multiple node charge collection," *IEEE Trans. Nucl. Sci.*, vol. 55, no. 6, pp. 2943–2947, Dec. 2008.

[7] J. A. Clemente *et al.*, "Statistical anomalies of bitflips in SRAMs to discriminate SBUs from MCUs," *IEEE Trans. Nucl. Sci.*, vol. 63, no. 4, pp. 2087–2094, Aug. 2016.

[8] 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.

[9] 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.

[10] G. M. Swift *et al.*, "Dynamic SEE testing of selected architectural features of Xilinx 28 nm Virtex-7 FPGAs," in *Proc. RADECS*, Oct. 2017, pp. 544–549.

[11] M. Cannon, A. Pérez-Celis, G. Swift, R. Wong, S.-J. Wen, and M. Wirthlin, "Move the laser spot, not the DUT: Investigating the new micro-mirror capability and challenges for localizing SEE sites on large modern ICs," in *Proc. 17th Eur. Conf. Radiat. Effects Compon. Syst. (RADECS)*, Oct. 2017, pp. 126–129.

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

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

[14] A. Gruwell, P. Zabriskie, and M. Wirthlin, "High-speed programmable FPGA configuration through JTAG," in *Proc. 26th Int. Conf. Field Program. Logic Appl. (FPL)*, Aug./Sep. 2016, pp. 1–4.