The Impact of Terrestrial Radiation on FPGAs in Data Centers

ANDREW M. KELLER and MICHAEL J. WIRTHLIN, Brigham Young University

Field programmable gate arrays (FPGAs) are used in large numbers in data centers around the world. They are used for cloud computing and computer networking. The most common type of FPGA used in data centers are re-programmable SRAM-based FPGAs. These devices offer potential performance and power consumption savings. A single device also carries a small susceptibility to radiation-induced soft errors, which can lead to unexpected behavior. This article examines the impact of terrestrial radiation on FPGAs in data centers. Results from artificial fault injection and accelerated radiation testing on several data-center-like FPGA applications are compared. A new fault injection scheme provides results that are more similar to radiation testing. Silent data corruption (SDC) is the most commonly observed failure mode followed by FPGA unavailable and host unresponsive. A hypothetical deployment of 100,000 FPGAs in Denver, Colorado, will experience upsets in configuration memory every half-hour on average and SDC failures every 0.5–11 days on average.

CCS Concepts: • **Hardware → Hardware accelerators**; **Reconfigurable logic applications**; **Failure prediction**; **Transient errors and upsets**; **Emerging architectures**; **Board- and system-level test**; **Fault models and test metrics;**

Additional Key Words and Phrases: SEU, configuration scrubbing, fault injection, neutron radiation testing

ACM Reference format:
Andrew M. Keller and Michael J. Wirthlin. 2021. The Impact of Terrestrial Radiation on FPGAs in Data Centers. *ACM Trans. Reconfigurable Technol. Syst.* 15, 2, Article 12 (December 2021), 21 pages.
https://doi.org/10.1145/3457198

1 INTRODUCTION

Field programmable gate arrays (**FPGAs**) are being used in large quantities in data centers around the world. A highly scaled deployment of FPGAs may include tens of thousands [5], to hundreds of thousands [8], to perhaps millions of devices. These devices are used for cloud-computing [1] and computer networking applications [22]. The number of devices deployed scales with demand, and the demand for these devices in data centers is expected to increase [7]. FPGAs are programmable integrated circuits that implement custom logic. These devices take advantage of parallelism, which accelerates applications. They offer potential improvement in performance

This work was supported by the I/UCRC Program of the National Science Foundation under Grant No. 1738550, by the Utah NASA Space Grant Consortium, and by the Los Alamos Neutron Science Center under proposal NS-2018-8031-A.
Authors' address: A. M. Keller and M. J. Wirthlin, Brigham Young University, 450 Engineering Building, Provo, Utah, UT 84602-4118; emails: {andrewmkeller, wirthlin}@byu.edu.

1936-7406/2021/12-ART12 $15.00
https://doi.org/10.1145/3457198

and power consumption over super-scalar CPUs [25] and GPUs [16]. Their reprogrammability allows the same device to be reused for implementing different applications or for updating a design that has already been deployed.

Static random access memory (**SRAM**)-based FPGAs may be reprogrammed an unlimited number of times. These devices use SRAM cells to store device configuration. Storing device configuration in SRAM cells allows the device to be re-configured an unlimited number of times without wearing out the device. The ability of SRAM-based FPGAs to be reprogrammed without adverse effects is one of the primary features that prompt their use in data centers.

The large amount of internal memory cells needed to support the reprogrammability of SRAM-based FPGAs makes these FPGAs susceptible to radiation-induced upsets [27]. Radiation found in terrestrial environments [4] can induce sufficient energy into these devices to alter the values stored in configuration memory. These alterations can modify the functionality of a design implemented on the FPGA, causing the design to misbehave, resulting in failure [6, 26]. The likelihood of terrestrial radiation causing design failure in a single FPGA instance is extremely small, but the use of a large number of FPGAs in data centers increases the likelihood of radiation-induced failure [19]. While the devices presented in [19] are dated, the concepts conveyed hold true and are applicable to modern devices and their respective sensitivities [27].

This article explores the impact of terrestrial radiation on FPGAs in data centers and examines the behavior of cloud-computing-like applications while they are exposed to the effects of terrestrial radiation. The applications evaluated are implemented on an FPGA that is attached to a host computer. The effects of terrestrial radiation are mimicked through artificial **fault injection** (**FI**) (i.e., purposefully writing corrupt values to configuration memory) and through accelerated radiation testing. This article builds upon a previous study [13]. The previous study examined the impact of soft errors through persistent FI where upsets were allowed to remain present throughout the execution of the test applications. This article continues the study with actual radiation testing and a new FI scheme that allows upsets to be present intermittently during application execution. Upsets are intermittent because configuration scrubbing is employed in the background. **Silent data corruption** (**SDC**), or the return of erroneous data from the FPGA without warning, is found to be the dominant failure mode. In a hypothetical system with 100,000 FPGAs deployed in Denver, Colorado, SDC is estimated to occur once every 0.5–11 days on average.

2 FPGAS IN DATA CENTERS

FPGAs are a remarkable technology. These devices contain configurable resources that are used to implement custom logic circuits. The same device can be used for several different applications or can be reused for design updates (assuming reprogrammability). Unlike CPUs or GPUs, FPGAs allow for complex implementations of custom architectures. These architectures have the ability to exploit more parallelism and make better use of resources, yielding higher performance and better energy efficiency. One study demonstrated a 60% higher performance at 2.3× better energy efficiency (i.e., performance per watt) on a Stratix 10 FPGA over a Titan X Pascal GPU [16]. Another study demonstrated an order of magnitude performance increase per joule on an FPGA over other computing platforms and a 250× performance improvement over a CPU [25]. Unlike **application specific integrated circuits** (**ASICs**), FPGAs can be reprogrammed after deployment. This greatly increases their versatility and reduces development cycles and costs. The use of FPGAs in data centers makes these advantages more widely available.

FPGAs in data centers accelerate many **high-performance computing** (**HPC**) applications. HPC applications exist in science, business, and everyday living. In one study, several Intel Stratix V FPGAs are used to accelerate DNA sequencing [2]—an application in Genomics [21]. In another study, FPGAs are used to accelerate financial applications [23]. A third study uses Intel Stratix

10 FPGAs to accelerate the serving of deep neural networks for real-time artificial intelligence applications [16]. Many other HPC applications are being accelerated using FPGAs.

This article is motivated by the use of FPGAs in data centers for networking [12, 22]. Data centers contain tens to hundreds of thousands of computers or network nodes that all need to be connected together. An FPGA provides an ideal platform for implementing a computer network because a single device can process terabits of information per second and is furnished with a number of high speed connections. The use of FPGAs for networking allows networking approaches to be tried and proven before they are taped out to an ASIC, and it also allows networking technology a more timely market entrance. To support a large number of connections in the data center, a large number of FPGAs are used. FPGAs used in these settings are expected to operate error free, for long periods of time, without any opportunity for being reprogrammed or power cycled expect through planned maintenance. Understanding the impact of radiation-induced soft errors on large-scale deployment is extremely important for the development of resilient system with strict reliability requirements.

While the primary focus of this article is on cloud computing, its findings are applicable to a broader domain. Motivated by the large-scale deployment of FPGAs seen in data centers, the findings of this article are applicable to cloud computing applications, computer networking applications, and any other large-scale deployment of FPGAs where reliability is a concern. A single instance of an FPGA in a terrestrial environment has little to worry about soft errors, but collectively a large number of FPGAs will experience an increased frequency of soft errors. This is problematic for service providers of FPGA-based services such as cloud computing and computer networking. This study examines the impact of terrestrial radiation on FPGAs in data centers. It demonstrates the prevalence of soft errors in scaled settings, and it estimates the occurrence rates of SDC failure events among other failure modes despite the employment of configuration scrubbing.

To meet FPGA cloud computing needs, companies such as Microsoft, Amazon, Baidu, Nimbix, Micron, and others have rolled out associated large-scale systems and products. Microsoft headed FPGA cloud computing efforts in 2014 with their release of project Catapult, an FPGA acceleration platform for the Bing search engine. A subsequent development in 2016 resulted in a test system that contained 50,000 Stratix V GS D5 FPGAs [5]. A 2017 report suggests that a follow on project, Brainwave, was making use of hundreds of thousands of FPGAs (potentially Stratix 10 FPGAs) [8]. Amazon pioneered the FPGA cloud computing instance and now offers F1 instances from data centers around the world: US East (N. Virgina), US West (Oregon), EU (Ireland), and AWS Gov-Cloud (US). They offer subscriptions to cutting edge FPGAs that can be scaled on demand. Projections suggest that the demand for FPGA cloud computing will continue [7].

FPGA cloud computing nodes tend to consist of a host computer paired with one or more SRAM-based FPGA instances [1, 5]. A host computer is likely to be connected directly to one or more FPGAs via a **peripheral component interconnect express** (**PCIe**) connection. This allows large amounts of data to be transferred back and forth between the host computer and accompanying FPGAs. It is also common for these cloud computing nodes to have DRAM connected to the FPGA as a shared memory between the FPGA and host that can be used by an acceleration application. In addition to these connections, a dedicated network connection or bump-in-the-wire architecture may be used to place the FPGA on a network [5]. This allows for remote allocation and pooled processing without directly going through a host computer.

Figure 1 depicts the typical setup of an FPGA instance in cloud computing. A portion of the FPGA resources are dedicated to the system infrastructure (e.g., PCIe controller, DDR controller, conventional interfaces) to provide a shell or an overlay in which the target application (i.e., kernel or role) can be instanced. The shell may not be refreshed or reprogrammed when a kernel is

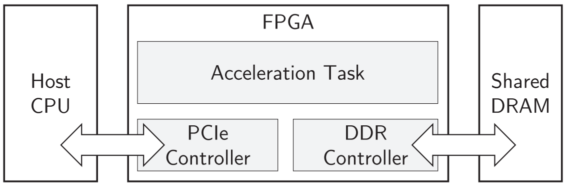

Fig. 1. Typical FPGA data center node.

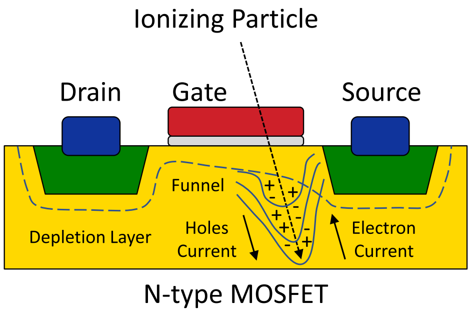

Fig. 2. Soft error fault mechanism; Adapted from Fig. 3.3 in [3].

loaded so as to provide continuous network services to a host computer when a bump-in-the-wire networking architecture is used, or the shell and kernal may be reprogrammed and refreshed together as part of the FPGA-host application initialization. Resource not utilized by the shell is made available to the target application. Using the soft fabric of the FPGA to implement the shell allows for architecture updates without requiring a new device. The baseline system used in this article follows the setup presented in Figure 1 with the shell and kernel programmed once before the initial execution of an FPGA-host application.

3 SOFT ERRORS IN DATA CENTER FPGAS

Soft errors are changes in the internal state of a device that result from device interactions with a single energetic atomic particle [10]. When ionizing radiation passes through an integrated circuit, such as an FPGA, it can deposit enough energy to disrupt the proper flow of electricity through the device. This fault mechanism, known as the funneling phenomenon [3], is depicted in Figure 2. It is associated with **single event effects** (**SEE**) or radiation effects caused by a single energetic particle. Soft errors are the subset of SEEs that are non-destructive. They do not permanently damage a device. The effects of a soft error SEE are temporary or can be overwritten.

An important type of SEE for SRAM-based FPGAs are **single event upsets** (**SEUs**). An SEU alters the value stored in a memory cell. This can have negative effects on an SRAM-based FPGA design because circuit configuration and state are stored in susceptible memory cells. When an SEU occurs in configuration memory, it can disconnect components, or short them together; it can alter logic equations, or change the intended behavior of a component. These alterations cause errors in the circuit that may propagate to the output of the device, causing unexpected behavior [6, 26].

Radiation present in terrestrial environments (i.e., on Earth) that can induce soft errors consists primarily of high-energy neutrons, low-energy thermal neutrons, and alpha particles [4]. Neutrons indirectly induce soft errors through interactions with device materials. Most of the neutrons present on Earth are a result of a complex cascade of interactions between intergalactic cosmic rays and particles in Earth's atmosphere. Primary focus in this article is given to high-energy neutrons.

Table 1. Neutron Flux at Various Locations

Location	Elevation	Relative Neutron Flux
Seattle, WA	160 ft	1.05
Moscow, Russia	490 ft	1.14
Chicago, IL	590 ft	1.19
Denver, CO	5280 ft	3.76
Los Alamos Natl. Lab.	7380 ft	5.60
Leadville, CO	10170 ft	10.79
White Mtn. Res. Sta.	12500 ft	15.07

3.1 Soft Error Rates

The rate at which soft errors occur is often measured in terms of **failures in time** (**FIT**) or **mean time to failure** (**MTTF**). These **soft error rate** (**SER**) metrics reflect the frequency of soft errors in relation to time. FIT is the average number of failures that occur within a billion hours of operation [10]. MTTF is the average amount of operational time that passes before a failure occurs. MTTF and FIT are inversely related as follows:

$$\frac{\text{MTTF in hours}}{1 \times 10^9 \text{ hours}} = \frac{1}{\text{FIT}}.$$

A 114 year MTTF is approximately 1000 FIT. SERs for large memory components are often reported in terms of FIT per megabit (FIT/Mbit, i.e., 1×10^6 bits). Neutron FIT rates are typically normalized to the amount of neutron radiation present in New York City (NYC). Radiation exposure rates are measured in terms of *flux* or the number of particles that pass through a particular area in a given amount of time. In NYC, outside, at sea level, during average solar activity, approximately 13 high-energy neutrons (i.e., greater than 10 MeV) pass through a square centimeter of area every hour (a high-energy neutron flux of 13 n cm^{-2} h^{-1}). This information can be used to scale normalized SERs to other locations.

Two main factors play a key role in the overall SER for FPGAs in data centers: location, and the number of FPGAs deployed. SERs are usually normalized to a fixed location and a fixed number of susceptible components (e.g., one megabit of memory, or a single device). The scaling of SERs, based on location and number of deployed devices, affects the impact of terrestrial radiation on FPGAs in data centers. SERs increase as FPGAs are deployed in locations where more radiation is present, and they also increase as more FPGAs are deployed.

SERs are higher at higher altitudes because there is less shielding of cosmic radiation from the atmosphere at higher altitudes. Several factors influence the amount of neutron radiation present including geomagnetic cutoff, solar activity, and atmospheric depth or elevation; but elevation is the most important parameter for determining terrestrial neutron flux [10]. Table 1 shows neutron flux as measured in various locations compared to the reference flux of NYC. As elevation increases, neutron flux increases exponentially. Some areas have considerably higher neutron flux. For example, the White Mountain Research Center in the USA receives 15× the reference flux with a flux of approximately 195 n cm^{-2} h^{-1}. This increase in radiation is expected to increase SERs by the same factor.

Although the configuration memory (CRAM) of an FPGA is susceptible to radiation-induced SEUs, the likelihood of an SEU occurring in a single FPGA instance (in a terrestrial environment) is very low. For example, an FPGA containing fifty-million CRAM bits at 25 FIT/Mbit would experience one SEU every hundred years on average. Modern FPGAs experience configuration upsets at a rate of 5-76 FIT/Mbit [27] and contain tens of millions to billions of CRAM bits. The frequency

Table 2. Neutron SER for 28-nm FPGAs

Device	Stratix V GX A7	Kintex 7 325T
CRAM FIT/Mbit	63	74 ± 18%
CRAM Bits	~99,000,000	~73,000,000
CRAM FIT/Device	6,200	5,400
CRAM MTTU NYC	18.3 years	21.2 years

Table 3. Stratix V GX A7 Mean Time to SEU at NYC Neutron Flux

FPGAs	Years	Days	Hours	Minutes	Seconds	Total (Seconds)
1	18	127	15	28	27	6E+8
10	1	304	23	8	50	6E+7
100		67	0	30	53	6E+6
1,000		6	16	51	5	6E+5
10,000			16	5	6	6E+4
100,000			1	36	30	6E+3
1,000,000				9	39	6E+2
10,000,000					57	6E+1

of SEUs increases with more bits and higher FIT rates, but it still tends to be low for a single device. If only a single instance is to be considered, then the frequency of SEUs may be of little concern. Concern over soft errors in terrestrial deployments is likely to be found in large-scale deployment of devices that themselves only carry a low probability of soft error occurrence.

The SER for the FPGA used in this article (a Stratix V GX A7 FPGA) is shown in Table 2. It is compared against another comparable FPGA from a different vendor [11]. The SERs shown are the neutron SERs for SEUs in CRAM normalized to the NYC neutron flux. The SEU CRAM FIT per device is the product of FIT/MBit and the number of CRAM bits in the device. A single instance of the FPGA used in this article would experience one SEU every 18 years on average, which by itself is infrequent. When considering larger quantities, the SER of this device becomes more concerning.

Large-scale deployments of FPGAs in data centers make use of large quantities of FPGAs, and this increases the overall SER. The SER of SEUs in CRAM increases linearly as more and more devices are deployed [19]. The per instance or per device SER remains unchanged, but the overall SER for the complete system increases. Table 3 shows how the SER increases in a hypothetical situation as the number of deployed Stratix V GX A7 FPGAs increases. The SER is given in terms of the mean time to SEU in CRAM based on a NYC reference flux. A large scale deployment of 100,000 FPGAs would experience one SEU every hour-and-a-half on average. This suggests that large-scale deployments of FPGAs in data centers could result in a significantly high SER.

A real world example of SER scaling caused by location and the number of devices deployed is found in [5]. This study reports that one SEU was observed in FPGA CRAM every 1,025 machine days on average (i.e., for a single FPGA instance). The FPGA deployed is a Stratix V GS D5 FPGA. This FPGA has approximately 79 million CRAM bits based on reports generated by vendor tools. Using a FIT/Mbit of 63 from [11] scaled to the number of CRAM bits in the devices yields a whole device SEU rate of approximately 5000 FIT. The equates to one upset every 8,333 machine days. The difference in SERs (1,025 vs 8,333 machine days) suggests that the FPGAs used in [5] are deployed in a data center where the neutron flux is approximately 8× greater than it is in NYC

Table 4. Breakdown of Memory in a Stratix V GX A7 FPGA

Type	Bits	Percentage
CRAM	91,170,156	60%
BRAM (M20K)	52,428,800	34%
LUTRAM (MLAB)	7,511,040	5%
Flip-Flop	938,880	1%

(8333 divided by 1025). This estimate is reasonable given the scaling of neutron flux shown in Table 1. An SER of 1,025 machines days for the 50,000 node study equates to one SEU every half hour. Thus, the scaling of SER based on location and number of devices deployed is observed in a real world system. The occurrence of SEUs every half hour brings into question the effects of SEUs on FPGAs in large-scale data center deployments.

4 CONFIGURATION SCRUBBING

Configuration scrubbing is a mechanism that is commonly offered in SRAM-based FPGAs. This scrubbing mechanism identifies SEUs in configuration memory and overwrites them with correct values [24]. By correcting corrupted values, SEU configuration scrubbing mechanisms lower the likelihood of failure and increase the likelihood of functional restoration. Configuration scrubbing is often performed in the background. When performed in the background, configuration scrubbing has no impact on the performance of a design. Configuration scrubbing does, however, require a small amount of additional power. The susceptibility of SRAM-based FPGAs to SEUs has prompted the development of such mechanisms.

Configuration scrubbing is commonly offered as a built-in feature to the device because it addresses a large portion of device memory that is susceptible to SEUs. Table 4 shows that configuration memory makes up 60% of all SEU susceptible memory in a Stratix V GX A7 FPGA (the device used in this article). Excluding user block memories (BRAM), which have alternate methods available to address SEUs, CRAM makes up more than 90% of the remaining memory that is susceptible to SEUs. Distributed memory held in writable look-up tables (LUTRAM) and registers (flip-flops) make up a much smaller portion of susceptible memory. By offering internal configuration scrubbing, a large portion of susceptible memory is benefited.

Configuration scrubbing is slow compared to the operational speed of an FPGA design. Configuration scrubbing of an entire device may complete on the order of 100 ms, whereas an FPGA design may operate at 100 MHz or 10 ns per clock cycle (nominal). In this scenario, an SEU is present on average for 50 ms or 5 million clock cycles before it is scrubbed.

While an SEU is present in configuration memory, between occurrence and scrub, it has a direct effect on the resources associated with the affected bit. There is no grace period or buffer to protect underlying resources from being affected by the SEU. FPGA configuration memory is connected directly to the physical resources that it governs. As soon as an SEU occurs, its effect takes place. The overall effect of an SEU on a design depends on the use of affected resources and the propagation of signals throughout the design. Given that an SEU may be present for millions clock cycles or more before being scrubbed, there is ample opportunity for the SEU to adversely affect the design. The circuit of the design may be incorrect during this period of time even though configuration scrubbing is employed.

Configuration scrubbing cannot undo adverse affects to design outcomes, but it can restore proper functionality. While an SEU is present, it can cause the circuit to misbehave. If the presence of the SEU does not cause any lingering side effects [15], then the circuit functionality may be

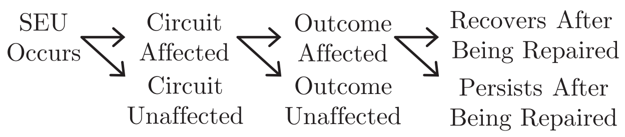

Fig. 3. Possible outcomes of an SEU occurrence.

restored when the offending SEU is scrubbed. Errors or failures that resolve themselves upon SEU scrubbing are non-persistent. Errors or failures that do not resolve once an SEU is scrubbed are persistent. Restoring the circuit to its proper functionality prevents future errors that would have occurred had the SEU been allowed to remain.

Configuration scrubbing lowers the likelihood of failure by decreasing the concurrence of SEUs and stimulus. In order for an error or a failure to occur, an SEU must be present *with* stimulus to activate affected resources. If the affected resources are not active (i.e., they will not affect the outcome of the design through signal propagation or otherwise), then the SEU will have no effect on the outcome of the design. The longer that an SEU is present, the more likely its presence is to coincide with the introduction of stimulus that activates affected resources, resulting in an error or failure. Resources that are used by a device are not likely to be active all at the same time. By continuously and quickly scrubbing SEUs as they occur, the likelihood of concurrence between SEU and activating stimulus decreases.

5 EFFECTS OF SEUS ON FPGA DESIGNS

SEUs in SRAM-based FPGAs corrupt device configuration and active design state. CRAM enables routes, sets **look-up table** (**LUT**) equations, and adjusts the behavior of circuit components. Active design state includes the values in registers (i.e., flip-flops), distributed memories (i.e., LUTRAMs), block memories (i.e., BRAMs), and control registers for specialized **intellectual property** (**IP**). Values stored in active design state update during design runtime whereas values in CRAM do not typically change once initialized. The corruption of values stored in CRAM and active design state by SEUs may lead to unexpected behavior.

The occurrence of an SEU does not guarantee that design failure will ensue. In order for a design to fail, the SEU must affect resources utilized by the design, the affected resources must be activated by stimulus presented to the design (i.e., must contribute to the outcome of the design), and the affected outcome of the design must fall outside of specified tolerable behavior. These requirements mitigate the severity of an SEU occurrence because they specify that only a portion of SEUs (when present during certain periods) will lead to design failure.

Figure 3 summarizes the possible outcomes of an SEU occurrence. When an SEU occurs, it could have no effect on the underlying circuit of the implemented design. This happens when an SEU affects a resource that is not utilized by the design such as a route, LUT, or flip-flop that is not a part of the implemented design. If an SEU does affect the underlying circuit of a design, it may have no effect on the outcome of the design. The effects of an SEU on a circuit may be masked functionally, logically, or temporally [22]. Furthermore, if the SEU is scrubbed before its effects on the circuit are activated by design stimulus, then the outcome of the circuit will be unaffected due to the apt scrub of the SEU. If the outcome of the design is affected, the behavior of the design must fall outside of tolerable specifications in order to be considered a failure. Finally, once a failure has occurred, scrubbing the SEU may allow the circuit to recover and perform correctly for future operations, or the circuit may have entered a persistent failure state where SEU scrubbing alone is insufficient to restore proper design functionality [15].

In the previous study [13], failures observed in several FPGA cloud-computing-like applications fell into three main categories: host unresponsive, FPGA unavailable, and SDC. A host unresponsive failure occurs when an SEU in an attached FPGA results in the host computer locking up. This may be related to the way in which drivers handled unexpected behavior from the attached FPGA. FPGA unavailable failures result when the host can no longer communicate with or control the attached FPGA. SDC failures result when the FPGA returns erroneous results to the host computer without any indication of the error.

SDC failures are perhaps the most concerning type of failure. Failures that occur in the background without warning can be more devastating than blatant failures that are easily detected. Host unresponsive and FPGA unavailable are easily detected and can be resolved through recovery operations such as a power cycle of the host computer or a reprogramming of the FPGA accompanied by a rescan of components on the host computer. SDC failures, on the other hand, can be harder to resolve. They encompass any data corruption that is unaccompanied by a warning or detection. The effects of an SDC range from minor to major. For example, an SDC could result in a minor loss of a single service request or result in a major permanent loss of service without system awareness. SDCs that carry more severe consequences require more serious attention.

Knowing that SEUs occur more frequently in large-scale deployments of FPGAs, it becomes necessary or helpful to better understand how FPGAs applications respond in the presence of SEUs. This information allows the impact of terrestrial radiation to be appropriately scaled from the occurrence of any SEU to the occurrence of SEU-induced failures.

6 TEST METHODOLOGY

The objective of this article is to better understand the impact of terrestrial radiation on FPGAs in data centers. This is accomplished by exposing several cloud-computing-like FPGA applications to SEUs and observing their response. The observed responses are then scaled to reflect the potential impact of terrestrial radiation on large-scale deployments of FPGAs in data centers.

The first type of testing conducted is FI. FI is a common way of emulating SEUs to observe system response [18]. SEUs are mimicked in an FPGA by purposefully writing incorrect values to CRAM. Incorrect values written to the CRAM are the faults injected. Often times, faults will be injected randomly into the device to better emulate the behavior of SEUs caused by radiation. Randomly injecting faults also allows evaluation to made through population sampling [20]. Random FI returns the percentage of faults that result in a failure. The statistical significance of this percentage is reflected by its confidence interval. The 95% confidence interval for all FI results provided in this article is determined using a normal approximation of the binomial distribution.

The percentage of faults resulting in failure evaluated through random FI can be used to estimate the real time FIT of failures for the evaluated design. This is accomplished by multiplying the estimated percentage by the CRAM FIT of the whole device found in Table 2. This effectively scales down the rate of any SEU occurrence to the rate of SEU occurrences that result in a failure.

The second type of testing conducted is accelerated radiation testing. This form of reliability testing exposes an integrated circuit to an accelerated radiation source. A high-energy neutron source that closely reflects the energies of neutrons observed in terrestrial environments is used in this article. Using an accelerated source allows for a large amount of data to be collected in a small amount of time. This testing approach provides a measurement that can be scaled to reflect the real time FIT of failures for the evaluated design. More details are provided in Section 8.

Several FPGA applications are tested to provide a general sense of how cloud-computing-like applications respond to SEUs. A total of seven applications are tested on an FPGA-host system using FI and accelerated radiation testing. This section presents the FPGA-host system and FPGA

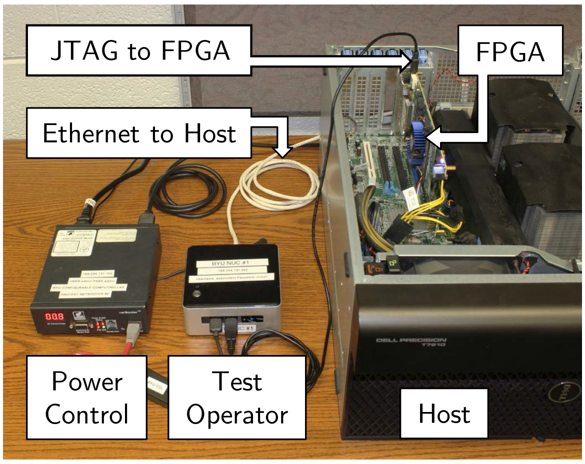

Fig. 4. Basic experiment setup.

applications used in this article. Subsequent sections discuss the FI and radiation testing approaches employed.

Three tests were undertaken to study the impact of SEUs on FPGA-based computing systems. First, an initial FI test was run against the FPGA applications. This test allowed injected faults (mimicked SEUs in CRAM) to remain present throughout design stimulus execution. The second test uses neutron radiation testing to induce SEUs randomly. This test made use of internal configuration scrubbing making the presence of SEUs during design stimulus execution intermittent. This change in fault behavior yielded unexpected results (lower susceptibility to radiation-induced failure than expected). A final intermittent FI test was run against the applications after the radiation test. This FI test incorporated asynchronous fault introduction with configuration scrubbing to yield a more accurate estimation of the application's response to terrestrial radiation as observed in neutron radiation testing.

6.1 FPGA-Host System

Figure 4 displays the test platform that was developed and used for the experiments in this article. The same test platform is used for all three tests: the initial FI, the accelerated radiation, and the intermittent FI. The platform consists of a single FPGA paired with a host computer. The Stratix V GX A7 FPGA resides on a Terrasic DE5-Net accelerator board, which also features 4 GB of shared DRAM controlled through the FPGA. This board is connected through a PCIe 8x connection to the host computer. The host computer is a Dell Precision T7610 workstation featuring 16 GB of ECC-protected DRAM, two Intel Xeon E5-2609 processors, and Windows 10 Professional as the operating system.

In addition to the FPGA and host computer, the test setup includes a network operated power outlet controller and an additional computer to operate the test. This computer orchestrates the experiment and controls the host computer, FPGA, and power controller. The test operator programs the FPGA and triggers a PCIe bus rescan to load the newly programmed FPGA device onto the computer. It also launches host applications associated with the programmed FPGA design on the host, performs SEU data collection and FI through the JTAG connection, and power cycles the host and FPGA as needed through the power controller.

Table 5. Benchmark Design Resource Utilization

Design	Total ALMs	Routing	Critical Bits
Mandelbrot	173,755 (74.03%)	29.40%	67.5%
Boardtest	57,547 (24.52%)	11.90%	24.4%
Video Downscaling	50,914 (21.69%)	10.80%	21.2%
Matrix Multiply	135,405 (57.69%)	28.40%	59.6%
Sobel Filter	48,573 (20.69%)	9.20%	19.3%
Vector Add	49,039 (20.89%)	9.70%	20.1%
JPEG Decoder	95,250 (40.58%)	17.70%	42.0%

The Stratix V GX A7 FPGA used in this article has configuration scrubbing features built in [9]. This device can detect and correct SEUs (single-bit and double-adjacent bit SEUs) in CRAM while the design is running. In this article, the internal scrub period is set to complete every 94-380 ms. This device also allows for the deliberate insertion of configuration errors through FI. Fault injection and radiation tests were conducted with internal and external scrubbing.

6.2 FPGA Applications

Several FPGA applications were tested to provide a general sense of how FPGA cloud-computing-like applications respond to SEUs. A total of seven FPGA applications were selected for this article from example designs included in the Intel FPGA SDK for OpenCL. These applications range from simple computations, like "Vector Add" and "Matrix Multiply", to video and image processing applications, to heavier computations and a board stress test. The most important characteristic of these applications is that they are realistic and make use of a cloud-computing-like hardware setup. The designs were compiled with Intel Quartus Prime 18.0 Standard using the Terrasic DE5-Net board support package.

The selected designs range in complexity and resource utilization. Table 5 lists the name of each design tested and denotes resource utilization. The number of **adaptive logic modules** (**ALMs**) that are used by a design is included. Each ALM is a small collection of logic resources consisting of LUTs, registers, and adder logic [9]. The use of more ALMs by a design generally indicates that the design is more spread out on the FPGA and consumes more resources. The percentage of routing resources utilized by the design is also included. Finally, the table includes the percentage of all CRAM bits that are deemed critical by vendor tools [9]. Critical bits, as determined by vendor tools, are CRAM bits that are potentially associated with resources utilized by a design. An SEU in a critical bit does not guarantee design failure. These metrics provide an idea of the differences in size and complexity of the selected designs.

7 INITIAL FAULT INJECTION

This is the first of the three tests that are conducted to better understand the impact of terrestrial radiation on FPGAs in data centers. This test uses FI to mimic the behavior of SEUs in FPGA configuration memory. As an initial test, it provides a starting point to understanding the true behavior of SEUs. This test was run before radiation testing. It exercised the experimental setup and provides an initial estimate on the impact of terrestrial radiation.

The initial FI test run on the FPGA applications in this article followed the generalized approach for FI as outlined in [18] according to the flow chart shown in Figure 5. The initial FI test is sequential in nature. First, the FPGA system is brought into a **working state** (**WS**). Second, a random fault is introduced into the FPGA with the design loaded. Faults are injected into any CRAM bit,

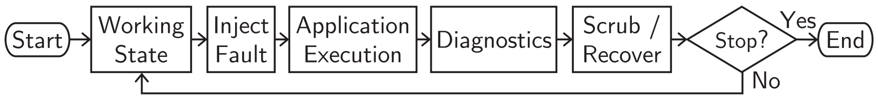

Fig. 5. Fault injection flow.

Table 6. Initial FI Results

Design	Faults Injected	Any Failure (95% Confidence Interval)	SDC	FPGA Unavailable	Host Unresponsive
Mandelbrot	9,301	11.6% (10.9%, 12.3%)	11.3%	0.3%	0.02%
Boardtest	12,247	4.1% (3.7%, 4.5%)	3.4%	0.6%	0.03%
Video Downscaling	11,641	3.7% (3.4%, 4.0%)	3.0%	0.6%	0.04%
Matrix Multiply	17,094	9.6% (9.2%, 10.0%)	9.0%	0.5%	0.03%
Sobel Filter	6,638	1.9% (1.6%, 2.2%)	1.3%	0.5%	0.03%
Vector Add	11,623	2.5% (2.2%, 2.8%)	2.0%	0.4%	0.04%
JPEG Decoder	7,948	7.0% (6.4%, 7.6%)	3.4%	3.3%	0.14%

they are not isolated to critical bits. Third, the FPGA-host application is executed in its entirety, which pushes test stimulus to the FPGA design and collects results from the device. Fourth, diagnostics are run to determine if the FPGA returned the expected response. Finally, the injected fault is scrubbed though external configuration scrubbing. The system is then recovered and brought back into a WS had any failures occurred. This cycle of sequential steps repeats again and again until sufficient data have been collected.

This initial FI approach emulates the worst case scenario for the presence of SEUs in the device. In this approach, the injected fault is allowed to remain present for the entirety of the test application execution, meaning that the fault is exposed to, and coincident with, all of the test stimulus presented to the design. Such a scenario would occur in an FPGA data center application (such as computer networking) where the CRAM of an FPGA is infrequently refreshed. However, many FPGA data center applications in cloud computing or short lived computations may frequently reprogram the FPGA and have configuration scrubbing enabled, which would reduced the coincident presence of SEUs and stimulus. The initial FI approach collects important information as to the worst case percentage of bits that cause failure, the distribution of failure modes, and the recovery behavior of the system; and the initial FI approach vetted the system for radiation testing, but the emulated behavior of the presence of SEUs inaccurately reflects the behavior observed in radiation testing.

The results from the initial FI approach are shown in Table 6. The data presented here is a subset of data that took approximately 100 days of continuous testing to obtain. These results provide an important reference point for the remaining results presented in this article. A 95% confidence interval is provided for the “Any Failure” condition to capture statistical significance. Uneven distribution is found in the number of faults injected. This results from additional data being collected to narrow confidence intervals where seen best at the time of data collection. Across all seven tested applications, failures were observed in 2–12% of all randomly injected faults. SDC was the most prevalent failure mode followed by FPGA unavailable and host unresponsive. These results are discussed in more detail in Section 10 where they are compared with the results of the two other tests.

8 NEUTRON RADIATION TEST

This is the second of the three tests conducted. It builds off of the initial FI test by using the same test setup in an accelerated neutron beam. Accelerated neutron radiation testing exposes the FPGA applications to large amounts of radiation that closely reflect the kind of radiation present in terrestrial environments. It generates a lot of data quickly and the results should reflect real time failure rates more accurately. The greatest difference between this test and the previous test is the way in which upsets are introduced and scrubbed. The previous test introduced faults before application execution and scrubbed them after. This test allows upsets to be introduced and scrubbed while the FPGA application executes.

Accelerated neutron radiation testing is the de facto standard for evaluating the impact of terrestrial radiation induced soft errors in semiconductor devices [10]. The purpose of accelerated radiation testing is to observe the effects of radiation on a device in a much shorter time scale. Real-time observation can be done as in [14], but real-time radiation testing typically takes much longer and requires a large number of devices to shorten the data collection period. With accelerated radiation testing, a single device can be exposed to a large amount of radiation in a short period of time. This allows data to be collected quickly. It is recognized that accelerated testing causes upsets to occur more quickly than they would under normal circumstances in a terrestrial environment. This can overwhelm the test system and cause unrealistic accumulations of upsets. In some sense, accelerated radiation testing provides a worst-case bound on expected behavior. There is no indication that the radiation testing data obtained in this article has been significantly skewed by the rate of radiation exposure.

The data collected from accelerated neutron radiation testing reflects the behavior of the device as it operates in the presence of radiation found in a terrestrial environment. Of the radiation found in a terrestrial environment that affect integrated circuits like SRAM-based FPGAs, high-energy neutrons are perhaps the most prevalent form [4]. By exposing a device to a large amount of high-energy neutrons, an estimate can be obtained as to its behavior in deployment.

Neutron radiation tests of the Stratix V accelerator applications were conducted at the **Los Alamos Neutron Science Center** (**LANSCE**) in December, 2018. This facility provides a spallation neutron source with an energy spectrum that is similar to that found in terrestrial environments. The rate of radiation exposure in the accelerated radiation beam is much higher than the reference neutron flux found in NYC. The instrument used at LANSCE provided an average high-energy neutron flux of 9×10^5 n cm^{-2} s^{-1}, which is approximately two-hundred fifty million times higher than the NYC high-energy neutron flux. Using this source for the experiment allows for greatly accelerated data collection with an SEU being introduced into the target device approximately every two seconds on average.

Figure 6 shows the Stratix V (SGXA7) FPGA development board mounted perpendicular to the neutron beam flight path with the 2 inch collimated beam centered over the FPGA on the board. The board is connected to a long auxiliary power cable and a PCIe riser cable, which allows the FPGA development board to be placed some distance away from its accompanying host. The host was placed outside of the beam path so as to minimize failures unrelated to SEUs in the FPGA.

The greatest difference between the initial FI approach and the neutron radiation test is the ordering of events. In the initial FI approach, the introduction of faults, the execution of the acceleration application, and the scrubbing of faults all happened sequentially, one after the other. In neutron radiation testing, all of these events occur in parallel, asynchronously to each other. A comparison of this behavior is shown in Figure 7. Not only does this change mean that applications execute during periods without SEUs, but it also means that SEUs occur during periods

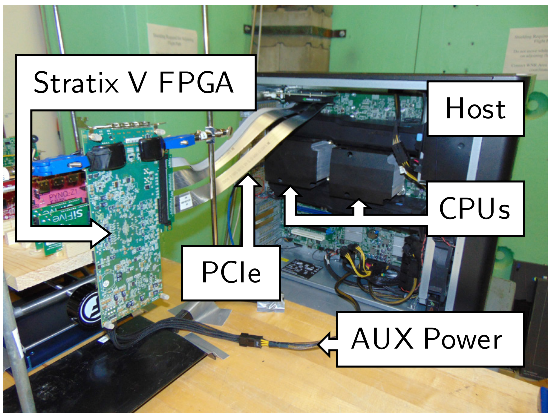

Fig. 6. Stratix V Neutron radiation test.

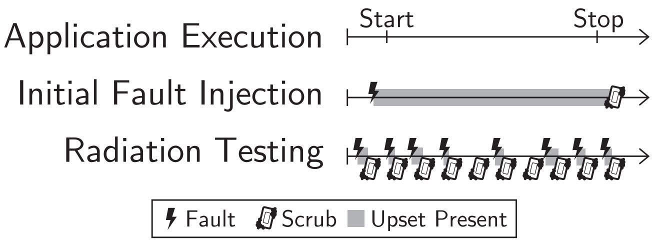

Fig. 7. Timeline comparison of events: initial fault injection verses radiation testing.

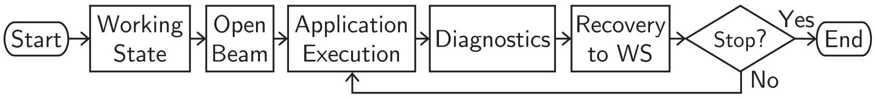

Fig. 8. Radiation testing flow.

when applications are not executing. Only the radiation present from an FPGA programming to a reprogramming or a power-cycle is included in the results.

Figure 8 shows the modified flow of the radiation test. First, the system is brought into a WS. Second, the beam shutter is opened, allowing the device to be exposed to the radiation. Next, the application is executed followed by some diagnostics. If a failure is detected, recovery attempts are made until the system returns to a WS. Recovery attempts escalate from re-running the application to see if configuration scrubbing was able to recover the FPGA design's functionality, to reprogramming the FPGA, to power-cycling the entire system. For radiation testing, scrubbing of SEUs in CRAM was performed asynchronously to the execution of the application primarily through internal configuration scrubbing. External configuration scrubbing was conducted on one of the applications with varying delays in between initiations of scrub cycles. The cycle of application execution, diagnostics, and recovery to WS is continued until sufficient data are collected.

The beam shutter is not cycled (closed and re-opened) between executions of the FPGA-host application. Instead, a detailed log of radiation exposure over time is used to exclude extraneous radiation exposure from the results. The results include radiation exposure that was incurred while the FPGA was ready for the execution of the FPGA-host application. Any radiation exposure

Table 7. Neutron Radiation Test Results for the Stratix V

Design	ALM Utilized	Fluence (n/cm^2)	Total Upsets	Total Failures	SDC	FPGA Unavailable	Host Unresponsive
Mandelbrot	74%	1.3×10^{10}	7,150	251	96.0%	2.4%	1.6%
Boardtest	25%	1.2×10^{10}	5,898	84	84.5%	13.1%	2.4%
Video Downscaling	22%	1.3×10^{10}	5,693	68	57.4%	41.2%	1.5%
Matrix Multiply	58%	3.8×10^{10}	18,349	85	85.9%	14.1%	0.0%
Sobel Filter	21%	1.8×10^{10}	7,407	46	67.4%	30.4%	2.2%
Vector Add	21%	1.7×10^{10}	6,710	33	75.8%	24.2%	0.0%
JPEG Decoder	41%	4.0×10^{9}	5,286	32	9.4%	90.6%	0.0%
External Scrub	74%	1.8×10^{9}	873	46	100.0%	0.0%	0.0%
External Scrub 1s Delay	74%	3.5×10^{9}	2,503	89	93.3%	6.7%	0.0%
External Scrub 15s Delay	74%	6.9×10^{9}	181	18	94.4%	5.6%	0.0%

incurred during an FPGA reprogramming or a power-cycle is excluded from the results. Excluding radiation exposure by physically cycling the beam stutter is impractical for an automated test because opening and closing the beam shutter must be done manually for safety.

Table 7 shows the results from neutron radiation testing. The name of each application tested is provided along with the percentage of ALMs utilized. The final grouping of tests are the results of external scrubbing applied to the Mandelbrot design (external scrubbing with no delay, 1 second delay, and 15 second delay between initialization of a scrub cycle). The total fluence or neutrons per cm^2 of radiation exposure was collected using LANSCE neutron dosimetry. Collected data were monitored real-time to estimate confidence intervals and distribute available beam time among test designs in an effort to tighten confidence intervals where needed. Thus, the fluence of radiation exposure varies between designs. Total upsets were recorded by continuously reading from the FPGAs internal SEU **error message register** (**EMR**). The EMR is updated during scrub cycles as upsets are encountered. Any SEU data collected is quickly stored to an on-chip buffer that is read out periodically. Total failures observed tallies the number of failures observed for any of the three failure modes. The distribution of failure is also provided.

Failures observed fall into three main failure modes: SDC, FPGA unavailable, and host unresponsive. SDC, as explained previously, occurs anytime incorrect data is returned by the FPGA and is not detected as corrupted by the system. In some scenarios, this failure mode can be devastating as it is the only failure mode that the system cannot detect and respond to. FPGA unavailable occurs when the host goes to execute the accelerator application but cannot correctly access the FPGA. Host unresponsive occurs when an upset causes the host to abruptly stop or no longer respond to any user input.

From the results, it is shown that SDC is the dominant failure mode. With the exception of the JPEG decoder design, 55–96% of all failures that occurred in the beam were SDC failures. SDC failures were detected by either running the same computation in software as was executed in the FPGA and comparing the results, or by comparing the results from the FPGA against a golden set of results stored on the hard drive of the host computer. FPGA unavailable and host unresponsive failure modes were also observed, but in far fewer quantities than SDC.

The greatest insights can be gained from neutron radiation testing by calculating the neutron cross section of a failure mode [3] and converting it to a failure rate. The cross section is determined by dividing the total upsets by the total failures and multiplying the result by percentage of failures corresponding to the desired failure type. The neutron cross section is converted to FIT by dividing the failure cross section by the total upset cross section (total upsets/total fluence, a constant for the device) and multiplying the result by the total device SEU FIT (from Table 2, a conversion of

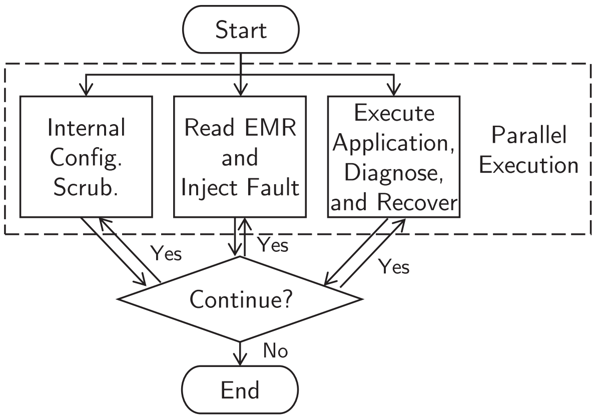

Fig. 9. Intermittent fault injection flow.

total device SEU cross section to FIT). A more accurate and direct conversion of neutron cross section to FIT is performed for SDC failures in Table 9. This table is used to compare the results from radiation testing against FI.

9 INTERMITTENT FAULT INJECTION

This is the third test of the three tests. This FI test was run after the previous radiation test. This test is an update from the first FI test in that the same FPGA applications and test setup are used, but a different approach is taken in the introduction of faults, scrubbing, and application execution. The first FI test allowed a single fault to be present throughout the entire execution of an application whereas this test allows faults to be introduced and scrubbed during application execution. This test is meant to more closely reflect the behavior of SEU in radiation testing with configuration scrubbing enabled. The results from this test should more closely reflect those of radiation testing and subsequently should more accurately estimate actual FIT rates (in situations where configuration scrubbing is employed).

The behavior of SEUs in neutron radiation testing (an intermittent presence that occurs asynchronously to application execution) prompted the development of a new FI scheme that would allow faults to be injected and scrubbed in parallel with the execution of the test applications. The initial FI approach, referred to as persistent FI, is effective at estimating the SEU response of a system with limited or no configuration scrubbing capability and infrequent device programming. In FPGA data center applications like cloud computing, reprogramming the FPGA, and enabling configuration scrubbing are likely to be more commonplace. Thus, a new FI approach, where SEU are present intermittently, is meant to better reflect the behavior of SEUs in FPGA data center deployments for cloud-computing-like applications.

The new FI scheme allows configuration scrubbing, FI, and application execution to all occur in parallel. Figure 9 depicts the updated flow of the FI campaign. When the test begins, internal configuration scrubbing is enabled, SEU data is collected from the EMR and new faults are injected, and the target application is executed with diagnostics and recovery as needed. The internal configuration scrubbing completes a cycle approximately every hundred milliseconds, SEU data is collected and a new fault is injected approximately every five to six seconds, and application execution and diagnosis complete every three to fifty-three seconds depending on the application being tested. Faults injected during recovery (reprogram or power cycle) are not considered. When sufficient data are collected, all three processes are halted and the test is ended.

Table 8. Intermittent FI Results

Design	Faults Injected	Any Failure (95% Confidence Interval)	SDC	FPGA Unavailable	Host Unresponsive
Mandelbrot	20,039	3.06% (2.82%, 3.30%)	2.95%	0.10%	0.01%
Boardtest	139,388	1.01% (0.96%, 1.06%)	0.87%	0.13%	0.01%
Matrix Multiply	15,051	0.16% (0.10%, 0.22%)	0.15%	0.01%	0.00%
Video Downscaling	12,208	0.64% (0.50%, 0.78%)	0.41%	0.21%	0.02%
Sobel Filter	27,494	0.31% (0.24%, 0.38%)	0.26%	0.04%	0.01%
Vector Add	18,211	0.29% (0.21%, 0.37%)	0.23%	0.05%	0.01%
JPEG Decoder	56,552	0.77% (0.70%, 0.84%)	0.12%	0.64%	0.01%

The intermittent FI campaign was conducted on all of the designs tested in neutron radiation. The results are shown in Table 8. The results follow the same format as those of the initial persistent FI campaign. This data represents 23 days of continuous data collection distributed among the different designs. The prevalence of SDC over other failures remains present in these results, but the percentages of randomly injected faults that result in a failure are much lower.

This change in fault behavior (mimicked SEUs) resulted in what appears to be far fewer interactions between faults and corresponding test stimulus that would result in a failure. With faults being present on the order of tens to hundreds of milliseconds, there is less opportunity for faults to co-exist with the execution of test vectors that would result in a failure. This is a plausible explanation for the reduction in failure percentage compared to the initial persistent FI approach. It is anticipated that this new FI approach will better approximate what is observed in neutron radiation testing.

10 COMPARISON OF COMBINED RESULTS

The results for SDC failures from initial FI, neutron radiation testing, and intermittent FI are all converted to FIT for comparison in Table 9. The SDC cross section (total failures divided by total fluence multiplied by the percentage of failures that were SDC) was converted to NYC FIT directly by taking the product of the NYC high-energy neutron flux per hour, one-billion (for hours of operation), and the neutron cross section. FI results were converted to FIT by multiplying the percentage of randomly injected faults that resulted in SDC failure by the whole device CRAM FIT rate found in Table 2. The cross section and estimated FIT (for radiation testing and FI) is accompanied by 95% confidence intervals that are calculated using conventional means [17, 18]. Ratios between initial FI and neutron testing estimated FIT and between neutron testing and intermittent FI estimated FIT are also included.

Initial fault injection FIT estimates for SDC failures are 3–22× larger than those estimated through neutron radiation testing. This result suggests that SDC failures are far less likely to occur when CRAM is continuously scrubbed and periodically refreshed via reprogramming. Through these results, some of the benefits of configuration scrubbing can be seen.

The intermittent FI results for SDC FIT are within a factor 1.3–2.9× the rates as measured through neutron radiation testing. All of these estimates are lower than the failure rate measured by radiation. This is expected in part because radiation testing can induce upsets in a larger superset of the device (block memories, registers, etc.). The results of intermittent FI mirror the results of radiation testing much more closely compared to the initial FI approach.

Table 10 shows the cross section and estimated NYC FIT of SDC failures from neutron radiation testing of the Mandelbrot design using external configuration scrubbing and three different periods of delay between scrub cycle initiations (no delay, 1 second, and 15 seconds). Bearing the overlap

Table 9. Silent Data Corruption Neutron Cross Section and FI Sensitivity Comparison

Design	Cross Section (cm^2) (95% Confidence Interval)	NYC FIT	Est. FIT from Persistent FI		Est. FIT from Intermittent FI	
Mandelbrot	1.9×10^{-8} $(1.6 \times 10^{-8}, 2.1 \times 10^{-8})$	243 (212, 274)	686 (648, 725)	2.8×	179 (165, 193)	1.4×
Boardtest	5.8×10^{-9} $(4.4 \times 10^{-9}, 7.1 \times 10^{-9})$	75 (58, 92)	206 (187, 226)	2.7×	53 (50, 56)	1.4×
Video Downscaling	3.3×10^{-9} $(2.3 \times 10^{-9}, 4.5 \times 10^{-9})$	43 (30, 59)	182 (163, 201)	4.2×	25 (18, 32)	1.7×
Matrix Multiply	1.9×10^{-9} $(1.5 \times 10^{-9}, 2.4 \times 10^{-9})$	25 (19, 31)	546 (520, 572)	21.6×	9 (5, 13)	2.9×
Sobel Filter	1.7×10^{-9} $(1.1 \times 10^{-9}, 2.4 \times 10^{-9})$	22 (15, 31)	79 (62, 95)	3.6×	16 (12, 20)	1.4×
Vector Add	1.5×10^{-9} $(9.5 \times 10^{-10}, 2.2 \times 10^{-9})$	19 (12, 28)	121 (106, 137)	6.3×	14 (10, 19)	1.3×
JPEG Decoder	7.6×10^{-10} $(1.5 \times 10^{-10}, 2.2 \times 10^{-9})$	10 (2, 29)	206 (182, 230)	20.9×	7 (5, 9)	1.4×

Table 10. Silent Data Corruption Neutron Cross Section: Mandelbrot with External Scrub

Design	Cross Section (cm^2) (95% Confidence Interval)	NYC FIT
No Delay	2.6×10^{-8} $(1.9 \times 10^{-8}, 3.5 \times 10^{-8})$	340 (248, 453)
1s Delay	2.4×10^{-8} $(1.9 \times 10^{-8}, 2.9 \times 10^{-8})$	312 (245, 379)
15s Delay	2.5×10^{-8} $(1.4 \times 10^{-8}, 3.9 \times 10^{-8})$	319 (186, 510)

in confidence intervals in mind, the estimated FIT is slightly larger than with internal scrubbing and there is little variation in FIT between the different settings. This suggests similarities between the benefit of external verses internal configuration scrubbing, but additional study is required.

Scaling failure rates to a large-scale deployment of FPGAs in a realistic environment adds perspective to the results. Table 11 scales the single instance SDC FIT normalized to NYC flux (see Table 9) to a large-scale deployment of FPGAs in a hypothetical data center located in Denver, Colorado, (3.8× NYC high-energy neutron flux). The scaling uses the SDC FIT estimate obtained from neutron radiation testing applied to a 100,000 node deployment.

The MTTF periods for SDC in a scaled environment from Table 9 have the same outcome comparison as the normalized FIT rates in Table 9. The scaled SDC MTTF periods estimated through

Table 11. SDC MTTF on a 100,000 Node System in Denver, CO

Design	FIT	MTTF (Days)
Mandelbrot	92,340,000	0.5
Boardtest	28,500,000	1.5
Video Downscaling	16,340,000	2.5
Matrix Multiply	9,500,000	4.4
Sobel Filter	8,360,000	5.0
Vector Add	7,220,000	5.8
JPEG Decoder	3,800,000	11.0

radiation testing are 3–22× longer than were originally estimated through persistent FI [13]. This likely results from the use of configuration scrubbing, which decreases the period of SEU presence thereby decreasing the likelihood of SEU and stimulus concurrence thereby decreasing the likelihood of failure. Scaled MTTF estimates obtained through intermittent FI are within a factor of 1.3–2.9× the MTTF estimates obtained through radiation testing.

In the hypothetical large-scale deployment of FPGAs in a data center setting, SDC would occur twice a day to once every 11 days on average. This is much more frequent than would be experienced by a single node. For comparison, this equates to a single node experiencing an SDC every 45 thousand to 1.1 million machine days on average. In a large-scale deployment, SDC failures may occur with enough frequency to cause significant disruption of service or loss of confidence in results. This type of failure is observed to occur more frequently and it is harder to detect and respond to than FPGA unavailable or host unresponsive failures.

The other failure modes (FPGA unavailable and host unresponsive) will occur less frequently in a large-scale system than SDC, but they will still occur. Using results from neutron radiation testing in the hypothetical system (one-hundred thousand FPGAs in Denver, Colorado), FPGA unavailable failures would have a 2.3–36 million FIT with a 9 million FIT average (4.7 days MTTF); host unresponsive would have up to a 1.5 million FIT with an average of 430 thousand FIT (3 months MTTF). These failures occur much less frequently than SDC and they are easier to detect and respond to. Even though they still do occur, they may be of less concern than SDC failure events.

11 CONCLUSION

In this article, several FPGA accelerator applications were tested for SEU sensitivity through FI and neutron radiation testing. Comparing neutron radiation testing against persistent FI results show a 3–22× reduction in SDC occurrence when internal configuration scrubbing is enabled. This suggests that significant benefit may be gained by enabling configuration scrubbing in large-scale deployments of SRAM-based FPGAs. An improved FI flow is demonstrated to better reflect the behavior of upsets in the neutron test environment when internal configuration scrubbing is enabled. SDC results from the updated FI approach are within a factor of 1.3–2.9× that of neutron radiation testing. It is observed also that configuration scrubbing reduces the occurrence of failures and increases the likelihood of system restoration after an SDC failure occurs.

SDC failures were observed far more frequently than other failures modes. The SDC results from neutron radiation testing are scaled to reflect expected behavior in a hypothetical system with 100,000 FPGAs deployed in a data center located in Denver, Colorado. In such a setting, SEUs would occur on average every half-hour and SDC failures would occur on average once every 0.5–11 days. For some applications, this could represent a significant venerability.

REFERENCES

[1] aws. 2018. Amazon EC2 F1 Instance Expands to More Regions, Adds New Features, and Improves Development Tools. Retrieved June 29, 2019 from https://aws.amazon.com/about-aws/whats-new/2018/10/amazon-ec2-f1-instance-expands-to-more-regions-adds-new-features-and-improves-development-tools/.

[2] J. Arram, Wayne Luk, and Peiyong Jiang. 2015. RAMETHY: Reconfigurable acceleration of bisulfite sequence alignment. In *Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. ACM, New York, NY, 250–259.

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

[4] R. C. Baumann. 2001. Soft errors in advanced semiconductor devices - Part I: The three radiation sources. *IEEE Transactions on Device and Materials Reliability* 1, 1 (Mar 2001), 17–22.

[5] Adrian Caulfield, Eric S. Chung, Andrew Putnam, Hari Angepat, Jeremy Fowers, Michael Haselman, Stephen Heil, Matt Humphrey, Puneet Kaur, Joo-Young Kim, Daniel Lo, Todd Massengill, Kalin Ovtcharov, Michael Papamichael, Lisa Woods, Sitaram Lanka, Derek Chiou, and Doug Burger. 2016. A cloud-scale acceleration architecture. In *Proceedings of the 2016 49th Annual IEEE/ACM International Symposium on Microarchitecture*. IEEE, 75–87.

[6] M. Ceschia, M. Violante, M. S. Reorda, A. Paccagnella, P. Bernardi, M. Rebaudengo, D. Bortolato, M. Bellato, P. Zambolin, and A. Candelori. 2003. Identification and classification of single-event upsets in the configuration memory of SRAM-based FPGAs. *IEEE Transactions on Nuclear Science* 50, 6 (2003), 2088–2094.

[7] Deloitte. 2017. Hitting the Accelerator: The Next Generation of Machine-Learning Chips. Retrieved December 12, 2018 from https://www2.deloitte.com/content/dam/Deloitte/global/Images/infographics/technology mediatelecommunications/gx-deloitte-tmt-2018-nextgen-machine-learning-report.pdf.

[8] B. Frank. 2017. Microsoft Unveils Brainwave, a System for Running Super-Fast AI. Retrieved December 12, 2018 from https://venturebeat.com/2017/08/22/microsoft-unveils-brainwave-a-system-for-running-super-fast-ai/.

[9] Intel Corporation 2020. *Stratix V Device Handbook, Volume 1: Device Interfaces and Integration*. Intel Corporation. Retrieved from https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/stratix-v/stx5_core.pdf.

[10] JEDEC Solid State Technology Association. 2006. *Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices*. JEDEC Solid State Technology Association. Retrieved December 12, 2018 from https://www.jedec.org/sites/default/files/docs/JESD89A.pdf.

[11] A. Keller, T. A. Whiting, K. B. Sawyer, and M. J. Wirthlin. 2018. Dynamic SEU sensitivity of designs on Two 28-nm SRAM-Based FPGA architectures. *IEEE Transactions on Nuclear Science* 65, 1 (2018), 280–287.

[12] A. Keller, J. Anderson, M. Wirthlin, S.-J. Wen, R. Fung, and C. Chambers. 2020. Using partial duplication with compare to detect radiation-induced failure in a commercial FPGA-Based networking system. In *Proceedings of the 2020 IEEE International Reliability Physics Symposium*. 651–656.

[13] Andrew M. Keller and Michael J. Wirthlin. 2019. Impact of soft errors on large-scale FPGA cloud computing. In *Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. ACM, New York, NY, 272–281.

[14] A. Lesea, S. Drimer, J. J. Fabula, C. Carmichael, and P. Alfke. 2005. The Rosetta experiment: Atmospheric soft error rate testing in differing technology FPGAs. *IEEE Transactions on Device and Materials Reliability* 5, 3 (Sep 2005), 317–328. DOI: https://doi.org/10.1109/TDMR.2005.854207

[15] K. Morgan, M. Caffrey, P. Graham, E. Johnson, B. Pratt, and M. Wirthlin. 2005. SEU-induced persistent error propagation in FPGAs. *IEEE Transactions on Nuclear Science* 52, 6 (Dec 2005), 2438–2445. DOI: https://doi.org/10.1109/TNS.2005.860674

[16] E. Nurvitadhi, G. Venkatesh, J. Sim, D. Marr, R. Huang, J. Ong Gee Hock, Y. Tat Liew, K. Srivatsan, D. Moss, S. Subhaschandra, and G. Boudoukh. 2017. Can FPGAs beat GPUs in accelerating next-generation deep neural networks? In *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. ACM, New York, NY, 5–14.

[17] Heather Quinn. 2014. Challenges in testing complex systems. *IEEE Transactions on Nuclear Science* 61, 2 (Apr 2014), 766–786. DOI: https://doi.org/10.1109/TNS.2014.2302432

[18] H. Quinn, D. A. Black, W. H. Robinson, and S. P. Buchner. 2013. Fault simulation and emulation tools to augment radiation-hardness assurance testing. *IEEE Transactions on Nuclear Science* 60, 3 (2013), 2119–2142.

[19] H. Quinn and P. Graham. 2005. Terrestrial-based radiation upsets: A cautionary tale. In *Proceedings of the 13th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*. 193–202. DOI: https://doi.org/10.1109/FCCM.2005.61

[20] P. Ramachandran, P. Kudva, J. Kellington, J. Schumann, and P. Sanda. 2008. Statistical fault injection. In *Proceedings of the 2008 IEEE International Conference on Dependable Systems and Networks with FTCS and DCC*. IEEE, 122–127.

[21] E. Schadt, Michael D. Linderman, Jon Sorenson, Lawrence Lee, and Garry P. Nolan. 2010. Computational solutions to large-scale data management and analysis. *Nature Reviews Genetics* 11, 9 (2010), 647–657.

[22] A. Silburt, A. Evans, A. Burghelea, S.-J. Wen, D. Ward, R. Norrish, and D. Hogle. 2008. Specification and verification of soft error performance in reliable internet core routers. *IEEE Transactions on Nuclear Science* 55, 4 (2008), 2389–2398.

[23] I. Stamoulias, C. Kachris, and D. Soudris. 2017. Hardware accelerators for financial applications in HDL and High Level Synthesis. In *Proceedings of the 2017 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation.* 278–285.

[24] A. Stoddard, A. Gruwell, P. Zabriskie, and M. J. Wirthlin. 2017. A hybrid approach to FPGA configuration scrubbing. *IEEE Transactions on Nuclear Science* 64, 1 (Jan 2017), 497–503. DOI: https://doi.org/10.1109/TNS.2016.2636666

[25] D. Thomas, Lee Howes, and Wayne Luk. 2009. A comparison of CPUs, GPUs, FPGAs, and massively parallel processor arrays for random number generation. In *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays*. ACM, New York, NY, 63–72.

[26] M. Wirthlin. 2015. High-reliability FPGA-Based systems: Space, high-energy physics, and beyond. *Proceedings of the IEEE* 103, 3 (2015), 379–389.

[27] Xilinx Inc. 2018. *Device Reliability Report*. Xilinx Inc. Retrieved December 12, 2018 from https://www.xilinx.com/support/documentation/user_guides/ug116.pdf.

Received January 2021; accepted March 2021