Terrestrial Cosmic Ray Induced Soft Errors and Large-Scale FPGA Systems in the Cloud

Andrew M. Keller and Michael J. Wirthlin
NSF Center for Space, High-Performance, and Resilient Computing (SHREC)
Department of Computer and Electrical Engineering
Brigham Young University
Provo, Utah 84602
{andrewmkeller, wirthlin}@byu.edu

***Abstract*—Radiation from outer space can cause soft errors in microelectronic devices deployed at terrestrial altitudes on Earth. Cosmic rays entering the Earth's atmosphere create a complex cascade of radioactive particles. The most likely form of cosmic radiation to cause soft errors in microelectronics at terrestrial levels are neutrons. SRAM-based FPGAs are susceptible to terrestrial cosmic ray induced soft errors. These soft errors occur infrequently for a single device deployed at terrestrial altitudes. When many FPGAs are deployed in a large-scale system, the impact of these soft errors on reliability can be significant. This study examines terrestrial cosmic ray induced soft errors and the effects they can have on large-scale deployment of FPGAs in cloud computing. Fifteen data-center-like designs were tested for sensitivity through fault injecting. Sensitivities ranged from less than 1% to about 12% of randomly injected faults resulting in unacceptable behavior. A hypothetical but realistic large-scale FPGA system, with 100,000 node deployed at a high-altitude, running the most sensitive design would experience the dominant failure mode of silent data corruption every 3.8 hours on average. This system would only be able to retain reliability level above 0.99 for about two minutes. Some soft error detection and recover approaches are discussed.**

I. Introduction

As cosmic rays enter Earth's atmosphere, they create a complex cascade of radioactive particles [1]. Less than 1% of the primary cosmic ray flux reaches sea level [2]. The rest reacts with particles in the atmosphere to produce muons, protons, neutrons, pions, and other particles. Muons and pions are short-lived and the Earth's magnetosphere greatly attenuates or traps charged particles, which leaves neutrons as the most likely cosmic radiation to cause soft errors in microelectronic devices deployed at terrestrial altitudes [2].

SRAM-based field programmable gate arrays (FPGAs) are susceptible to terrestrial cosmic ray induced soft errors [3]. Soft errors do not cause permanent damage to the device, but they can corrupt the values stored in configuration memory. When configuration memory is corrupted, the circuit operating on the FPGA can be altered. Alteration to the operating circuit can result in unintended behavior. In this way, terrestrial cosmic ray induced soft errors can cause circuits implemented on SRAM-based FPGAs to fail.

For low yield deployments of FPGAs in terrestrial applications that are not mission critical, cosmic radiation may not present a significant challenge. The likelihood of cosmic radiation inducing failure in a single FPGA deployment at terrestrial altitudes is extremely low. On a single moderately sized FPGA in terrestrial altitudes, upsets in configuration memory caused by cosmic radiation may occur only once ever twenty years on average and only a subset of the upsets that occur will actually cause the operating design to fail.

One the other hand, terrestrial cosmic ray induced soft errors present a significant challenge to large-scale FPGA deployments. Large-scale FPGA systems are more sensitive to terrestrial cosmic radiation ray induced soft errors because they include a larger number of FPGAs. Previous work has explored how FPGA soft error rates scale with changes in deployment quantity and location [4]. As more FPGAs are deployed, the risk of cosmic radiation causing failure increases.

The study presented in this paper examines the impact of terrestrial cosmic ray induced soft errors on large-scale FPGA systems within cloud computing. Today, FPGAs are used in large-quantities in cloud computing systems throughout the world to perform application specific computation. Soft errors caused by radiation have been observed in these large-scale systems [5]. This study examines the actual risk that these soft errors pose. Experimental data and material used in this paper are taken from a recent publication of the presented study [6].

The paper introduces terrestrial cosmic ray induced soft errors and their effects on FPGA designs. It presents the observed response of several data-center-like FPGA designs to upsets in configuration memory through fault injection. Fault injection data is collected from an example FPGA compute node that has an Intel Stratix V GX A7 FPGA connected to a host CPU via PCIe. The observed response is scaled to reflect what might be observed in a hypothetical but realistic large-scale FPGA system.

Tested designs ranged in sensitivity from less than 1% to almost 12% of randomly injected faults causing a failure. Silent data corruption (SDC), or the corruption of data without the system being aware, was identified as the dominant failure mechanism. A large-scale system running a design with 12% percent sensitivity on a hundred-thousand nodes at a high altitude, (i.e., Denver, Colorado), would experience SDC every 3.75 hours and could only operate for about two minutes with a probability of no SDC greater than 99%. This results does

not consider the application of fault tolerance techniques but some soft error detection and recovery schemes are discussed.

II. Terrestrial Cosmic Ray Induced Soft Errors

Radiation from outer space can cause failure in microelectronic devices deployed here on Earth. Space is a harsh radiation environment with heavy-ion particles from the explosion of novas and supernovas, high-energy electrons traveling near the speed of light, and protons and other particles originating from the sun [7]. These cosmic rays enter our atmosphere and create a complex cascade of secondary particles, of which neutrons are the most likely to cause soft errors in microelectronic devices.

The earth's atmosphere protects its inhabitants, including microelectronics, from radiation sources found in space. Figure 1 depicts the Earth's magnetosphere deflecting and collecting charged particles traveling towards the Earth. The atmosphere does not completely prevent radiation from reaching Earth's surface, but it does act as a filter and greatly attenuates the amount of radiation present on the surface compared to radiation in space. Thus electronics deployed in terrestrial environments are less likely to experience cosmic ray induced failure then those used in space or avionic applications [7].

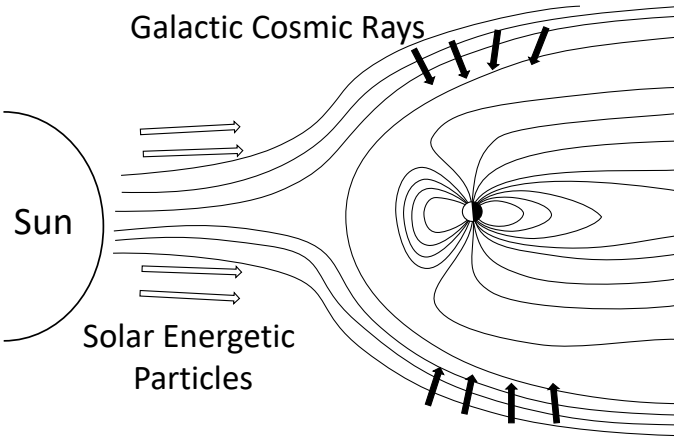

Fig. 1. Sources of Ionizing Radiation. Adapted from [8].

When a cosmic ray enters the Earth's atmosphere, several chain reactions may occur before resulting particles reach Earth's surface. It is possible for a primary cosmic ray to reach the Earth's surface, but that occurs for less than 1% of all primary cosmic rays entering the atmosphere. Figure 2 depicts the reaction of a primary cosmic ray with particles in the Earth's atmosphere. When the cosmic ray collides with a particle in the atmosphere a complex cascade of secondaries results, which include pions, muons, protons, neutrons and other particles. This shower of particles can in turn react with other particles, creating tertiary particles, and this pattern repeats until the array finally reaches the Earth's surface. Charged particles get attenuated by or trapped in the Earth's magnetosphere. Pions and Muons are short-lived. Thus, the most likely form of cosmic radiation to cause soft errors in a microelectronic device are neutrons [2].

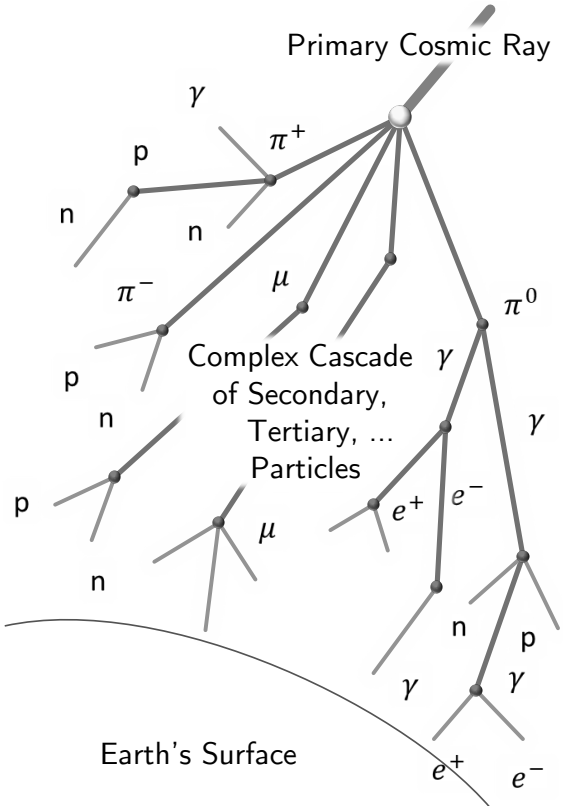

Fig. 2. Complex Cascade of Particles from Cosmic Ray Reactions. Adapted from [1].

When a single energetic particle passes through a microelectronic device, it can deposit a charge that causes an observable change in the device. This response is known as a single event effect (SEE) [9]. There are many types of SEEs. Some are destructive such as a single event gate rupture. Others do not cause permanent damage to the device such as single event transients, which appear as short lived power glitches unless latched into a memory element. The most common SEE in SRAM based FPGAs are single event upsets (SEUs) where values stored in memory elements are inverted by the particle strike.

Radiation effects effect in microelectronic devices are primarily due the funneling phenomenon [7]. This phenomenon is depicted in Figure 3 and is where an energetic particle passes through the device and deposits sufficient charge to alter the flow of current through the device. Neutrons themselves do not carry a charge. When they collide with other particles in the device, the reaction releases high-energy ionizing secondary particles. It is these secondary particles that cause SEUs in the target device, not the neutrons themselves [2].

Cosmic ray neutrons are not the only source of radiation that can cause soft errors in microelectronic devices here on Earth. Alpha particles from contaminants in device packaging and material can also cause soft errors [2]. Both sources of radiation should be considered when measuring the effects of radiation on microelectronic devices deployed in terrestrial ap-

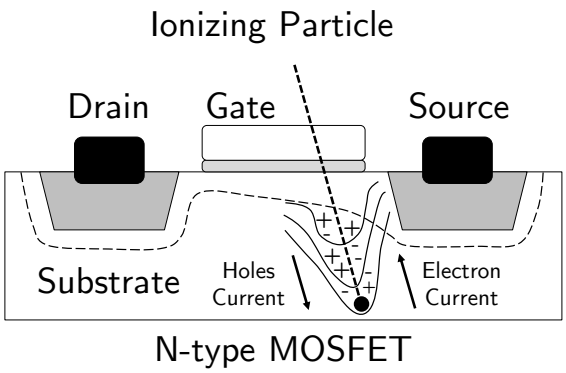

Fig. 3. The Funneling Phenomenon in an N-Type MOSFET. Adapted from [7].

plications [9]. Emphasis is given in this paper to high-energy, (greater than 10 MeV), neutrons due to their prevalence and variation based on deployment location.

Because cosmic ray and other sources of radiation can induce soft errors into microelectronic devices, standardized test procedures and metrics have been develop to quantify the vulnerability of a component to terrestrial radiation induced failure. The JEDEC 89A test standard for, "Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices," provides a standardized way to test microelectronic devices for the effects of terrestrial radiation. This standard helps ensure device safety and regulates awareness of the effects of terrestrial radiation on microelectronic devices. Microelectronic devices, including FPGAs, are typically tested by vendors to determine soft error rates for a given device.

A. Soft Error Rates

The reference value for cosmic-ray-induced high-energy neutron flux is 13 neutrons per cm^2 per hour. This value is the high-energy neutron flux in New York City (NYC,) at sea level, outdoors, during a time of average solar energy. This is the reference flux level that most reports scale soft error rates to, but high energy neutron flux can vary with altitude, location in the geomagnetic field, and solar magnetic activity [9]. Table I shows the relative high-energy neutron flux at various locations. Increase in altitude, included in the table, increases neutron flux exponentially. From this data, taken from [9], it is clear that some locations have significantly higher neutron flux than other. The White Mountain Research Station, at 12,500 feet in altitude, has a relative high-energy neutron flux that is 15× greater than the NYC flux.

When a device is tested using an accelerated neutron radiation beam, the cross section of a failure mode is measured by dividing the total number of failures that occur by the total amount of fluence or radiation exposure. Cross section is a hypothetical target area that will result in failure if a particle cross through it and fluence is the total number of particles that pass through a given area (usually cm^2) [7]. Many considerations must be made when testing a complex system with

TABLE I
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

radiation testing [10], but a test can be developed to measure the cross section of a single upset in the configuration memory of an FPGA. This cross section can be scaled to estimate the soft error rates for SRAM-based FPGA configuration upsets in different locations throughout the globe.

Two common metrics for reporting soft error rates are failure in time (FIT) and mean time to failure (MTTF). FIT is defined the average number of failures in one billion hours of operation [9], and MTTF is defined as the average amount of time from working operation to failure. The two metrics are inversely related, a higher MTTF means a lower FIT and visa-versa. A FIT of 1000 roughly corresponds to a 100 year MTTF. Soft error rates for large memory arrays, such as the configuration memory in an SRAM-based FPGA, are often reported in terms of FIT/Mbit, (i.e., 10^6 bits).

Cross section measurements from accelerated neutron radiation testing of FPGA configuration memory can be converted to FIT/Mbit by scaling the results to the amount of neutron radiation present in the target environment.

$$\frac{\text{FIT}}{\text{Mbit}} = 10^9 \text{ hours} \times \frac{\text{Neutrons}}{\text{cm}^2 \text{ hour}} \times \text{Cross Section} \times \frac{10^6 \text{ bits}}{\text{Megabit}}$$

Using this conversion, the NYC neutron soft error rate for configuration upsets in two similar 28-nm FPGAs is shown in Table II. Cross section data for this table was obtained from [3] and [11]. As can be seen from the table, the mean time to upset (MTTU) for a single deployment of one of these moderately sized FPGAs at NYC sea level is about 20 years. This soft error rate scales with the relative neutron flux of the deployed location (see Table I).

TABLE II
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

The number of devices deployed in a system also increases the overall occurrences of upsets in the system. The soft error rate of a single device does not change as more devices are deployed, but having more devices deployed increases the overall soft error rate. Table III show how the MTTU rate scales linearly with the number of FPGA devices deployed in a large-scale system. A single Stratix V GX A7 FPGA has about a twenty year MTTU at NYC sea level, but a large-scale

system with 100,000 of these FPGAs would experience a 1.5 hour MTTU at NYC sea level.

TABLE III
STRATIX V GX A7 MEAN TIME TO UPSET AT NYC SER

FPGAs	Years	Days	Hours	Min.	Sec.	Total (Sec.)
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

B. Effects of Upsets on FPGA Designs

SRAM-based FPGAs are programmable computer chips that can be configured to implement custom hardware designs. They have an array of resources whose functionality is controlled by a set of configuration memory bits. These devices have a set of lookup tables and flip-flops that are used to implement combinational and sequential logic. They also can contain a large amount of I/O, digital signal processing resources, and other sets of custom resources. The components a design uses, how they should behave, and how they are connected are programmed into a bitstream that gets loaded into the configuration memory of the device. With the bitstream loaded, the FPGA will implement the associated design.

When an upset occurs in configuration memory, the underlying circuit implemented on the FPGA can become corrupted. Corruption of values used as the logic equations of lookup tables can produce erroneous output from combinational logic. Connections between components in routing that use programmable interconnects can be lost or created where they should not be. The behavior of flip-flops or other used components can be altered by corruption of the configuration bits that govern control signals. The configuration memory controls the implementation of the underlying circuit. Thus, corruption of configuration memory in an FPGA can have a dramatic effect on the correct functionality of the design [12].

Design state stored in user memory elements can also be corrupted by radiation induced upsets. Table IV shows a breakdown of state in a Stratix V GX A7 FPGA. Most of the state in an FPGA is dedicated to configuration memory (CRAM), but a significant portion is available for use as large block memories for users (BRAM), distributed memories (LUTRAM), or as user flip-flops. Some other memory elements, accessible from user logic, are use to dynamically configure component behavior. All of this state is susceptible to radiation induced upsets. An upset is any FPGA state may cause instability in the implemented design.

C. Architectural Vulnerability Factor

Not all upsets in configuration memory will cause an FPGA design to behave incorrectly. Correct functionality depends on the integrity of the underlying circuit. When an upsets occurs in configuration memory, it can alter the the underlying circuit, but it may also have no effect on the underlying circuit. Having an effect on the underlying circuit depends on if the upset bit is associated with the underlying circuit or not and several other factors. If an upset bit controls the logic equation of an unused lookup table or another unused resource, then it will have no effect on the underlying circuit. If an upset does alter the underlying circuit, masking behavior [13] may prevent the alteration from causing incorrect circuit behavior.

TABLE IV
BREAKDOWN OF STATE 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%

A main contribution of the presented study is identifying the architectural vulnerability factor for several data-center-like FPGA designs. The architectural vulnerability factor or (AVF) for an FPGA design is the percentage of random upsets that actually result in unacceptable behavior. This concept is similar to AVF found in computer architecture [14] and reflects a down scaling of raw soft error rates to match the radiation induced failure rate of the design operating on the FPGA.

Another major contribution of the presented work is a comparison of architectural vulnerability factor against the percentage of bits reported by vendor tools as being potentially used by the active design. Vendor tools can classify bits in configuration memory as being potentially used by the active design or not. Intel Quartus Prime can generate a mapping file that designates which bits are *critical* and which are not. User can select which hardware modules to include in the classification and can give unique tags to different regions. Designating a bit as critical means that it may be used by the implemented design, but it does not guarantee that an upset in this bit will cause the design to fail. The presented study explores the conservative nature of critical bit classification.

III. LARGE-SCALE FPGA SYSTEMS IN THE CLOUD

A single FPGA instance has a relatively low risk of cosmic ray induced design failure at terrestrial altitudes, but as more devices are deployed in large-scale systems at terrestrial altitudes with higher neutron flux, the risk of cosmic ray induced design failure increases. Large-scale FPGA systems in cloud computing provide an interesting case study in which to examine this behavior. Large-scale deployment of any technology increases the likelihood of a single node failing. Increase in soft errors to the point of system disturbance has been observed in DRAM [15], microprocessors [14], and even FPGAs [4].

Today, FPGAs are being used in large quantities in data centers throughout the world to perform application specific computations. As of August 2017, Microsoft has deployed hundreds of thousands of FPGAs in their data centers [16]. Amazon now offers FPGA compute nodes in four different

locations with three additional locations recently announced. Additionally, other companies, such as Nimbix, Baidu, and Micron, are involved with large-scale FPGA deployment in the cloud. FPGAs' presence in data centers for cloud computing is likely to continue and increase in prevalence [17].

It is common for FPGA nodes in cloud computing to be coupled with a host CPU or placed on a network as a globally accessible resource [5]. Figure 4 depicts a common FPGA co-processor configuration where an FPGA is attached to a host CPU via PCIe and shared DRAM resources are made available to the FPGA and host through controller logic on the FPGA. The FPGA design has a static shell used to support the needed infrastructure and communication and a region of the device is reserved and dedicated for different hardware kernels that will run on the device. This typical configuration was used in the experiments presented in this paper.

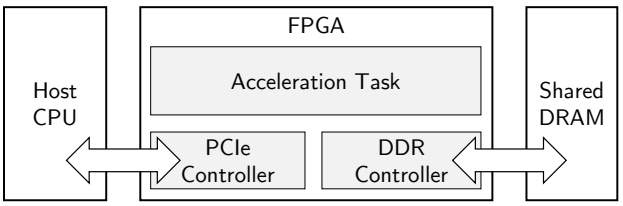

Fig. 4. Typical FPGA Data Center Node

When radiation upsets occur in a computation node such as this, one of three main failure modes may occur. First, the host may become unresponsive. Second, the FPGA may become inaccessible by the host. Third, data returned by the FPGA may be corrupted and the receiving system may be unaware of corruption. The third scenario is commonly referred to as silent data corruption or SDC. All of these failure modes were observed in fault injection, and a series of recovery options were employed to bring the system back into a working state.

While the presented study focuses on terrestrial cosmic ray included soft errors in large-scale FPGA systems within cloud computing, the presented material and conclusions can be applied to any large-scale deployment of FPGAs at terrestrial altitudes. Applications in the automotive, communications, (e.g., wired and wireless), and other industries can benefit from better understanding the risks that terrestrial cosmic ray induced soft error pose on large-scale terrestrial deployments of SRAM-based FPGAs.

IV. Fault Injection Experiments

Fault injection is a commonly used technique that emulates the effects of soft errors by purposefully corrupting the contents of configuration memory in FPGAs and observing the resulting behavior. It is used to augment radiation test results and to better understand the effects that radiation induced upsets can have on a circuit. When enough fault injection data is collected to be statistically significant, this data can be used to estimate the overall sensitivity or AVF of the targeted design [18]. Fault injection is used in the presented study to estimate the AVF of several data-center-like designs. Collected data is compared against vendor classification of bits used by the design and error rates are scaled to a hypothetical but realistic large-scale FPGA system.

A diverse set of fifteen benchmarks were included in the presented study. All of the benchmark designs used in the presented study originate from designs posted on the Intel FPGA SDK for OpenCL - Developer Zone [19] and the Terrasic OpenCL board support package for the DE5-Net FPGA developer board [20]. Table V lists each of the tested benchmarks and amount of logic and routing resources that they utilize. Designs range in size and complexity from a simple "Hello World" to large computational and memory intensive designs. Generally speaking, larger designs are expected to have a higher sensitivity to upset because they use more resources.

TABLE V
Benchmark Design Resource Utilization

Design	Total ALMs		Routing
FD3D	190,612	(81.21%)	30.50%
Mandelbrot	173,755	(74.03%)	29.40%
Channelizer	145,180	(61.85%)	23.40%
Matrix Multiply	135,405	(57.69%)	28.40%
FFT1D	129,767	(55.29%)	20.80%
FFT2D	121,015	(51.56%)	22.20%
JPEG Decoder	95,250	(40.58%)	17.70%
Compute Score	94,575	(40.29%)	22.10%
Boardtest	57,547	(24.52%)	11.90%
Video Downscaling	50,914	(21.69%)	10.80%
Vector Op	49,503	(21.09%)	9.90%
Vector Add	49,039	(20.89%)	9.70%
Sobel	48,573	(20.69%)	9.20%
Hello World	46,329	(19.74%)	8.60%

Related to design resource utilization, the total number of critical bits for each design is listed in Table VI. These bits are designated by vendor tools as possibly used by implemented designs. The percentage of critical bits should not be confused with a designs AVF. Critical bits provide an upper bound on the number of bits in the device that *could* cause design failure if upset. AVF provides an estimate on the percentage of configuration bits that will actually result in design failure if upset.

The fault injection experiments conducted in the presented study were performed using the setup shown in Figure 5. A Dell Precision T7610 server was used as the host. It was running Window 10 Professional and has two Intel Xeon processors and 16 GB of ECC protected RAM. The FPGA accelerator card used is a Terrasic DE5-Net with a Stratix V GX A7 FPGA. The test operator is an external computer that monitors design execution and injects faults into the FPGA as part of the fault injection flow.

The fault injection flow used in the presented study follows the basic fault injection algorithm presented in [18]. First, the system is brought into a working state. Then, a fault is injected into the system. Next, the target application is executed to stimulate the design under test followed by a series of diagnostics that determine if the design operated correctly or not. Observed behavior is recorded, the injected fault is

TABLE VI
CRITICAL BITS

Design	Injectable	Total Critical	
FD3D	98,502,636	69,740,486	(70.8%)
Mandelbrot	98,029,036	66,122,603	(67.5%)
Channelizer	98,534,636	62,673,556	(63.6%)
Matrix Multiply	98,573,036	58,774,590	(59.6%)
FFT1D	98,514,796	57,182,508	(58.0%)
FFT2D	98,439,276	56,797,420	(57.7%)
JPEG Decoder	98,386,796	41,360,019	(42.0%)
Compute Score	98,394,476	44,450,948	(45.2%)
Boardtest	98,578,156	24,009,160	(24.4%)
Video Downscaling	98,587,116	20,934,957	(21.2%)
Vector Op	98,549,996	19,589,044	(19.9%)
Vector Add	98,587,116	19,838,814	(20.1%)
Sobel Filter	98,597,996	19,067,514	(19.3%)
Hello World	98,603,107	18,132,374	(18.4%)

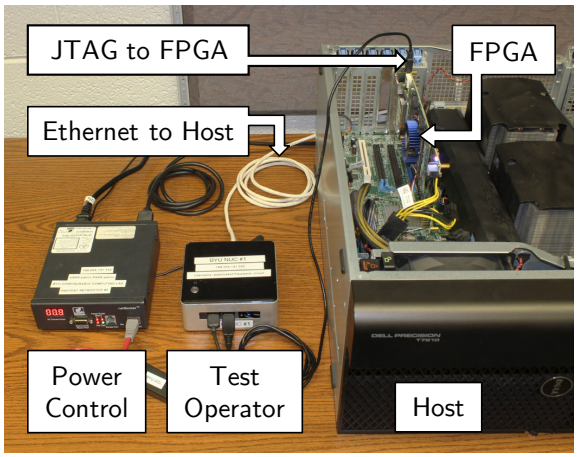

Fig. 5. Experiment Setup for Fault Injection Testing

repaired, and the system is recovered if necessary to bring it back into a working state. This cycle is repeated until sufficient data is collected.

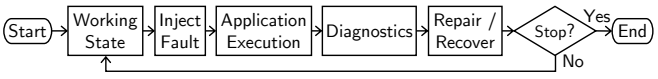

Fig. 6. Fault Injection Flow

In this experiment, faults were injected before executing the host application, the host application was allowed to run completely, and then the injected fault was removed. Diagnostics made sure the host application ran without error and that the output of data from the FPGA matched either a golden copy of data on the host's hard drive or matched the output of the same computations run again on the host CPU. When an injected fault did result in failure, recovery consisted of removing the injected fault from the device and trying again, reprogramming the FPGA and trying again, and power cycling the host and FPGA and trying again until the system returned back to a working state.

Results from the fault injection campaign are presented in Table VII. Here the logic resource utilization and the percentage of bits that are critical are shown side by side. While these metrics are correlated, having a higher resource utilization than another design does not guarantee having a higher percentage of critical bits. The AVF of any failure mode ranges from less than one percent to almost 12% and is broken down into the three major failure modes. Silent data corruption was found to be the dominant failure mode and the percentage of critical bits were found to overestimate the AVF of a given design by at least 5×.

V. FAILURE RATE FOR LARGE-SCALE SYSTEMS

The AVF for silent data corruption in Table VII is used in Table VIII to scale the hypothetical SDC failure rates of a the same designs running on a 100,000 node FPGA system. The hypothetical system is made up of Stratix V GX A7 FPGAs deployed at high-altitude in Denver, Colorado. In this environment, the most sensitive design would cause the system to experience an SDC event every 3.8 hours on average.

TABLE VIII
SDC MTTF ON A 100,000 NODE SYSTEM IN DENVER, CO

Design	FIT	MTTF
Mandelbrot	266,000,000	3.8 Hours
Matrix Multiply	212,000,000	4.8 Hours
FFT2D	200,000,000	5.0 Hours
FFT1D	193,000,000	5.2 Hours
FD3D	193,000,000	5.2 Hours
JPEG Decoder	80,000,000	12.5 Hours
Compute Score	125,000,000	8.0 Hours
Channelizer	120,000,000	8.3 Hours
Boardtest	80,000,000	12.5 Hours
Video Downscaling	71,000,000	14.2 Hours
Vector Op	61,000,000	16.3 Hours
Vector Add	47,000,000	21.2 Hours
Sobel Filter	31,000,000	1.3 Days
Hello World	2,000,000	2.5 Weeks

Although MTTF provides a useful metric for understanding the overall failure rate of a system, it does not adequately represent the fact that many failures in the system will occur in sooner than the MTTF estimate. Figure 7 shows that 63% of failures will occur in a time that is less than the MTTF. A better metric is to estimate the time in which the system can operate above a pre-specified level of reliability. This metric is called mission time. It is a function of the minimum reliability constraints given as a probability from zero to one. Table IX presents the mission time in seconds for various minimum reliability constraints of the most sensitive design running on the hypothetical system. The hypothetical system would be able to maintain a reliability greater than 0.99 for approximately 2 minutes. Such a short period may not be long enough for application tasks to complete and still meet reliability requirements.

VI. SOFT ERROR DETECTION AND RECOVERY

FPGA vendors provide methods for detecting upsets in configuration memory and repairing them. These features are often referred to as configuration memory scrubbing and are largely based on error correction codes (ECC). Other features are made available to users to mitigate the effects of soft errors

TABLE VII
FAULT INJECTION RESULTS, AVF FOR SPECIFIC BEHAVIORS (NORMALIZED TO ALM UTILIZATION)

Design	ALM Utilization	Critical Bits	Faults Injected	Silent Data Corruption	FPGA Unavailable	Host Unresponsive	Any Failure AVF	Critical Bits to AVF Ratio
Mandelbrot	74%	67%	9,301	11.3% (15.3%)	0.3% (0.4%)	0.02% (0.03%)	11.6% (15.7%)	5.8×
Matrix Multiply	58%	60%	17,094	9.0% (15.5%)	0.5% (0.9%)	0.03% (0.05%)	9.6% (16.6%)	6.2×
FFT2D	52%	58%	5,223	8.5% (16.4%)	0.5% (0.9%)	0.00% (0.00%)	9.0% (17.4%)	6.4×
FFT1D	55%	58%	7,389	8.2% (14.8%)	0.4% (0.7%)	0.18% (0.32%)	8.8% (15.9%)	6.6×
FD3D	81%	71%	8,094	8.2% (10.1%)	0.5% (0.6%)	0.00% (0.00%)	8.8% (10.8%)	8.0×
JPEG Decoder	41%	42%	7,948	3.4% (8.5%)	3.3% (8.2%)	0.14% (0.34%)	7.0% (17.2%)	6.0×
Compute Score	40%	45%	7,310	5.3% (13.2%)	1.2% (3.0%)	0.00% (0.00%)	6.6% (16.3%)	6.8×
Channelizer	62%	64%	10,709	5.1% (8.2%)	0.4% (0.6%)	0.03% (0.05%)	5.5% (8.9%)	11.6×
Boardtest	25%	24%	12,247	3.4% (13.7%)	0.6% (2.6%)	0.03% (0.13%)	4.1% (16.6%)	5.9×
Video Downscaling	22%	21%	11,641	3.0% (13.7%)	0.6% (2.9%)	0.04% (0.20%)	3.7% (17.1%)	5.7×
Vector Op	21%	20%	6,790	2.6% (12.2%)	0.4% (1.7%)	0.04% (0.21%)	3.0% (14.3%)	6.6×
Vector Add	21%	20%	11,623	2.0% (9.6%)	0.4% (1.8%)	0.04% (0.21%)	2.5% (11.7%)	8.0×
Sobel Filter	21%	19%	6,638	1.3% (6.2%)	0.5% (2.5%)	0.03% (0.15%)	1.9% (9.0%)	10.2×
Hello World	20%	18%	15,442	0.1% (0.4%)	0.2% (0.9%)	0.01% (0.03%)	0.3% (1.5%)	61.3×

TABLE IX
MISSION TIME FOR DIFFERENT RELIABILITY CONSTRAINTS

r	0.5	0.9	0.99	0.999	0.9999	0.99999
$MT(r)$ (sec)	9,446	1,436	137	13.6	1.36	.14

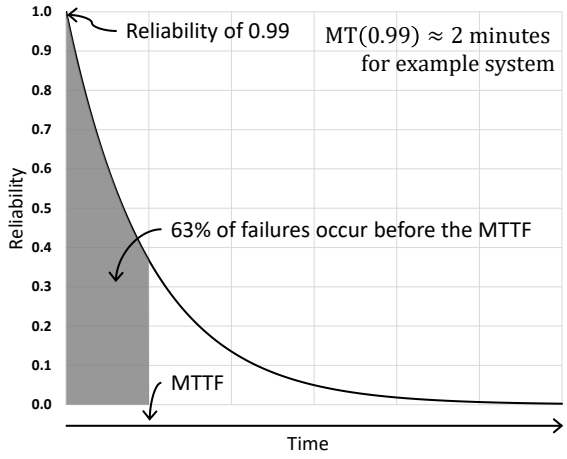

Fig. 7. Simplex System Reliability Over Time Verses MTTF

TABLE X
SDC SYSTEM RESTORATION

Design	Events	Scrub	Reprogram	Power Cycle
Mandelbrot	1,050	99.5%	0.5%	0.0%
Matrix Multiply	1,530	99.2%	0.8%	0.0%
FFT2D	442	99.1%	0.9%	0.0%
FFT1D	606	99.2%	0.8%	0.0%
FD3D	667	98.7%	1.3%	0.0%
JPEG Decoder	273	98.2%	1.1%	0.7%
Compute Score	390	97.2%	2.8%	0.0%
Channelizer	541	98.0%	2.0%	0.0%
Boardtest	412	97.1%	2.9%	0.0%
Video Downscaling	345	94.2%	5.8%,	0.0%
Vector Op	174	95.4%	4.6%	0.0%
Vector Add	233	98.3%	1.3%	0.4%
Sobel Filter	85	90.6%	8.2%	1.2%
Hello World	12	66.7%	8.3%	25.0%

on block memories. These features are made available to users, and signaling of upset occurrence is available in user logic. How the design responds to an upset event is left up to the users digression.

In this experiment, when a failure occurs, successive recovery attempts are made until the system is back in a working state. Removing the injected, which is similar to configuration scrubbing, is tried first followed by reprogramming the FPGA followed by power cycling both the FPGA and host. Table X shows the breakdown of recovery techniques that were successful for recovering the system from SDC failure occurrence. Scrubbing away the injected bit was very effective for restoring the system but did not restore the system in all cases. It is important to note that recovering the system to a working state does not undo the corruption of data that has already occurred.

In responding to SEUs in large-scale systems, designers can take one of several approaches. They could disable configuration scrubbing and respond only when a failure is detected. This may be useful in situations where SDC is not a great concern. They could enable scrubbing and only respond to detected failures. This is a low cost approach that would allow for more rapid system recovery. They could respond every time an upset is detected, or they could use vendor designated critical bits to only respond upsets in critical regions.

Care should be taken when responding to upsets that sufficient data is discarded when an upset is detected. Scrub cycles are periodic and upsets persist in configuration memory for half a scrub cycle on average before they are detected by the scrub engine. During that time, their presence can induce incorrect behavior in the design and produced data that should not be trusted. Figure 8 shows a timeline of upset and detection events that depicts this scenario. At position 1, a scrub cycle completes with no detected errors. At position 2, an upset occurs. At position 3, SDC propagates out of the device. At position 4, a subsequent scrub cycle completes, but already cleared the location of the upset before it occurred, so no upset was reported. At position 5, the scrub engine finally detects and corrects the upset. At position 6, the SDC either flushes out or persists, and at position 7, the third scrub cycle

completes and reports that an error was found.

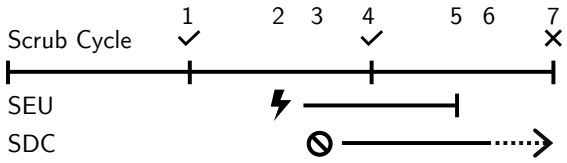

Fig. 8. Scrub cycles, SEU occurrence, and SDC

Several possible SEU response approaches approaches can be taken. It is advisable to incorporate vendor provided configuration scrubbing into large-scale FPGA systems in the cloud due to the benefit it provides and the low overhead it requires. In situations that require high-reliability or improved reliability, additional hardware fault tolerance techniques such as triple modular redundancy [21] or duplication with compare [22] should be applied. Awareness of the risk and available detection and recovery mechanisms can greatly assist large-scale FPGA system readiness to address terrestrial cosmic ray induced soft errors.

VII. Conclusion

In this paper, terrestrial cosmic ray induced soft errors and the effects they have on large-scale FPGA systems in the cloud are examined. The soft error rate for configuration upsets in a single device deployed at terrestrial altitudes is very low, (about once every twenty year on average for a moderately sized FPGA), but is significant for a large-scale FPGA system deployed in terrestrial altitudes. The presented study examines the architectural vulnerability factor, or the estimated percentage of configuration bits that will actually cause undesired behavior if upset, of 15 data-center-like FPGA designs via fault injection. AVF ranges from less than 1% to about 12% with SDC as the dominant failure mode, which is at least 5$\times$ smaller than the percentage of bits deemed potentially used by vendor tools.

Running the most sensitive design on a hypothetical but realistic large-scale FPGA system with 100,000 Stratix V GX A7 nodes at high-altitude in Denver, Colorado, would cause the system to experience SDC with a 3.8 hour MTTF. This system would be able to maintain a probability of no SDC greater than 0.99 for approximate two minutes. Several possible soft error response schemes were discussed and designers of large-scale FPGA systems are advised to incorporate vendor provided configuration scrubbing into their terrestrial cosmic ray induced soft error response approach. For application that require improved reliability, additional soft error fault tolerance techniques should be applied.

References

[1] J. F. Ziegler *et al.*, "IBM experiments in soft fails in computer electronics (1978–1994)," *IBM Journal of Research and Development*, vol. 40, no. 1, pp. 3–18, jan 1996. [Online]. Available: http://ieeexplore.ieee.org/document/5389432/

[2] R. C. Baumann, "Soft errors in advanced semiconductor devices - Part I: The three radiation sources," *IEEE Transactions on Device and Materials Reliability*, vol. 1, no. 1, pp. 17–22, mar 2001.

[3] Xilinx Inc., "Device Reliability Report, Second Half 2015."

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

[5] A. M. Caulfield *et al.*, "A cloud-scale acceleration architecture," in *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, oct 2016, pp. 1–13. [Online]. Available: http://ieeexplore.ieee.org/document/7783710/

[6] A. M. Keller and M. J. Wirthlin, "Impact of soft errors on large-scale FPGA cloud computing," in *Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, ser. FPGA '19. New York, NY, USA: ACM, 2019, pp. 272–281. [Online]. Available: http://doi.acm.org/10.1145/3289602.3293911

[7] N. Battezzati *et al.*, "Reconfigurable field programmable gate arrays: Failure modes and analysis," in *Reconfigurable Field Programmable Gate Arrays for Mission-Critical Applications*. Springer, 2011, pp. 37–83.

[8] "Sources of ionizing radiation in interplanetary space." [Online]. Available: https://photojournal.jpl.nasa.gov/select/PIA16938

[9] J. JEDEC, "Measurement and reporting of alpha particles and terrestrial cosmic ray-induced soft errors in semiconductor devices: Jesd89a," pp. 1–85, 2006. [Online]. Available: https://www.jedec.org/sites/default/files/docs/JESD89A.pdf

[10] H. Quinn, "Challenges in testing complex systems," *IEEE Transactions on Nuclear Science*, vol. 61, no. 2, pp. 766–786, apr 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6786369/

[11] A. M. Keller *et al.*, "Dynamic SEU sensitivity of designs on two 28-nm SRAM-based FPGA architectures," *IEEE Transactions on Nuclear Science*, vol. 65, no. 1, pp. 280–287, jan 2018. [Online]. Available: http://ieeexplore.ieee.org/document/8103796/

[12] M. Ceschia *et al.*, "Identification and classification of single-event upsets in the configuration memory of SRAM-based FPGAs," *IEEE Transactions on Nuclear Science*, vol. 50, no. 6, pp. 2088–2094, dec 2003. [Online]. Available: http://ieeexplore.ieee.org/document/1263846/

[13] A. L. Silburt *et al.*, "Specification and verification of soft error performance in reliable internet core routers," *IEEE Transactions on Nuclear Science*, vol. 55, no. 4, pp. 2389–2398, aug 2008. [Online]. Available: http://ieeexplore.ieee.org/document/4636892/

[14] S. Mukherjee *et al.*, "A systematic methodology to compute the architectural vulnerability factors for a high-performance microprocessor," in *Proceedings. 36th Annual IEEE/ACM International Symposium on Microarchitecture, 2003. MICRO-36.*, 2003, pp. 29–40.

[15] B. Schroeder, "DRAM errors in the wild: A large-scale field study," *Communications of the ACM*, vol. 54, no. 2, pp. 100–107, 2011.

[16] B. Frank, "Microsoft unveils Brainwave, a system for running super-fast AI," 2017. [Online]. Available: https://venturebeat.com/2017/08/22/microsoft-unveils-brainwave-a-system-for-running-super-fast-ai/

[17] Deloitte, "Hitting the accelerator: the next generation of machine-learning chips," 2017. [Online]. Available: https://www2.deloitte.com/content/dam/Deloitte/global/Images/infographics/technologymediatelecommunications/gx-deloitte-tmt-2018-nextgen-machine-learning-report.pdf

[18] H. Quinn *et al.*, "Fault simulation and emulation tools to augment radiation-hardness assurance testing," *IEEE Transactions on Nuclear Science*, vol. 60, no. 3, pp. 2119–2142, 2013.

[19] Intel, "Intel fpga sdk for opencl – developer zone," 2018. [Online]. Available: https://www.intel.com/content/www/us/en/programmable/products/design-software/embedded-software-developers/opencl/developer-zone.html

[20] Terasic, "Stratix V - DE5-Net FPGA development kit," 2018. [Online]. Available: https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=526

[21] B. Pratt *et al.*, "Improving FPGA design robustness with partial TMR," in *2006 IEEE International Reliability Physics Symposium Proceedings*, March 2006, pp. 226–232.

[22] J. Johnson *et al.*, "Using duplication with compare for on-line error detection in FPGA-based designs," in *2008 IEEE Aerospace Conference*, mar 2008, pp. 1–11.