Neutron Radiation Beam Results for the Xilinx UltraScale+ MPSoC

Jordan D. Anderson, Jennings C. Leavitt, Michael J. Wirthlin

***Abstract*—The paper summarizes the single-event upset (SEU) results obtained from neutron testing on the UltraScale+ MPSoC ZU9EG device. This complex device contains a large amount of programmable logic and multiple processor cores. Tests were performed on the programmable logic and the processing system simultaneously. Estimates of the single-event upset neutron cross section were obtained for the programmable logic CRAM, BRAM, OCM memory, and cache memories. During the test, no processor crashes or silent data corruptions were observed. In addition, a processor failure cross section was estimated for several software benchmark operating on the various processor cores. Several FPGA CRAM scrubbers were tested including an external JTAG, the Xilinx "SEM" IP, and the use of the PCAP operating in baremetal. In parallel with these tests, single-event induced high current events were monitored using an external power supply and monitoring scripts.**

I. Introduction

The Xilinx UltraScale+ Multi-Processor System-on Chip (MPSoC) is a device that combines programmable logic and multiple processors on a single die. Using 16nm FinFET technology, this device combines a large amount of programmable logic resources with multiple processors and a large number of fixed I/O interfaces [1]. The processing power and architectural flexibility make this device very attractive for a wide variety of embedded and high-performance computing applications including space processing. The objective of this paper is to measure and understand the sensitivity of various components of the MPSoC device to neutron radiation.

The ZU9EG was subjected to neutron radiation testing at the Los Alamos Neutron Science Center (LANSCE) in several different visits. A novel testing methodology was used to collect single-event effect (SEE) results simultaneously from the processing system (PS) and the programmable logic (PL). In parallel, high current events on the device power rails were monitored using an external power supply and real-time power monitoring. The neutron cross section of the following MPSoC architectural components was measured: FPGA CRAM, FPGA BRAM, on-chip memory (OCM), L1 and L2 caches, and the processor executing an operating system and bare metal benchmarks. While high current events were seen during testing, there were no single-event upsets that caused the processor to fail.

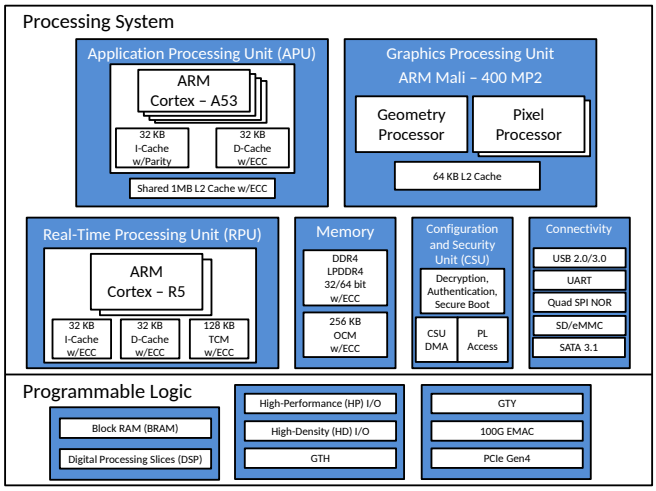

Fig. 1: Block diagram of ZU9EG MPSoC.

The remainder of the paper will be organized as follows: Section II provides an overview of the MPSoC device. Section III provides an overview of the experiment and Section IV explains the experimental setup with information about the LANSCE facility. Section V describes a study of SEU effects on the PL and Section VI summarizes a study performed on the SEU effects on the PS. Section VII discusses current events observed on the MPSoC during neutron testing. The paper concludes in Section VIII.

II. MPSoC System Overview

The objective of this section is to describe the architecture of the MPSoC and highlight the important components of the device. Fig. 1 shows a block diagram of the MPSoC. The architecture has two distinct regions: the processing system (PS) and the programmable logic (PL). The PS contains a number of processors that execute software from main memory and an external DDR memory. In addition, the device contains a large number of useful I/O interfaces and a 256KB on-chip memory (OCM). Together, these components provide a significant amount of computational performance and user customization.

The PS contains an application processing unit (APU), a real-time processing unit (RPU), and a graphics processing unit (GPU). The APU is comprised of a quad-core ARM

Manuscript received August 10, 2018. This work was supported by the I/UCRC Program of the National Science Foundation under Grant No. 1738550.

This work was also supported by Los Alamos Neutron Science Center (LANSCE) under proposals NS-2016-7268-F, NS-2017-7598-A, & NS-2017-7574-A.

The authors are with Department of Electrical and Computer Engineering, Brigham Young University, Provo, UT 84602, USA. The authors are also with the NSF Center for Space, High-Performance, and Resilient Computing (SHREC), (e-mails: jordan_anderson59@byu.edu, leavit92@byu.edu, wirthlin@byu.edu).

978-1-5386-8263-0/18/$31.00©2018 IEEE

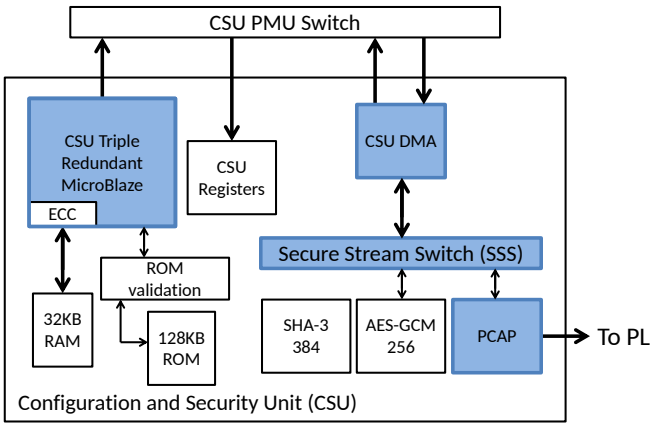

Fig. 2: Block diagram of CSU with features for PCAP access highlighted.

Cortex-A53 with each core containing a 32KB L1 instruction and data caches. All cores share a 1MB L2 cache. The L1 instruction cache is protected by parity to support error detection and the L1 data and L2 caches are protected by single error correction double error detection (SECDED) error correction codes (ECC) [1]. The RPU is comprised of a a dual-core ARM Cortex-R5 that can run in individual (parallel) or lock-step modes. Lock-step mode causes one core of the RPU to become a duplicate of the other core and they run synchronized. If at any point the two cores differ, the lock-step will trigger an error on the processing system. Each core of the RPU has its own 32KB instruction and data caches along with 128KB of tightly-coupled memory (TCM) [1]. The L1 caches, OCM, and the TCM are all protected by SECDED ECC. Lock-step mode and ECC suggest the RPU would be very reliable in the presence of SEUs. The GPU is comprised of a ARM Mali-400 MP2 but was not used in the experiments described in this paper.

In addition to these processors, the device contains a Configuration and Security Unit (CSU) shown in Fig. 2. The CSU is used to configure the PL and manage system security. The CSU is used in this work to program the PL through the processor configuration access port (PCAP). Access to the system's PCAP is only available through the CSU and a dedicated CSU direct memory access (DMA) controller. The CSU also contains a dedicated triple-redundant processor, a read-only memory (ROM), and a small private RAM for security sensitive data [1].

The ZU9EG contains a large amount of programmable logic resources. This includes 548,160 flip-flops, 274,080 look-up tables (LUTs), 2,520 digital signal processing slices (DSPs), and 912 user accessible block RAM (BRAM) modules that are each 36Kb. The complete device contains a total of 212,086,240 configuration RAM (CRAM) bits [1]. The PL also provides 24 GTH transceivers and 328 input/output (I/O) pins. The PL was built using 16nm FinFET design procedures. This technology node provides higher CRAM density, faster switching speeds, and less leakage current. These improvements result in higher performance and lower power consumption when compared to the planar technologies [2], [3]. The smaller 16nm FinFET technology node is expected to have a much smaller neutron radiation cross section than the previous planar FPGA CRAM.

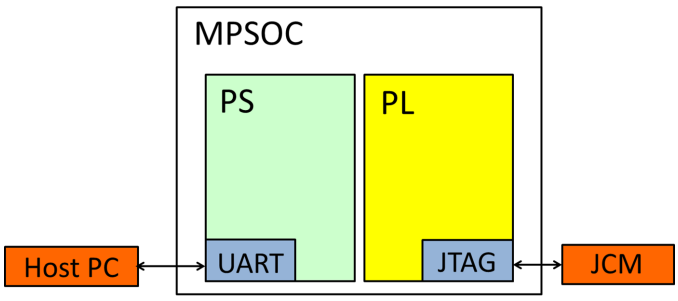

Fig. 3: Block Diagram of PS/PL separation.

III. Experiment Overview

Understanding the radiation effects within all of the components of a complex device like the MPSoC is very difficult. Unlike programmable processors and FPGAs tested in the past, the MPSoC contains many processors, peripherals, subsystems, and a large amount of programmable logic all on a single device. It is very difficult to organize a radiation test that captures the effects of radiation on all individual components of this device. Further, individual tests must be designed for a single component and should isolate this component from other components in the system as much as possible [4]. In addition, there are no generally accepted methods for performing radiation tests on such complex systems. The primary challenge of this work is to prepare radiation test methods for collecting radiation effects data on as many components on the MPSoC as possible.

The primary goal of these tests was to collect single-event effect radiation data on several of the key components of this complex device. In particular, these tests collected data for portions of the programmable logic (PL) and the processing system (PS) at the same time. One of the challenges of these tests was instrumenting the device so that data from both of these device regions could be collected at the same time as well as isolating these components from each other during the test.

A simple block diagram of the test organization is shown in Fig. 3. Data was collected from the PL using several different scrubber methods which will be described in detail in Section V. Particularly, the Zynq-based JTAG configuration manager (JCM) board, developed at the Configurable Computing Laboratory of Brigham Young University (BYU) [5], was used for collecting much of the SEU data through JTAG. The JCM reads configuration memory and detects SEUs independent of the PS.

To collect data simultaneously from the PS, the processor executed binaries containing a variety of benchmark programs that were stored on a SD card. The software binary was loaded on power-up by a first-stage bootloader (FSBL). A watchdog timer was enabled to detect processor hang events and cause the processor to reboot and reload the software. All software benchmarks generated output that was logged through

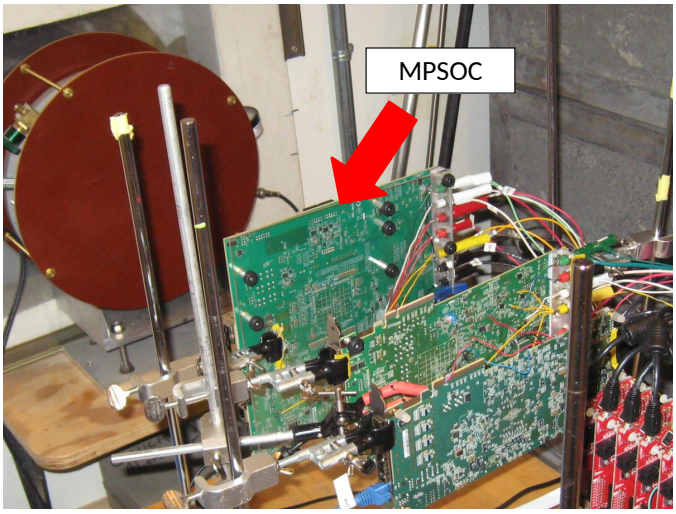

Fig. 4: MPSoC setup in neutron beam at LANSCE.

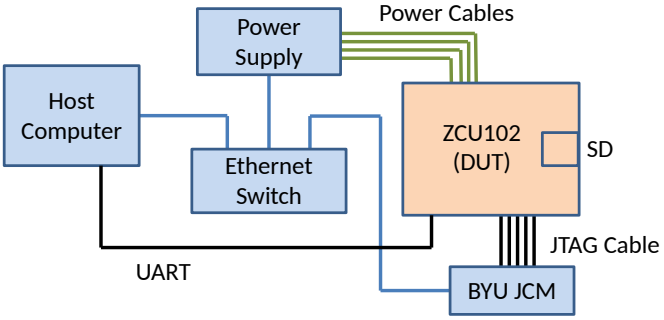

Fig. 5: Block diagram of MPSoC neutron test setup.

UART by a remote desktop computer. Crashes and silent data corruptions were identified by specific output messages or the lack thereof.

A series of three neutron tests were performed on several ZCU102 evaluation boards. During the first test in December of 2016, an engineering sample 1 (ES1) version of the device was tested with no modifications to the board. Due to an unanticipated failure in a power regulator on this evaluation board, no useable data was collected. Subsequent tests were performed on the ZCU102 board with a ES2 version of the device in August 2017 and November 2017.

IV. Experimental Setup

The ZU9EG MPSoC device was tested at the Los Alamos Neutron Science Center (LANSCE) neutron beam facility. The ZCU102 evaluation board from Xilinx was placed in the neutron flight path using stands and clamps as shown in Fig. 4. A block diagram of the test setup is shown in Fig. 5. A host computer was used to interact with the Zynq-based BYU JCM, the external power supply, and the MPSoC through UART. The host computer also logged the output from each component of the experiment. An Ethernet switch was used to connect the host computer to an external power supply and the BYU JCM through Ethernet. The BYU JCM was connected to the MPSoC by a JTAG cable that was used for external JTAG scrubbing.

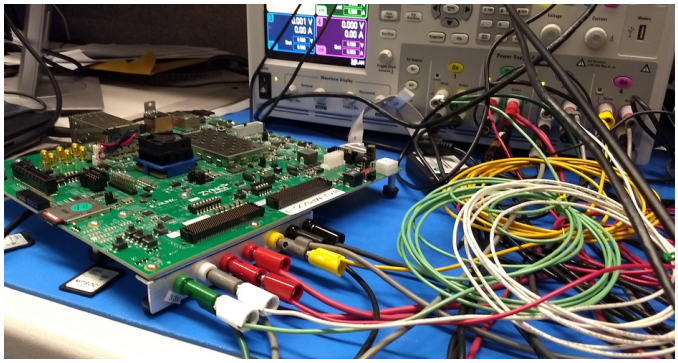

Fig. 6: Power supply and MPSoC setup.

A. Power Monitoring

To address the regulator failure in the first test, some modifications to the ZCU102 board were made to protect the board and device from unanticipated power failures. Specifically, the on-board power regulators were bypassed and the board was powered by an external Keysight N6705B power supply that carefully monitored system voltage and current as seen in Fig. 6. This setup was used during the August and November of 2017 LANSCE tests.

The power supply and Python scripts, running on the host computer, were used to monitor system current. These scripts would poll the power supply for the real-time current and voltages of each of the power rails. If the voltage or current was above a predetermined level, the script would send a command to the power supply to re-power the power rail. The high-current monitoring was performed in parallel with the SEU experiments but will be described separately in section VII.

B. LANSCE Neutron Beam

The LANSCE neutron beam was used for testing the MPSoC with high-energy neutrons. LANSCE provides several neutron beam flight paths ranging from 0.1 MeV to more than 600 MeV. The beam is generated by an 800 MeV proton linearly accelerated beam striking an unmoderated tungsten spallation source [6].

All experiments were performed on the 30° flight paths which have a neutron spectrum similar to the neutron spectrum in the atmosphere caused by cosmic rays. The neutron spectrum is shown in Fig. 7 and matches the JEDEC standard for spallation neutron beams [8]. Although the results from neutron testing do not assist in qualifying this device for use in space, such testing is very useful in identifying failure modes and unexpected device behavior that would be seen in other radiation environments such as with heavy ion.

V. Programmable Logic (PL) SEU Tests

The objective of these tests was to understand the impact of neutron radiation on the programmable logic (PL). The primary goal of the PL test was to estimate the neutron cross section of the configuration RAM (CRAM) as well as the internal user accessible block memory (BRAM). The cross section was estimated by counting the number of SEUs through scrubbing methods.

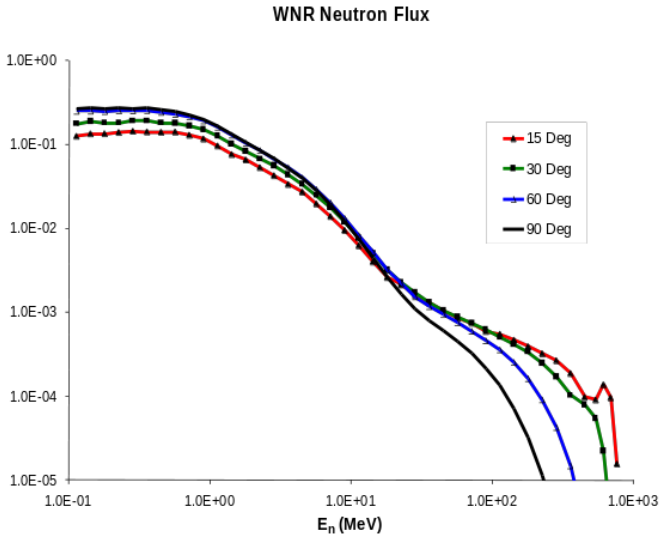

Fig. 7: Calculated neutron spectra for flight paths at LANSCE from [7]. The ”30 Deg” flight path was used in these tests.

A. PL Scrubbing Methods

The procedure for measuring the CRAM cross section is to perform continuous configuration “readback” and “scrubbing” on the user accessible CRAM memory. Scrubbing is a method of correcting single bit upsets in an FPGA by performing a readback of the configuration memory and comparing the readback data against a golden readback file. Differences between the readback and the golden file are counted and then corrected by writing the correct bits into the configuration memory frame thereby “repairing” the contents of CRAM [9], [10].

Three different CRAM scrubbing mechanisms were used during this experiment to count CRAM upsets and each of these methods were used to obtain an estimate CRAM memory neutron cross section. The three different scrubbing mechanisms used in this study include external JTAG scrubbing [5], internal scrubbing with the Xilinx “SEM” IP [11], [12], and scrubbing utilizing the processor configuration access port (PCAP) [13]. All three scrubber tests were ran independently in separate experiments as it would cause conflict on the FPGA memory otherwise.

The first scrubbing method utilizes the JTAG configuration port to read configuration memory, store a golden file of the FPGA configuration memory, identify upsets, and scrub the configuration memory. A custom external device called the BYU JTAG configuration manager (JCM) was used to provide high-speed JTAG configuration access, store the golden file, and perform upset identification and scrubbing. The JCM was placed outside of the beam flight path to prevent neutron interaction with the external scrubber.

The second scrubber utilizes the Xilinx Soft-Error Mitigation IP core or SEM IP. The SEM IP is placed into the FPGA fabric and performs SEU detection and correction through the system’s internal configuration access port (ICAP). The SEM IP also supports fault injection (“simulate” an SEU), which was used in testing the SEM IP configuration, and UART communication, which was logged by the external remote host computer. Since the SEM IP is placed into the PL fabric, the SEM IP is susceptible to SEUs which could change the memory used by the SEM IP or change the internal scrubber itself.

	Number of Errors	**Fluence (n/cm^2)**	**Cross Section (cm^2/bit)**	95% **Confidence**
JCM	16070	3.00×10^{11}	2.52×10^{-16}	3.98×10^{-18}
SEM IP	1648	3.16×10^{10}	2.46×10^{-16}	1.21×10^{-17}
PCAP	2313	5.12×10^{10}	2.13×10^{-16}	8.85×10^{-18}
BRAM	17554	1.74×10^{11}	3.02×10^{-15}	3.02×10^{-17}

Table I: Per-bit cross section of MPSoC FPGA configuration memory and BRAM.

The final scrubbing method used the PCAP, which is accessible by the various processors within the MPSOC. Through custom software, these processors can access the PCAP through the Configuration and Security Unit (CSU). The CSU DMA and related components are configured to transfer configuration data to and from the PCAP which allows for programming and reading of the FPGA CRAM data. The scrubber ran from a baremetal program on the RPU and the configuration memory readback was stored in the DRAM. The reliability of the PCAP scrubbing is based on the reliability of the MPSOC processors and I/O sub-system.

Similar to scrubbing of the FPGA CRAM, the BRAMs were tested using similar principles. The BRAMs were tested by creating an FPGA design that filled all available BRAMs and reading the contents of the BRAM with the BYU JCM. This data is compared against a golden file to identify and count BRAM upsets. The golden file is updated with the SEU induced changes to the BRAMs so as not to count BRAM upsets more than once. This change to the golden file is necessary because BRAMs cannot be “repaired” by scrubbing.

B. PL SEU Results

The SEU neutron cross section estimates of the FPGA configuration memory collected by the BYU JCM, the Xilinx SEM IP, and the PCAP scrubber are shown in Table I along with the results obtained for the the BRAMs. The cross section estimates from each scrubber differ slightly because of variations with the scrubbers and possibly missing some SEU counts due to the nature of the tests and scrubbers. A 2σ error 95% confidence interval is also shown in the table based on [4].

The neutron cross section estimate using the JCM scrubber is compared to other Xilinx device families in Fig. 8. The FPGA cross section of the MPSoC is 27.7x smaller than the 28nm devices and 10.1x smaller than the UltraScale devices [14].

The SEM IP did experience several failures due to SEUs. The SEM IP would get stuck in its “idle mode” and would fail to return to its “observation mode” (scrubbing mode). This is most likely due to parts of the SEM IP being corrupted by SEUs or corruption in the SEM IP’s BRAMs. This could be remedied by a full reconfigure of the PL; however, the experiment showed that a simple reconfiguration could result

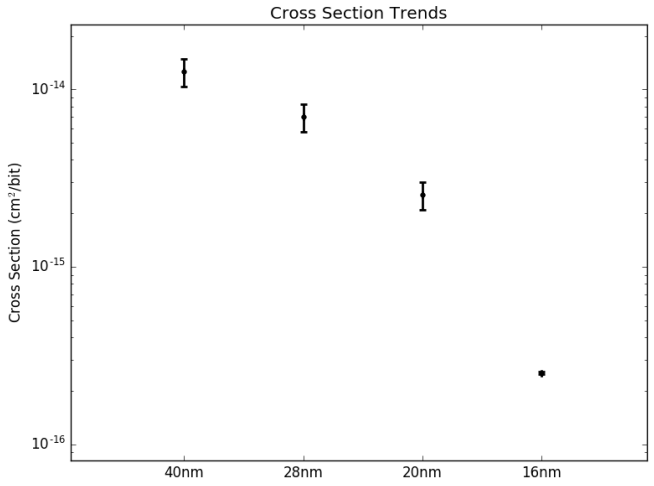

Fig. 8: Per-bit cross section comparison of Xilinx FPGA configuration memory based on [14].

	Number of Errors	Fluence (n/cm^2)	Cross Section (cm^2)	95% Confidence
SEM IP	3	3.16×10^{10}	9.50×10^{-11}	1.84×10^{-10}

Table II: Cross section of SEM IP.

in another SEM IP failure soon after (only counted as a single failure). A complete reboot and reconfiguration was the most reliable mechanism for repairing the failure. An estimate neutron radiation cross section for the SEM IP is shown in Table II.

VI. Processing System (PS) SEU Tests

The goal of these tests was to study the effects of neutrons on the execution behavior of a processor and its associated memories. These tests were performed on the MPSoC's ARM Cortex-A53 (APU) and ARM Cortex-R5 (RPU). There were two goals of the PS tests: first, to measure the cross section of the entire processing system executing a variety of software benchmarks, and second, to estimate the cross section of the memory cells used within the MPSoC including the OCM and the processor caches. These two goals will be described separately below.

A. Software Benchmark SEU Measurements

A variety of benchmarks were used to measure SEU sensitivity in the form of silent data corruption (SDC) and software crashes. Several baremetal benchmarks were used to test processor functionality. In addition, the Linux operating system was installed and tested while running an arithmetic benchmark. In all benchmark cases, the goal of the test was to measure the neutron fluence between each processor failure. These measurements were averaged and then used to create a neutron cross section estimate.

A number of baremetal benchmarks used in these tests were based on the benchmarks outlined in [15]. These benchmarks include the advanced encryption standard (AES), cache tests, and matrix multiply (MxM). The objective of using these benchmarks was to see if a neutron SEU could affect the processor in a way that would cause an output error in the program (SDC) or cause the software to crash. Each benchmark contains a self-checking system where a correct "golden" is created and then each iteration of the benchmark is compared against the golden. If the golden and the current iteration's results differ, specific output messages are sent and logged through UART. SDCs are determined by parsing the output of the benchmarks for these specific error messages. A watchdog timer was running to identify any processor hang/crash events. The number of SDCs and crashes will be used to calculate the neutron SEU cross section of an application running on the MPSoC.

	Number of Errors	Fluence (n/cm^2)	Cross Section[a] (cm^2)	95% Confidence
AES	0	1.31×10^{10}	$< 7.66 \times 10^{-11}$	2.82×10^{-10}
MxM	0	3.70×10^{10}	$< 2.70 \times 10^{-11}$	1.00×10^{-10}
Lnx/Dhr	0	6.33×10^{10}	$< 3.95 \times 10^{-12}$	5.85×10^{-11}

Table III: Software executable cross section operating on the Cortex-A53 APU.

[a]Though no software errors were observed, a value of one was used in the calculation of these results. This is used to show a worst-case cross section.

The different benchmarks were intended to fill different parts of the caches by using different sets of operations, which could make one of the benchmarks more susceptible to SEUs on this particular platform. Each of the tests were performed with the error correction code (ECC) enabled on the OCM and on the caches.

The Linux operating system was used as a comprehensive benchmark that involves much of the processor architecture. We anticipated that Linux running a software benchmark would have a larger cross section than that of our baremetal tests. To increase the likelihood of observing software errors, Linux was ran quite extensively in the processor portion of the tests. The Linux test was setup to run the operating system on all four cores of the Cortex-A53 processor using symmetric multiprocessing (SMP). Each of the four processor cores ran a separate instance of the Dhyrstone benchmark.

Surprisingly, *no* SDCs or crashes were seen in any of the software tests. With over 2.51×10^{11} n/cm^2 neutron fluence, we were unable to positively identify a processor SDC or a processor hang. The cross section for the software was calculated by assuming a single error even though no errors were observed [4] (see Table III). We believe that the cross section of these processor benchmarks is much lower than that reported in Table III but much more neutron fluence is needed to obtain a more accurate estimate.

The very low processor cross section is likely due to the presence of SECDED ECC on the L1 data and L2 memory caches and interleaving of cache memory words. These features help mitigate SEUs from propagating from the cache memories to the output of the benchmark as SDCs. When an ECC error is detected, the bit is repaired in the cache and the correct bit is transmitted to the processor for use.

	Number of Errors	Fluence (n/cm^2)	Cross Section (cm^2/bit)	95% Confidence
OCM	168	5.49×10^{10}	1.46×10^{-15}	2.25×10^{-16}
Caches	1302	8.28×10^{10}	1.50×10^{-15}	8.31×10^{-17}

Table IV: Per-bit cross section data from the processor.

B. PS Components SEU Measurements

Our second objective was to measure the cross section of individual memory cells within the on-chip memory (OCM) and caches. Even though we were not able to observe any processor operational failures, we were able to observe upsets within these memories by capturing exceptions and reading processor performance registers.

To test the OCM memory and estimate a cross section, a baremetal program was created that initialized the 256KB OCM to a known set of values and enabled the ECC and system interrupts. The program then repeatedly read all OCM words and checked the OCM contents. If an error was detected by the OCM ECC, an interrupt would be fired and the software would capture and count the interrupt.

To measure upsets within the cache memories, baremetal programs were written to exercise the ARM A53 memory system by reading from all data cache locations. As memory is read from the cache, the internal cache ECC records cache memory upsets. The baremetal program repeatedly reads the internal error syndrome co-processor registers on the A53 to count ECC errors. These registers record memory errors that occur within the caches, what upset, where in memory, and how many times that same location has been upset. These registers were used to detect cache errors in the L1 instruction and data caches, as well as the L2 shared cache, in the baremetal and Linux applications. The programs that exercised the caches include a baremetal matrix multiply test, with the error syndrome registers, and Linux running individual Dhrystone benchmarks on each core, with the error detection and correction (EDAC) module. The neutron SEU cross section of the OCM and caches are shown in Table IV.

VII. High Current Events

In addition to measuring the single-event effect behavior of the MPSoC, our experiments monitored the presence of radiation-induced latch-up or high current events. The objective of this study was to monitor the power events on the board and determine the extent of the issue presented by the events. This study emerged after the power anomaly that occurred during the first test.

This test was performed by bypassing the on-board power regulators and monitoring the current and voltage of each output channel from an external power supply (Figs. 6 and 9). In this setup, the 3.3V, 1.8V, 1.2V, and 0.85V power lines were only powered. This leaves many board *peripherals* unpowered like the Ethernet and USB because they require a 5V power line.

The power supply was monitored by the host computer running a Python script. The Python script reads all of the supply currents and monitors the currents, comparing the values based on a pre-determined threshold current value. The threshold value was determined by monitoring the current during normal execution and adding 20%. If the current is above the pre-determined threshold, the script sends a reboot signal to the power supply, causing the MPSoC to power cycle. The full current draw of these events was not recorded as the purpose of these scripts is to protect the board from damage rather than fully characterizing the high current event.

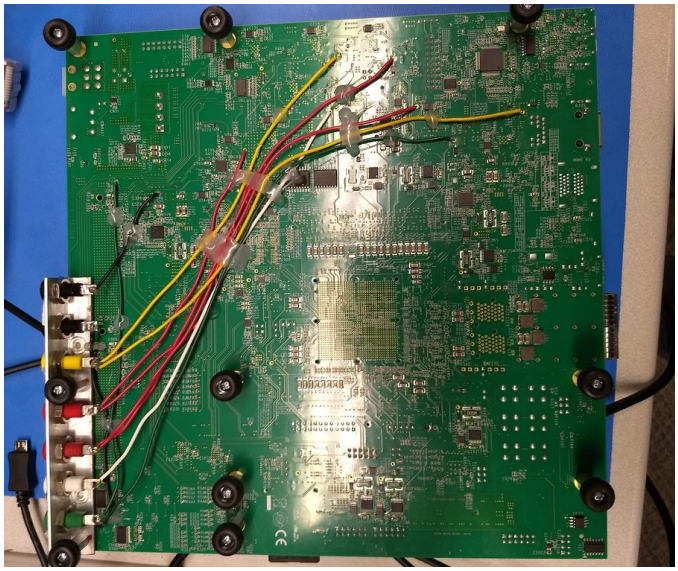

Fig. 9: Power regulators bypassed through board test points.

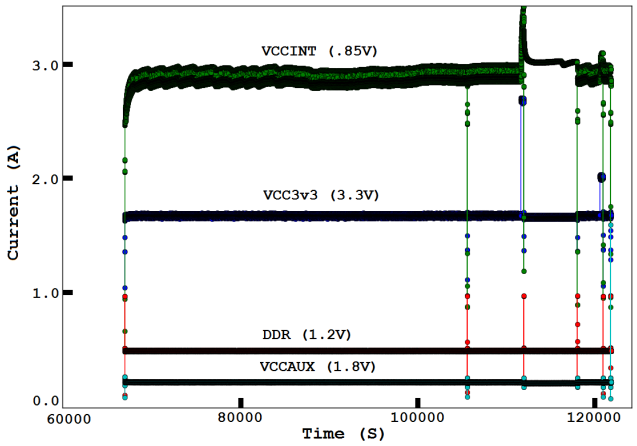

Fig. 10: Current plot of one segment from the second test. Notice the large spike in current near the end of the segment.

Numerous current events were observed and analyzed through the monitored currents of the external power supply. These results showed various different power events occurring on the board including voltage spikes and current spikes (Fig. 10 and Fig. 11).

A neutron cross section for the power events was estimated in Table V. The only power lines that exhibited high current events in our tests were the VCCAUX 1.8V line (the auxiliary power rail for the PL) and the VCCINT 0.85V line (the internal power rail for the PL). The VCCAUX power line appears to be the most susceptible line as it was the problem power line

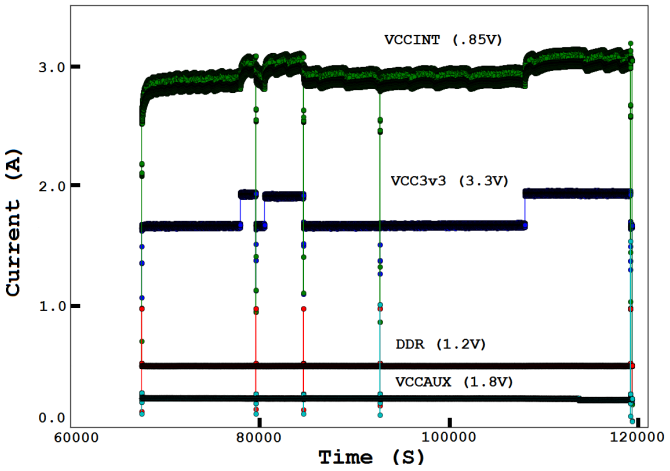

Fig. 11: Current plot of one segment from the second test. Notice the increased current during several periods.

	Number of Errors	Fluence (n/cm^2)	Cross Section (cm^2/line)	95% Confidence
VCCAUX	25	3.09×10^{11}	8.09×10^{-11}	3.82×10^{-11}
VCCINT	1	3.09×10^{11}	3.24×10^{-12}	1.49×10^{-11}
Total	26	3.09×10^{11}	8.42×10^{-11}	3.88×10^{-11}

Table V: Power Event Cross Section Results.

for the majority of the reported current events.

On average, a current event caused an increase of 330 mA on the particularly affected line, while some events are more extreme, such as the one that was shown in Fig. 10. However, these tests were performed with a current limitation to protect the evaluation board from any potential damage.

During the third test, a second MPSoC board was tested with the objective of understanding the nature of the events, on-board mitigation technique effectiveness, and wear of the chip due to these current events. This second MPSoC had no power modifications done to the evaluation board and was connected to a large uninterruptible power supply (UPS) so as to guarantee that the current event would not be lost due to a loss in power.

A current event was "captured" on the MPSoC by carefully monitoring the voltages and currents of the evaluation board through a Maxim Integrated PowerTool™. Once a high current event was detected, attempts were made to mitigate the situation without re-powering the board. These attempts included resetting the processor, performing a power-on reset (PoR), and triggering resets through external JTAG commands. A current event appears only to be resolved through a complete power cycle of the board.

The "captured" event showed that communication to the MPSoC board can be lost due to a current event. Communication through UART and JTAG were lost on several (but not all) of the current events. Additionally, the event conditions appear to persist until the board has been powered down. This MPSoC board currently still has a "captured" current event and will be used in a follow-up study of the long-term effects of the current events. Other ideas to recover the board without power down, like reducing voltages of certain power rails, will also be tested in the future.

VIII. Conclusion

Simultaneous tests involving neutron SEU FPGA tests and SEU processor tests have been performed on the Xilinx ZU9EG MPSoC board. The first part of the tests was performed focusing on the FPGA resources of the MPSoC and showed the resulting neutron SEU cross section as determined by the BYU JCM, Xilinx's SEM IP, and a baremetal software PCAP scrubber. The second part of the tests was focused on the processor portion of the chip. The neutron SEU results showed that the processor did not experience any SDCs or system crashes during the extent of the tests, but that the chip was being upset through the SEUs observed on the caches. This series of tests has shown the neutron SEU cross section of the various test components using several different methods.

In parallel with the PL and PS tests, high-current monitoring tests were performed using an external power supply and power monitoring scripts. Numerous high-current events were observed on the MPSoC and logged through continuous current and voltage monitoring and recording. Various events including voltage spikes and current spikes were observed on the MPSoC during radiation testing. The 1.8V VCCAUX appears to be the most susceptible power rail to power events induced by neutron radiation and an estimate neutron cross section obtained.

References

[1] *Zynq UltraScale+ Device - Technical Reference Manual*, Xilinx.
[2] L. Hansen, *Unleash the Unparalleled Power and Flexibility of Zynq UltraScale+ MPSoCs*, Xilinx.
[3] S. Patil and V. S. K. Bhaaskaran, "Optimization of power and energy in FinFET based SRAM cell using adiabatic logic," in *2017 International Conference on Nextgen Electron Technologies: Silicon to Software (ICNETS2)*. Chennai, India: IEEE, March 2017.
[4] H. Quinn, "Challenges in testing complex systems," *IEEE Transactions on Nuclear Science*, vol. 61, no. 2, pp. 766–786, Apr 2014.
[5] A. Gruwell, P. Zabriskie, and M. Wirthlin, "High-speed programmable FPGA configuration through JTAG," in *2016 26th International Conference on Field Programmable Logic and Applications (FPL)*. Lausanne, Switzerland: IEEE, Sept 2016.
[6] LANSCE. Weapons neutron research facility at lansce. [Online]. Available: http://www.lansce.lanl.gov/facilities/wnr/index.php
[7] LANSCE. Weapons neutron research flight paths. [Online]. Available: http://www.lansce.lanl.gov/facilities/wnr/flight-paths/index.php
[8] JEDEC, *Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices*, JEDEC Standard.
[9] I. Herrera-Alzu and M. Lopez-Vallejo, "Design techniques for Xilinx Virtex FPGA configuration memory scrubbers," *IEEE Transactions on Nuclear Science*, vol. 60, no. 1, pp. 376–385, Feb. 2013.
[10] A. Stoddard *et al.*, "A hybrid approach to FPGA configuration scrubbing," *IEEE Transactions on Nuclear Science*, vol. 64, no. 1, pp. 497–503, Jan. 2017.
[11] *Soft Error Mitigation Controller v4.1*, Xilinx.
[12] T. Bates and C. Brideges, "Single event mitigation for Xilinx 7-series FPGAs," in *2018 IEEE Aerospace Conference*. Big Sky, MT, USA: IEEE, March 2018.
[13] A. Stoddard, A. Gruwell, P. Zabriskie, and M. Wirthlin, "High-speed PCAP configuration scrubbing on Zynq-7000 All Programmable SoCs," in *2016 26th International Conference on Field Programmable Logic and Applications (FPL)*. Lausanne, Switzerland: IEEE, September 2016.
[14] *Device Reliability Report - Second Half 2017*, Xilinx.
[15] H. Quinn *et al.*, "Using benchmarks for radiation testing of microprocessors and FPGAs," *IEEE Transactions on Nuclear Science*, vol. 62, no. 6, pp. 2547–2554, Dec 2015.