Challenges in Assessing Single Event Upset Impact on Processor Systems

Wojciech A. Koszek, *Member, IEEE,* Austin Lesea, Member, IEEE, Glenn Steiner, Member, IEEE
Dagan White, Member, IEEE and Pierre Maillard, Member, IEEE
Xilinx Inc.

***Abstract*—This paper presents a test methodology developed at Xilinx for real-time soft-error rate testing as well as the software framework in which Device-Under-Test (DUT) and controlling computer are both synchronized with the proton beam controls and run experiments automatically in a predictable manner. The method presented has been successfully used for Zynq®-7000 All Programmable SoC testing at the UC Davis Crocker Nuclear Lab. Presented are the issues and challenges encountered during design and implementation of the framework, as well as lessons learned from the in-house experiments and bootstrapping tests performed with Thorium Foil. The method presented has helped Xilinx to deliver high-quality experimental data and to optimize time spent in the testing facility.**

***Keywords*—Error detection, soft error, architectural vulnerability, statistical error, confidence level, beam facility control**

I. Introduction

To assess the impact of Single Event Upset on silicon systems, empirical data are necessary. The data are typically gathered from a DUT equipped with instrumentation and exposed to a radiation source such as Thorium foil or a high-energy particle beam. Most efforts are targeted towards so-called one-shot testing, where each exposure is followed by a manual read-back from the DUT and data validity check [1]. Regardless of the facility or source of radiation used, following is an algorithm representing a typical testing scenario:

SEU-TESTING:
1) Understand general setup and device behavior
2) Tuning the source of radiation to the setup
3) Conducting a static or dynamic test of the DUT
4) Analyzing experimental data
5) Repeat from (3) until enough samples are obtained

Steps 1 and 2 are important, since they help to understand the scope of testing effort and help with planning, but they are performed only once per visit to the facility, optimizing them is not critical and may not bring significant benefits. We believe that for the general silicon testing this technique may be appropriate, however we argue that a more methodic approach for System-on-Chip (SoC) testing is necessary to deliver reliable data with less error margin and higher confidence. This paper presents results [2] of a 4-month project to characterize the Zynq-7000 SoC.

The rest of the paper is organized as follows: In Part II, we present initial goals and technical assumptions made before infrastructure the effort start. Part III discusses technical details behind the setup used and testing facility. Part IV explains the choice of test code and the process of bootstrapping our test-bench environment. Part V explains our automation testing framework used. Park VI presents summary and discussion on our results.

II: Goals and assumptions

The Xilinx Zynq-7000 SoC is a 28nm SoC device [3] where FPGA Programmable Fabric is bridged through a high-speed AXI interface [4] to a dual-core ARM Cortex-A9 [5] Processing System. Within the Processing System are multiple hard IP blocks surrounding the CPU. A high level block diagram of the Zynq-7000 SoC structure is depicted on the following diagram:

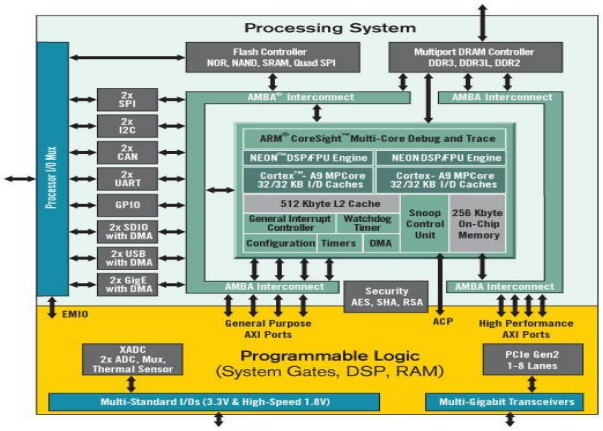

Figure 1: Diagram showing a structure of Zynq SoC

Xilinx has extensive prior experience with SEU characterization in Field-Programmable Gate Array (FPGA) devices [6] [7] , and its results are publicly available [8], however SoC testing with an ARM Processing System has never been performed. For the purposes of testing, significant planning effort has been accomplished. Zynq SEU characterization was a result of work of R&D, A&D, Marketing and System Software groups, and the requirements were set after many weeks of discussions. The list of things which were to be tested includes, but is not limited to:

- Testing of L1, L2 cache and On-Chip-Memory (OCM)

- Testing of both ARM Cortex-A9 cores together with NEON SIMD ISA
- Testing of DDR controller and AXI interconnect
- Testing of of other hardened blocks suc as Snoop Control Unit (SCU)

To reach the expected confidence levels and to provide comprehensive data on each of the blocks, the general idea was that thousands of data samples were necessary. It is important to mention that the large part of the design was driven by the need to minimize costs of the testing effort. The following table presents approximated hourly rates of the known testing facilities:

Facility	USD / Hr
Thorium foil	0
Proton	700-950
Neutron	1000-1500

Table 1: Average hourly rate of known test facilities

For example, with a price as high as 850USD/Hr, the 35 second boot time of the operating system costs:

$$\frac{850USD}{3600s} \times 35s \approx 8.26USD$$

With a plan to perform several hundred experiments, it was very important to remove unnecessary wasted time on board startup. To be able to achieve such effort, members of our team had both hardware and software background. Moreover some members were not exposed to Single-Event-Upset (SEU) testing before. Xilinx performs many visits to beam testing facilities per year, and one of such visits has been used to educate the team and to predict possible challenges of the environment.

III: TESTING FACILITY AND THE SETUP

The following figure 2 shows the layout of the control facility with a typical setup:

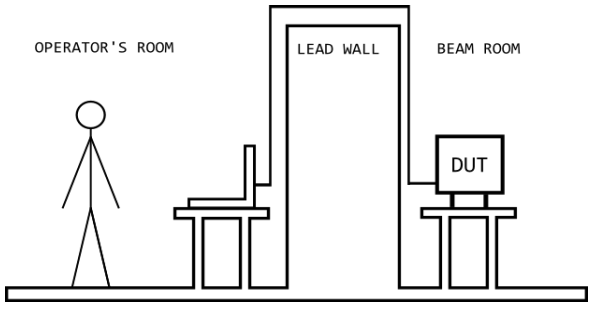

Figure 2: Typical control facility and DUT placement

Some of difficulties encountered include:

- Control room separated from beam room
  - Limited visibility of the setup, only though the video camera
- Connectivity with the DUT is limited
  - Typically RS232 (serial port) with cable expanders. Impossible with modern boards due to USB cables. Solved by placing a PC in the beam room
- Highly-reliable board/setup is required, since stopping the experimentation to adjust the DUT is very expensive
- Heterogeneous environment within the lab
- Many computers running different software with different operating systems

The beam facility provides only rudimentary equipment, thus for purposes of extensive testing we devised a checklist of 23 items which were brought to the testing facility. The most important items were:

- Backup board (in case of setup problems)
- Keyboard-Video-Mouse switch (KVM) to be able to provide communication with the control PC
- Docking stations with additional USB-Ethernet adapters to let operators have access to the lab network and the Internet

During the Thorium Foil bootstrapping we've had a physical access to the setup in Xilinx's reliability lab. Basic functionality at the beginning of the test development was visually inspected. This approach was helpful in further efforts to debug problems, and is suggested during test development. Once basic functionality of the setup was ready, we made it available remotely through Secure Shell (SSH) connection. From this stage we tried to prototype the setup for the target environment at the testing facility.

Our setup consisted of the Xilinx Zynq ZC702 board with zc7020 soldered Ball Grid Array (BGA) XC7020 part.

Figure 3: ZC702 board and ZC7020 Zynq chip

The board supports booting from Secure Digital (SD) card and Quad-Serial Peripheral Interface (QSPI) flash card. Initialy JTAG [9] was considered as a possible configuration technique, because it provided the best instrumentation and debugging capability. However, its speed and flexibility was unacceptable for our needs. Additionally, it would add a lot of complexity to our setup by forcing us to depend on a large amount of software running on the PC controlling the experiment. Measurements showed that QSPI provided the fastest (2s) initialization time, but it had limited capacity. Our OS (mentioned later) comes in the form of a 1MB Executable and Linkable Format (ELF) file. To target testing toward the functionality we were interested in, our medium had to hold

many ELF files, each with a slightly different configuration. To solve the capacity problem, a hybrid approach has been used for board startup: 250kB U-Boot loader [10] starts from QSPI and loads our custom 1MB ELF files from the Ethernet network through the Trivial File Transfer Protocol (TFTP) protocol from the operator's laptop. Thanks to this approach, during one emergency case in which we realized one of Zynq's interrupt lines was wrongly configured, we were able to quickly apply a fix to the source code, recompile our ELF files and to deploy it to the experiment instantly.

IV: TESTING CODE AND BOOTSTRAPPING

The large number of hardened IP blocks presented a problem for us, since providing SEU instrumentation for all of these blocks would require significant software development effort. Xilinx SDK software potentially gave us a good start to research the path where testing software templates were generated from the tool. However this solution has been quickly postponed, since instrumentation necessary to provide required feedback during testing was not present and would require significant changes and big engineering effort, which would be considered as a cost and which was not a real focus of this project. For example: during initial Thorium Foil testing we've understood that the chip's memories are the most likely to encounter upsets, which triggered large amount of Data and Pre-fetch Aborts returned from the software running on the Central Processing Unit (CPU). In the typical embedded application running without Memory Management Unit (MMU) enabled, Data and Pre-fetch aborts are rare, thus no standardized library templates include capability of handling them easily from the embedded application.

We only created reference static tests for memories from the tool. The way the On-Chip Memory test program works is as follows:

OCM-TEST:
- Load a known pattern to the memory
- Wait in the inactive state
- Read back and compare the memory

Interestingly, for testing L1 and L2 caches, the solution wasn't as straight-forward. In the process of test code development we've learned L1 and L2 memories don't provide a mechanism for predictable population of the L1/L2 caches. A method devised by Xilinx for cache memory testing helped us deliver data which match our theoretical predicted values, and is currently being patented.

After cache memory testing, further testing is normally conducted by running simplistic tests crafted to concentrate on SEU upsets in Arithmetic Logic Unit (ALU), Floating Point Unit (FPU) and memory units. However with Thorium Foil, we haven't observed any upsets that way since the ALU/FPU area is very small in Zynq's CPUs.

Xilinx's approach to SEU characterization was instead to model the real-world customer scenario in terms of the functionality and the complexity of software running on the Processing System. Running Linux was explored, but given previous failures to run an industry-grade operating system under radiation [11] [12] this approached was postponed.

Xilinx already has had an industry-grade testing methodology called System-Level Test Operating System. SLTOS is a custom-made Operating System (OS) equipped with drivers to all peripherals within Zynq-7000, and comes with extensive instrumentation. Maturity of SLTOS spoke for itself, since for the last 10 years it has been successfully used to catch 100+ silicon issues in Xilinx devices during pre- and post-silicon testing and many more fixes applied to our products. Short list of SLTOS features:

- run tests on both ARM CPUs and NEON units in parallel
- trigger simultaneous Direct Memory Access (DMA) transfers from and to available memories
- run with interrupts and traps enabled and use them within normal testing
- possibility of unexpected interrupts/trap detection
- through internal Analog-to-Digital Converter (ADC) monitor the system for voltage and temperature changes
- provide text-file based configuration which lets to customize SLTOS configuration (devices enabled, instruction mix)

SLTOS upon detecting an upset prints a dump similar to UNIX ``*dmesg*'' command, from which exact cause of failure can be obtained. Based on this data we derive information such as MTTF and details on the failing IP block state.

Initial experiments with the SLTOS proved to be effective, with significant amount of expected issues to be caught. Thus, the decision was made to repurpose SLTOS for the SEU testing. Changes added for SEU testing include implementation of watchdog timers, a redundant Universal Asynchronous Receiver/Transmitter (UART) connection, and some additional monitoring of the board's state.

Because of cost factors, our decision was to use Thorium foil for most of the development and bootstrapping effort, since it could be performed in-house at Xilinx. For the Thorium foil, to enable the silicon die to receive the Alpha particles, the chip's molding compound protecting the die has to be removed. This process is known as de-capping and is achieved by applying nitric acid to the chip's cover. Depicted is the de-capped Zynq chip:

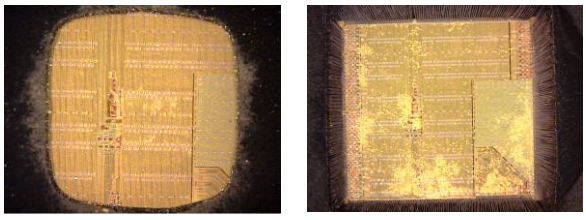

Figure 4: Incorrectly de-capped part with leftovers of molding compound (left), and correctly de-capped part (right)

Several attempts were required to get the device with the compound de-capped enough to expose the whole die to alpha particles. For units where this process failed [Picture 4], detailed testing showed unexpected characteristics, e.g.: for a unit where L1 cache was still protected, we haven't observed any memory upsets. Because of that, it is advised to run static

tests on basic memories first, so that after dynamic tests are started, one can be convinced that the results reflect real system's behavior.

Die covers act as a protecting surface of the compound, and us removing it resulted in devices becoming more fragile. We believe at least 2 devices were destroyed in the process of plugging the device into the open-top socket.

To eliminate this problem, we've devised a new way to use Thorium foil with de-capped units: we use standard BGA closed-lid sockets which Xilinx boards are designed for, and in which Thorium foil is trapped in a device and covered by a socket's metal lid. The advantage of this method is that device, once mounted in a socket, is protected from external conditions such as humidity and dust and has proven to work very reliably. We've used the same device for many weeks during the initial bootstrapping. Disadvantage of this method is that the expensive socket becomes contaminated with Thorium and must be properly disposed when it does.

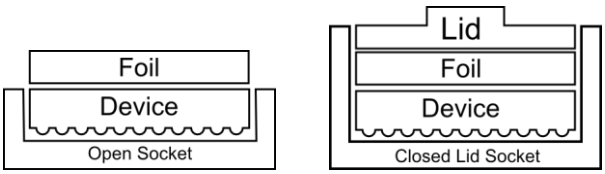

Figure 5: Thorium-foil placement, as used before (left) and after proposed change of the foil location (right)

Initial setup was invaluable for the development of the testing infrastructure. Choice to use the Xilinx developed ZC702 board was very good, since it let us to get assistance from Xilinx board's group. Also having an access to multiple boards of the same type was helpful, since in case of problems, it was important to understand whether it's a setup problem, or an issue related to the Alpha radiation. We ended up having a separate replicated setup with a normal, soldered part in a room conditions just for the purposes of bug reproducing.

SEU testing was not a supported use case for a customer board, and as a result of that several issues had to be addressed. For example: frequent power-cycling of the board caused problems with control PC's USB controller, since each time the board was power-cycled, the USB subsystem required the USB enumeration to happen to correctly attach the required drier to our USB-UART connection. This behavior was initially believed to be a result of radiation and the problem on a board's side, but turned out to be general systems problem.

Minor change to USB-controled relay to use System-Reset was made, to keep the USB UART powered during the reset procedure. Because of this approach, the system has become more similar to a real-world scenario where power is likely to be applied to the system permanently, and USB UART remained powered during the reset procedure, and didn't cause any additional problems.

Number of elements of the system was believed to need a redundant replacement. For example we've implemented a support for the 2nd UART in SLTOS, in case UART connection gets impacted. However it turned out to not be necessary. We have not observed problems with UART in our testing.

V: AUTOMATION

Automation has been present in our in-house experiments, but was based on the company-wide automation system. The automation system gives us an ability to power board on and off, reset the CPU, connect to UART and schedule jobs in isolated way on the boards and this is how test-driving with Thorium Foil has been done. Xilinx automation however makes an extensive use of Xilinx network resources, NFS filesystem and other complex tools which made direct porting of the system to the standalone PC environment very hard. Upon a success with the foil, we decided to develop our SEU automation from scratch so that we could run experiments in the proton beam facility in a similar way.

The testing facility doesn't provide any automation, other than the ability to synchronize through with the cyclotron. This mechanism is only available to older computers with PCMCIA interface and specialized card, which we didn't know about upfront. Thus we didn't make use of this specialized interface.

Our framework is entirely software based, and is distributed across 3 computers, which synchronize together and track the progress of each experiment made. Unlike standard testing, we have all computers participating in the experiment bridged in a common Ethernet network. Ethernet was much more convenient for the purposes of testing, since in case of failure, we were able to quickly fetch log files from the control PC and back it up on operator's laptops (in case control PC would crash) as well as to plot the results. For this to happen, it was necessary to plug a USB-Ethernet adapter to the beam control PC and reconfiguring it to be able to communicate with operator's laptops. Every computer, when connected to the Ethernet switch could communicate with each other.

Pictured is a diagram of elements in our infrastructure:

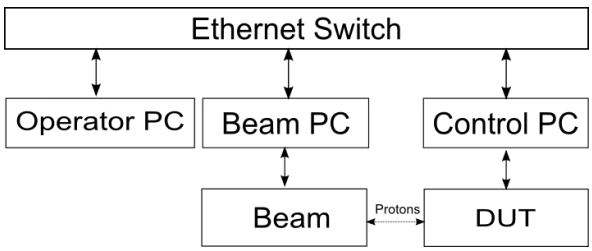

Figure 6: Elements of our infrastructure

The operator PC is a laptop running Microsoft Windows 7. The beam PC is connected directly to the cyclotron beam and came with Microsoft Windows XP. It has the BeamOn 6.30.10 program installed for communication and control of the beam. Both computers are located in the operator's room. The control PC was running the FreeBSD UNIX, which provides excellent stability [13]. The control PC is running a main management program, and algorithm executed is a loop:

A) Reset the system and wait for the boot-loader to start

B) Issue a command to download an ELF binary and wait for it to finish
C) Run the binary and wait for the board to initialize correctly
D) Start running tests
E) Start the beam
F) Wait till upset reported
G) Stop the beam
H) Obtain a dump
I) Goto A

Steps A-D prepare the setup and are performed without any radiation applied to the setup. To control the setup reset functionality, we wired USB-controlled relay to board's POR reset pins. We wrote a ``usb_power'' control program based the ``libusb library'' [14] and operated from within our framework.

Step E is the most important and is accomplished by our framework sending mouse/keyboard events through the network from the Control PC to the Beam PC. Automation of that kind was necessary, since we haven't had any other way to provide automation of BeamOn program because it is a GUI application. Sending mouse and keyboard events required calibration, but worked extremely robustly after initial setup.

Once the beam was started, framework awaits for potential upsets. Detection is performed by continuous monitoring of the UART output. Once the triggering warning or unexpected output is seen, framework stops the beam in step (H) and waits for information dump to be printed out. Dump is recorded on the Control PC disk through the ``cu'' terminal program, and then the whole algorithm restarts itself.

The most important function of the framework is monitoring for unexpected situations. The table presents approximated *maximal* time for the certain stages of the setup and SLTOS startup which are considered acceptable:

Name of the stage	**Timeout value [s]**
Boot-loader	2
OS fetch from network	5
SLTOS boot	35
Test execution	500
Dumping data	1000

Table 2: Maximal time for the certain stages of the DUT startup

Our framework implements a timed state machine and with each timeout, the framework stops the beam and restarts the experiment.

Automation of that kind provided Xilinx an opportunity to run several hundreds of experiments in the facility during two 2-day visits to the Crocker Nuclear Lab.

IV: SUMMARY

Presented is a description of the testing methodology developed by Xilinx for the purposes of SEU impact assessment in Xilinx Zynq SoC family of products. We state that the presented method for Thorium Foil testing can be valuable for efforts requiring significant investment in time, where reliable configuration is necessary. We believe general guideline for experiment operating in the testing facility can be valuable and help to prevent others from unexpected surprises while conducting experiments. We claim that developed framework has helped us perform at least 5 times more experiments to what could physically be possible without our automated approach.

IV: ACKNOWLEDGEMENTS

The authors would like to thank Nathalie Chan King Choy and Dan Isaacs for providing feedback and suggestions during the writing of this paper and Jeff Barton and Gary Swift for their help during the project.

V: REFERENCES

[1] Q. Heather, "61(2):766-786. DOI: 10.1109/TNS.2014.2302432," *IEEE Transactions on Nuclear Science*, no. April 2014, pp. 766 - 786, 2014.

[2] Austin Lesea, Wojciech Koszek, Glenn Stainer, Gary Swift, Dagan White, "Soft Error Study of ARM SoC at 28 Nanometers," *SELSE2014*, 2012.

[3] http://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf, Zynq-7000 All Programmable SoC Technical Reference Manual.

[4] http://bit.ly/13JVJat, AXI Reference Guide.

[5] http://www.arm.com/products/processors/cortex-a/cortex-a9.php, Cortex-A9 Processor.

[6] D. White, Considerations Surrounding Single Event Effects in FPGAs, ASICs, and Processors, http://bit.ly/1zdlXiw.

[7] G. Swift, "Dynamic testing of Xilinx Virtex-II field programmable gate array (FPGA) input/output blocks (IOBs)," *IEEE Transactions on Nuclear Science*,, no. Dec. 2004, pp. 3469 - 3474, 2004.

[8] I. Xilinx, "Device Reliability Report," http://bit.ly/16A481b.

[9] I. 1. W. Group, "Standard Test Access Port and Boundary-Scan Architecture," http://grouper.ieee.org/groups/1149/1/.

[10] D. S. Engineering, "Das U-Boot," www.denx.de/wiki/U-Boot.

[11] F. Irom, "Guideline for Ground Radiation Testing of Microprocessors in the Space Radiation Environment," Jet Propulsion Laboratory , 2008.

[12] A. B. D.M. Hiemstra, "Single event upset characterization of the Pentium(R) MMX and Pentium(R) II microprocessors using proton irradiation," 2000.

[13] http://www.freebsd.org, "The FreeBSD Project," 1993-2014.

[14] http://www.libusb.org/, "The LibUSB Library".