3-11-2011

Radiation Induced Fault Detection, Diagnosis, and Characterization of Field Programmable Gate Arrays

Thomas B. Getz

RADIATION INDUCED FAULT DETECTION, DIAGNOSIS, AND CHARACTERIZATION ON FPGAS

THESIS

Thomas B. Getz, Captain, USAF

AFIT/GE/ENG/11-12

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT/GE/ENG/11-12

RADIATION INDUCED FAULT DETECTION, DIAGNOSIS, AND CHARACTERIZATION ON FPGAS

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Thomas B. Getz, BS

Captain, USAF

March 2011

AFIT/GE/ENG/11-12

RADIATION INDUCED FAULT DETECTION, DIAGNOSIS, AND CHARACTERIZATION ON FPGAS

Thomas B. Getz, BS

Captain, USAF

Approved:

Dr. Yong C. Kim (Chairman)

14 MAR 2011
Date

Dr. James C. Petrosky (Member)

14 Mar 11
Date

Ronald A.

Dr. Ronald A. Coutu, Jr (Member)

14 Mar 11
Date

AFIT/GE/ENG/11-12

Abstract

The development of Field Programmable Gate Arrays (FPGAs) has been a great achievement in the world of micro-electronics. One of these devices can be programmed to replace the need for thousands of individual specialized devices. Despite their great versatility, FPGAs are still extremely vulnerable to radiation from cosmic waves in space. Extensive research has been conducted to examine how radiation disrupts FPGAs. This research incorporates and enhances current methods of radiation detection. The stuck-at fault model and delay model are used to represent common radiation induced issues, single event efects and total ionizing dose respectively.

An active sensor network design is created that has the ability to detect flipped bits and delay errors caused by radiation along with their location, amount and duration. All of this is accomplished and reported in real time. During this research, total ionizing dose errors are successfully modeled, detected, quantified and reported. The single event effect detection method is also a success, but is not validated in a radiation environment. More testing is required, but once that is done this system can be incorporated to enhance current FPGA reconfiguration methods that automatically place application logic away from failing sections of the FPGA. This system has great potential to become a valuable tool in fault mitigation.

Acknowledgments

I want to thank the members of my committee, for the time they dedicated toward this effort, giving me advice and assistance during my research. I was very fortunate to have great lab partners beside me as classmates and friends. They were always willing to help out when I ran into problems. I'd also like to thank my friends and family for their ongoing support.

Thomas B. Getz

Table of Contents

Page

List of Figures

Figure Page

List of Tables

RADIATION INDUCED FAULT DETECTION, DIAGNOSIS, AND CHARACTERIZATION ON FPGAS

I. Introduction

Recently, the introduction of Field Programmable Gate Arrays has allowed developers to replace several different specialized circuitry with one dynamic device. FPGAs implement reprogrammable logic to take the form of any number of devices an unlimited number of times. The versatility of FPGAs has made them a very desirable tool for multiple platforms [1]; one of which being space operations. While normal conditions pose little threat to a FPGA device, radiation present in the space environment is another story. Radiation can cause numerous different types of failures on these devices. These failures range from localized, temporary failures in which incorrect values traverse throughout a logic circuit to failures characterized by poor performance and speed degradation. In any case, these failures cause unwanted effects that can lead to major damage of other systems dependent upon the FPGA. Therefore, understanding the effects of radiation on FPGAs is becoming very important.

1.1 Motivation

The study of the behavior of electronics in a space environment is crucial for reliable operation. One technique of protecting circuits from radiation is adding a physical layer of protection to make it radiation hardened. While this method is effective, it is also very costly [2]. This is why recent efforts have included incorporating the adaptive nature and small feature size of the FPGA to design circuits that are more fault-resistant. FPGAs also offer the versatility to create a platform that can characterize and evaluate the damaging effects of radiation. Studies have been conducted in the past to attempt to achieve this; however, little research has involved real-time testing which could provide much more insight into this problem. A better understanding of radiation effects can lead to better counter measures and error prevention techniques.

1.2 Scope

Research in this thesis focuses on the continuation of previous studies in the field of radiation effects on electronics. In particular, hard, stuck-at value faults and delay faults are targeted. Stuck-at faults, when caused by radiation, are known as Single Event Upsets (SEUs) where a signal's value is stuck high or low regardless of its driving logic. When radiation causes degraded performance over time, leading to timing issues, it is known as a Total Ionizing Dose (TID) effect. The research preceding this effort involved the irradiation of a Virtex-4 Mini Module that reported to a Virtex II Pro Evaluation Board with flash memory. In an attempt to simplify the system, this research incorporates Xilinx Virtex-4 FX 12 Evaluation Board reporting directly to a hyperterminal.

1.3 Contributions

The goal of this research is to characterize the effects of different types of radiation on integrated circuits. Three steps are necessary to achieve this goal. First, models of the perceived effects of radiation must be developed. Next, an architecture must be designed and implemented to the FPGA that will detect different types of radiation-induced faults, along with their location, amount, and duration. Finally, this architecture must be implemented onto a FPGA and exposed to an environment simulating radiation exposure while the fault data from the previous step is reported and analyzed.   Ideally, the results will lead to a suitable alternative to physically hardening circuits by using improved designs on FPGAs.

In order to gather relevant data to achieve these results, a fault detection and diagnosis algorithm is designed to identify two types of faults caused by radiation as they occur. Faults are detected with 14 networks of signals that are monitored by comparator and analyzer logic on the FPGA. The signals model static memory and implemented logic that are designed to detect SEUs and TIDs, respectively. Faults are detected when any of the signals is carrying an unexpected value. This is easy to accomplish for static memory because the values of the signals never change under normal operating conditions. Detecting TID faults is more complicated as it requires a timing element that involves determining how far a generated input propagates through a network within one half of a clock cycle. Additionally, bit values written to block memory are checked for bit-flips and reported. The nature of these faults is somewhat unpredictable and can last as short as a couple of nanoseconds. For this reason, the algorithm is designed to run at the maximum allowable frequency so it can detect and report as many faults as possible.

To provide even more information regarding radiation faults, the algorithm is designed to report the location, amount and duration of faults present on a FPGA. The signals of the design are physically placed in separate sections of the FPGA. Each of the sections contains its own set of signals and analyzers. By keeping the networks separated, it can easily be determined which section is experiencing faulty signals based on the corresponding network that is reporting failures. Meanwhile, the analyzer of each network adds up all the faulty signals within the network and a duration counter increments every clock cycle in which a fault is present. All of this information is recorded and reported to a hyperterminal in real-time.

This system can be interleaved with an operational circuit to catch faults before major errors occur. Another objective of the network is to determine if there are specific locations on the FPGA that are more vulnerable to radiation than others. In addition to the real-time fault diagnostic algorithm, this research will provide more information regarding how radiation type, intensity, and length of exposure have different impacts on FPGAs. Primarily, the differences between TID and SEU failures are investigated. The contributions of this research an integral step for the creation of an effective, yet affordable radiation hardened design.

II. Background

This section describes some of the necessary background information needed to understand the fundamentals of fault testing, FPGAs, radiation and its effect on electronics. Previous work in this field will also be covered in this chapter.

2.1 FPGAs

Integrated circuits can be categorized into two main categories: Application Specific Integrated Circuits (ASICs) and FPGAs. FPGAs are unique in that they have the ability to be reprogrammed an unlimited number of times as long as the device is functional. They consist of a field of transistors that form thousands of logic gates. The gates are mainly organized into configurable logic blocks (CLBs). CLBs are made up of basic elements such as look-up tables (LUTs), multiplexors, and flip flops along with routing logic, pass transistors, and I/O pads. Each CLB has the ability to carry out Boolean functions and can be linked together with routing blocks to form larger, more complex logic. The CLBs are connected by a routing matrix that operates via arrays of routing switches [4]. When a design is written in a description language such as VHDL (Very high-speed integrated circuit Hardware Description Language), the FPGA is programmed by making the necessary routing connections between the proper CLBs to implement the design. Furthermore, multiple designs can be routed on the FPGA as long as there is still room for them on the device.

Designs are implemented onto an FPGA with the use of development software. In this research, Xilinx ISE Design Suite 12.4 is used. This software assists in the process of translating VHDL code to a routed map of CLBs that can be programmed to the FPGA. The software simplifies the process immensely with a relatively user-friendly interface. It even optimizes the VHDL code to make the routed design faster and more compact. However, optimization in this research is not necessarily a good thing, as discussed later in this section.

FPGAs have been in high demand because of their flexibility, high performance, low cost, and on-the-fly programming capability. These attributes have made FPGAs desirable for applications such as digital signal processing, software-defined radio, aerospace, and defense systems, ASIC prototyping, medical imaging, computer vision, speech recognition, cryptography, bioinformatics, computer hardware emulation, radio astronomy, metal detection and a growing range of other areas [5]. FPGAs have been used in space operations for over a decade; however, the radiation-filled environment has presented several problems that are still not fully understood. The CLBs, memory elements and routing matrix are susceptible to ionization and physical damage from particles that are present in space. This damage can actually alter the performance of FPGAs. Developers are trying to overcome these problems by finding a way to implement fault deterring logic on FPGAs [6].

2.2 Fault Detection and Diagnosis

Fault detection and diagnosis are an important part of digital circuit design. A fault occurs when an unexpected output results from a given input. The most common fault varieties are bridging faults, delay faults, and stuck-at faults. The single stuck-at fault model is the most versatile model used for testing circuit logic thus far. The model presumes that a fault only affects the connection between gates, not the gates themselves. Therefore, a circuit will have twice as many fault possibilities as the number of connections between gates for the two stuck at values (one and zero). Ideally, it would be possible to test for every single fault. However, this becomes impractical considering the time it would take to test a large circuit with hundreds of inputs. The list of faults can be reduced by eliminating equivalent faults by using methods such as fault equivalence and fault dominance [7].

After the reduced fault list is created, the goal is to create a test that will test for as many of these faults as possible with the fewest number of test input combinations (test vectors). There may be some faults that are not detectable with any test vector. These are referred to as redundant faults. Achieving 100% fault efficiency means that all detectable faults are tested and the maximum fault coverage is attained. Most input vectors will test for multiple faults, so utilizing every possible input combination is not necessary to achieve 100% fault efficiency. By implementing an Automatic Test Pattern Generation (ATPG) program, a minimum number of test vectors will be produced that will achieve maximum fault efficiency. Maximum efficiency will be achieved (not necessarily 100%) because the ATPG program will not test for faults that take too long to detect in larger circuits and it does not test for bridging or delay faults. However, cycling through the ATPG produced set of test vectors and checking the outputs is the quickest way to detect a fault in a circuit.

Once a fault is detected, its location must be determined using a diagnostic approach. There are two ways to accomplish this: statically and dynamically. The static method will continue cycling through the reduced vector list and analyzing the failed vectors along with the incorrect output. This information is then sent to a large reference table to determine the fault location. This method is easy to set up after the look-up table is made. However, this takes a long time to accomplish because every test vector must be applied. Dynamic testing is a much quicker method to diagnose a fault, but its algorithm is much more complicated. Using this method, the program will determine the next test vector to run based on the previous output.

2.3 Radiation Effects on Electronics

As stated before, space-bound systems are at great risk of failure due to radiation, and restoration of failed components can be a difficult task. In space, circuitry is exposed to radiation consisting mainly of protons, electrons, and heavy ions. Long periods of exposure to these energy particles can degrade performance of a device before eventually leading to failure. This best describes TID failures, but radiation can also cause instant failure known as single-event effects (SEEs).

2.3.1 TID Effects.

The TID effect refers to the results of radiation accumulated in a device over a long period of time. This long-term exposure causes the threshold voltage to shift to the point where the device characteristics change. TID also causes increased leakage current and power consumption in addition to timing issues such as propagation delay and slower transition time. A device's insulation and conductive properties are also deteriorated as a result of TID. All of these effects can occur at unpredictable lengths of exposure [8].

Radiation damage in CMOS devices starts with electron-hole pairs created by the radiation source penetrating into the oxide layer of the device. Many of these particles recombine with other atoms immediately, while the rest are left to drift towards an opposing field. The leftover electrons naturally have a greater mobility than the holes, and are able to exit the oxide layer quickly. Meanwhile, the remaining holes are left behind. Under positive bias, the holes will slowly move toward the silicon layer in the same direction as the current. However, hole traps created from imperfections in the device delay the holes on their way, making their travel unpredictable. A larger number of trapped holes in the oxide layer change the overall charge if the oxide, altering the threshold voltage of the device. Figure 1 shows the threshold voltage shift caused by TID for a 'p' and 'n' transistor.

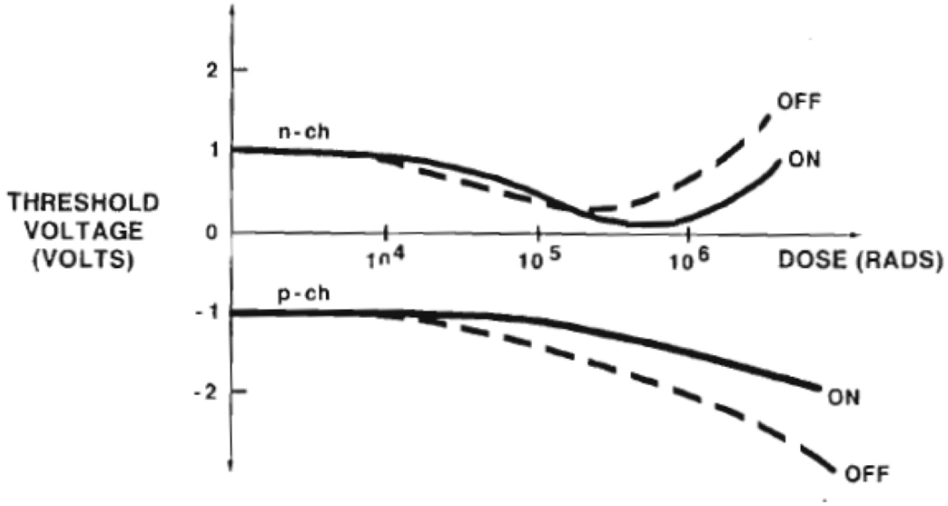


Figure 1. Threshold Voltage of 'n' and 'p' Transistors During Irradiation [10]

Once the holes finally make it to the silicon layer, their absence in the oxide creates electron trap sites at the interface. High temperatures or applied voltage will cause the trapped holes to gradually anneal and the performance of the device will eventually return to normal working condition [9]. A graph depicting the annealing characteristics with respect to time and temperature can be seen in Figure 2. Since newer devices are becoming smaller and smaller, the transistors implemented in them are also becoming smaller along with the width of their oxide layers.

Thinner oxides will trap less total charge and the annealing time will be less overall. Therefore, newer technologies are inherently becoming more radiation resistant [11]. However, TID will still cause problems in devices, regardless of size. The most common source for TID testing is gamma radiation, which is present in space and can be caused

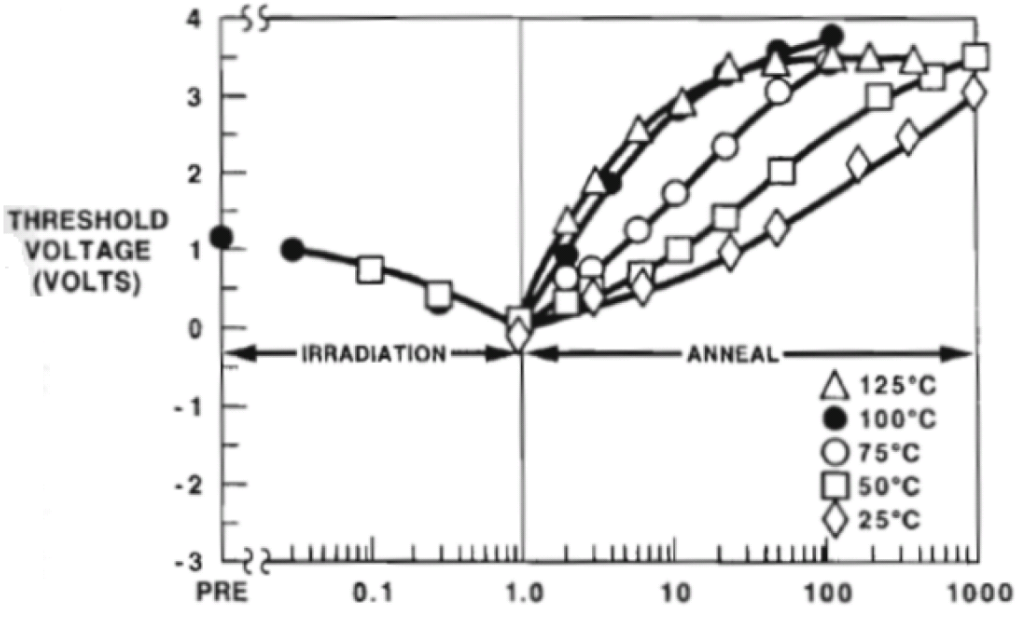


Figure 2. Irradiation and Annealing Effects with Respect to Time and Temperature [10]

by a nuclear blast. For experimental purposes, gamma can be created with Co-60 which emits photons at 1.173 and 1.332 MeV energies. Exposure to these, long range photons causes ionization uniformly over the entire device [10]. X-rays are also a popular source for TID testing, with energy ranges between 100 eV to 100 keV. They have similar effects to FPGAs as gamma irradiation. X-rays can be generated by bombarding tungsten with an electron beam. Even electrons themselves can cause TID through ionization. A van de Graaf can generate an electron beam with energies of 100 keV and 10 MeV [12].

2.3.2 Single Event Effects (SEEs).

Sub-atomic particles found in cosmic rays can penetrate into the FPGA and cause SEEs. When this happens, the high-energy particle leaves behind an ionized path of electron-hole pairs that can cause a temporary device failure as seen in Figure 3[8]. The

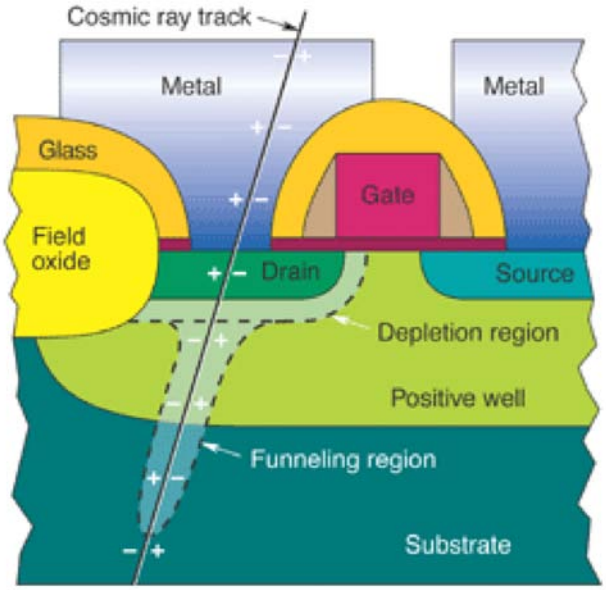


Figure 3. Cosmic Ray Strike Through the Strain of a NMOS Transistor, Leaving Ionized Path of Electron-Hole Pairs [13]

amount of energy contained in the particle determines how much ionization is left in the device, about one electron-hole pair per 3.6 eV in silicon [12]. There are many types of SEEs including single-event latchup (SEL), single-event transients (SETs), single-event upsets (SEUs), and single-event functional interrupts (SEFIs). SELs are the only destructive event out of the aforementioned, but they are all troublesome. SEFIs, while not common, are cause for concern because they affect the device control to the point where the device needs to be reprogrammed. They are also difficult to recognize during a test since the only evidence of one occurring is failed communication between testing equipment and the device or the test fixture crashing altogether [12]. This research focuses mainly on SEUs.

Sources that cause SEUs include protons, alphas, heavy ions and neutrons. Protons are the most tested as they are the prime cause of SEEs in the belts of ionizing particles trapped by the Earth’s magnetic field [10]. The energy levels of these protons are typically greater than 15 MeV and can be reproduced from a cyclotron. Alpha particles represent about 14% of cosmic ray particles and have a similar effect to that of protons. Alphas used in experimentation are generated by the decay of the nuclei of large, radioactive elements. When produced by decay, they generally have much lower energies between three and seven MeV. In space, however, they have much higher energies.

Heavy ions reside in cosmic ray outside of the Earth’s magnetic field with energies of 10 to 1000 MeV [10]. They have less range in silicon than protons and alphas due to

Table 1. Table of Ions Commonly Used for SEU Testing [12]

Ion	Energy (MeV)	Energy per Nucleon (MeV A^{-1})	LET (MeV mg^{-1} cm^{2})	Range in Si
^{7}Li	44	6.3	0.45	253
^{11}B	50	4.5	1.6	91
^{12}C	70	5.8	1.9	105
^{14}N	69	4.9	2.8	73
^{16}O	100	6.25	3.05	95
^{19}F	100	5.3	4.3	73
^{24}Mg	125	5.2	6.8	61
^{28}Si	137	4.9	8.9	53
^{32}S	160	5.0	10.8	53
^{35}B	145	4.1	12.8	43
^{40}Ca	160	4.0	16.3	39
^{58}Ni	132	2.3	28.7	24
^{127}I	100	0.8	47.5	15.5
^{197}Au	127.5	0.65	59.3	1

larger atomic masses but can have higher Linear Energy Transfer (LET) values to cause SEUs. LET is the amount of energy deposited by a particle per unit of track length. In order to cause a SEU, ions must not only reach the transistors, but must also have a high enough LET. The range required to cause an upset is about 10 to 45 MeV $mg^{-1}cm^{2}$. Table 1 shows a list of ions commonly used for radiation testing [12]. Secondary neutrons released from heavy ions have been known to cause SEUs as well. In an experimental setting, a fluence of 10^{15} neutrons per cm^{2} is needed to induce an SEU.

SEUs have a functional effect on FPGAs that result in functional and memory bit flips ('0' to '1' or vice versa). SEUs can be categorized as single-bit upsets (SBUs) or multiple-bit upsets (MBUs). A SBU is classified as a fault that occurs in one place. The FPGA is geometrically divided into columns and rows that map out the individual components on the chip as seen in Figure 4 for the Virtex II. SBUs occur in a single cell from this mapping while MBUs occur in tow or more adjacent cells as seen in Figure 5. Unfortunately, it is not always easy to distinguish between the two since some MBUs can have the same affect on a FPGA as a SBU.

2.4 Previous Work

This field of study has become increasingly popular over the past few years. Organizations such as Los Alamos National Labs [14], AFIT [3], MDA(Ontario) [15],

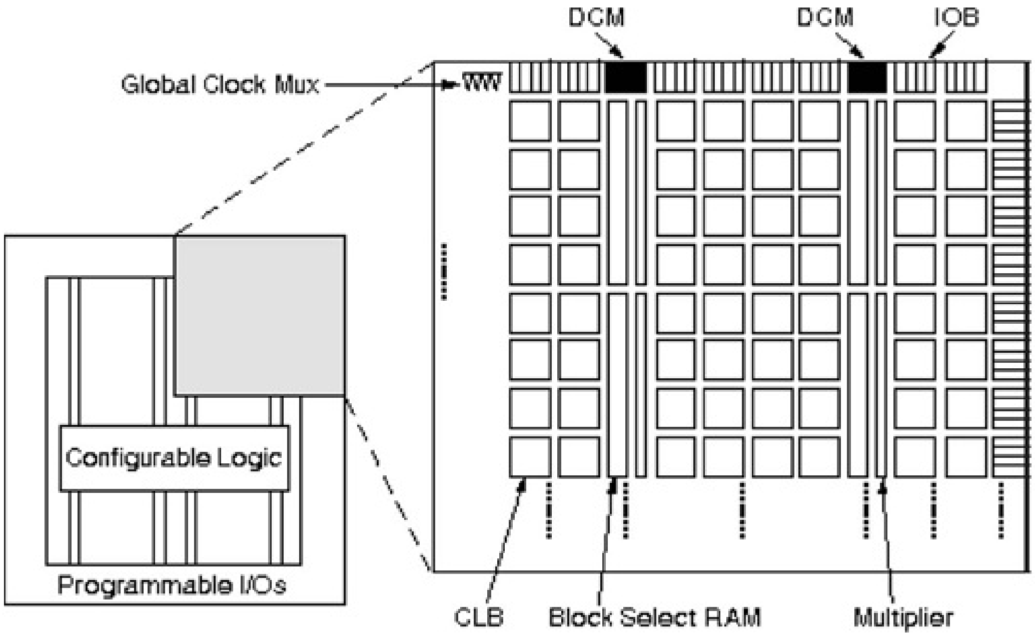
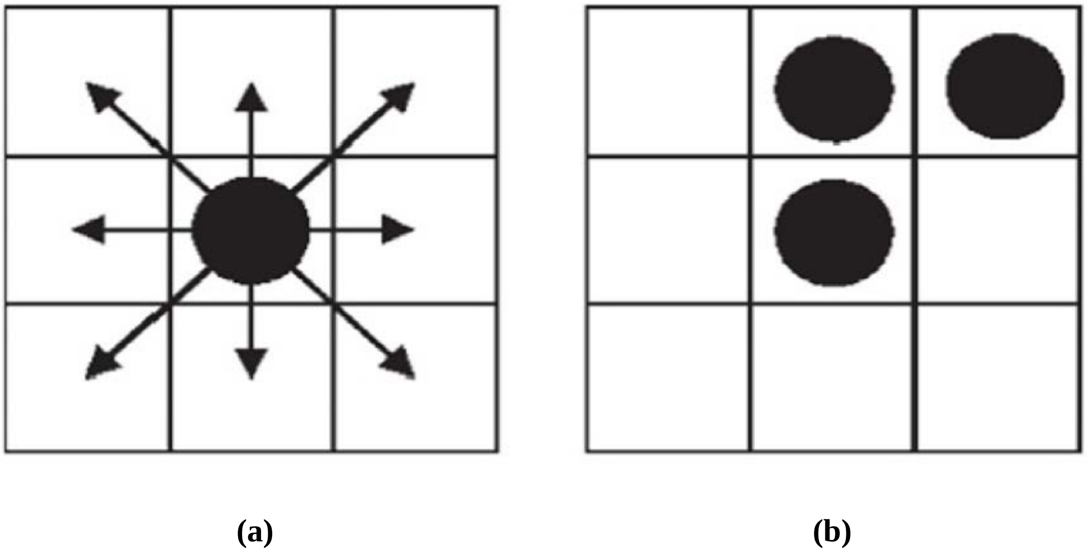


Figure 4. Physical Layout of the Virtex-II [14]

NASA [16], and Italy [17] have recently produced relevant papers in the area of radiation faults in FPGAs.

2.4.1 Los Alamos National Laboratory Efforts.

Some of the most complete analysis of radiation effects on FPGAs has come from the national labs in Los Alamos. Using proton and heavy ion radiation, they have surveyed SEUs on several boards from the Vitrex family. They recorded SBUs and MBUs on various FPGA components exposed to an array of radiation intensities. Table 2 shows the amount SBU and MBUs on each of the boards tested. The increase in MBUs with the newer boards can be attributed to the smaller, compact technology on the newer devices. Another interesting trend is shown in Figures 6 and 7 where BRAM errors become more prevalent with increased radiation on the Virtex 4, while the converse is true on the Virtex 5.



(a) (b)

Figure 5. (a) Upset Adjacency Neighborhood (b) MBU of Three Upset Bits [14]

Table 2. Frequency of Upset Events and Percent of Total Events Induced by Proton Radiation (65 MeV) for Five Xilinx FPGAs

Family	Total Events	1-Bit Events	2-Bit Events	3-Bit Events	4-Bit Events
Virtex	241,166	241,070 (99.96%)	96 (0.04%)	0 (0%)	0 (0%)
Virtex-II	541,823	523,280 (98.42%)	6,293 (1.16%)	56 (0.01%)	3 (0.001%)
Virtex-II Pro	10,430	10,292 (98.68%)	136 (1.30%)	2 (0.02%)	0 (0%)
Virtex-4	152,577	147,902 (96.44%)	4,567 (2.99%)	78 (0.05%)	8 (0.005%)
Virtex-5 (65 MeV)	2,963	2,792 (94.23%)	161 (5.43%)	9 (0.30%)	1 (0.03%)
Virtec-5 (200 MeV)	35,324	31,741 (89.86%)	3.105 (8.79%)	325 (0.92%)	110 (0.43%)

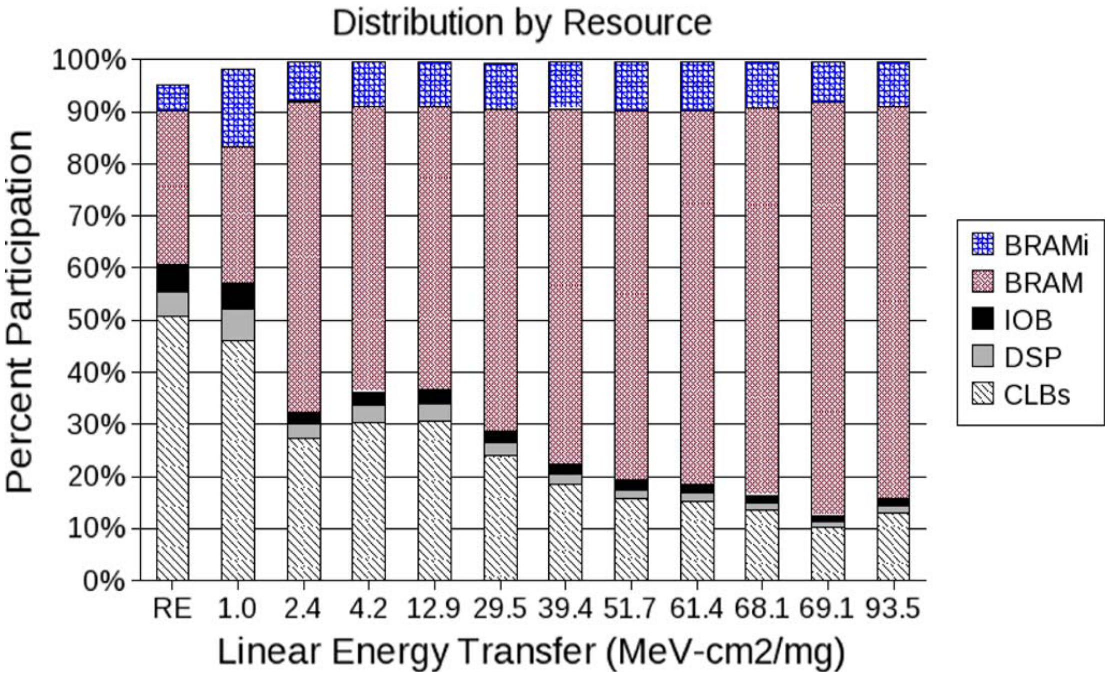


Figure 6. Distribution of Events by Resource on a Virtex-4 Irradiated with Heavy Ions

This information is very informative and groundbreaking as there are still many unknowns regarding FPGAs under radiation. However, there is still some analysis that is yet to be performed. There were not many details regarding how the upsets were detected (probably due to proprietary information), but it is known that the samples were evaluated after radiation. Additionally, there is no information on any trends regarding location of the upsets. For this reason, AFIT has been working on an effort to detect radiation- induced faults as they occur and to characterize any trends in their location on the FPGA to someday be able to prevent these faults.

2.4.2 AFIT Efforts.

The research presented in this thesis is a continuation of research accomplished by past students. Most recently, a successful test setup was built that implemented a 15 foot

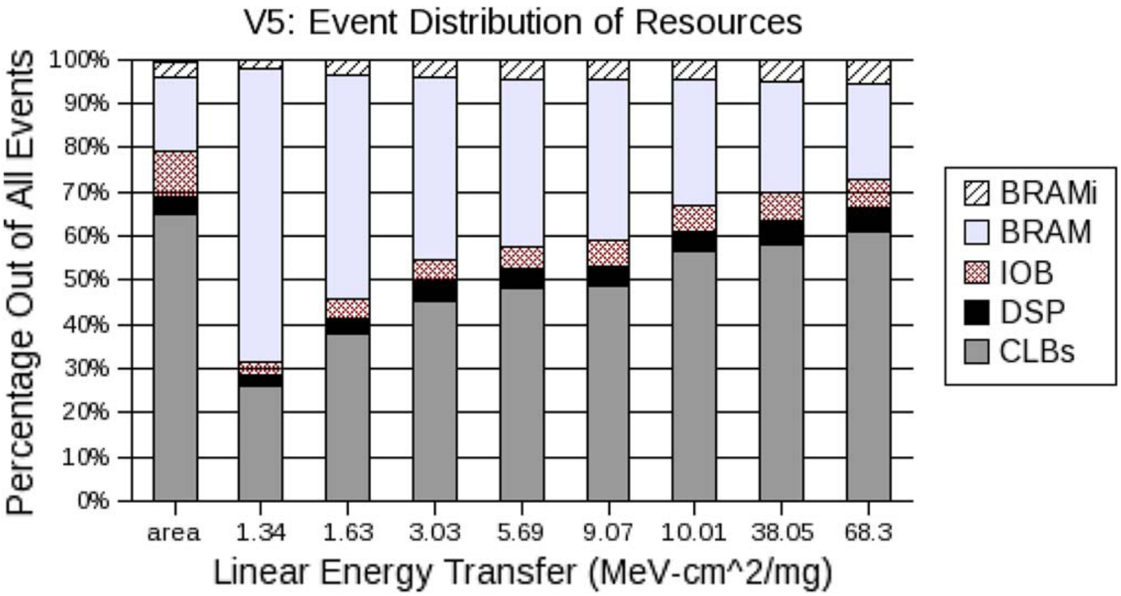


Figure 7. Distribution of Events by Resource on a Virtex-5 Irradiated with Heavy Ions

cable connecting a Virtex-4 being irradiated to a Virtex-II Pro performing the fault diagnosis. The cable was customized for the long, aluminum tube in the gamma reactor at the Ohio State Nuclear facility. This setup also included two feet of wire that separated the FPGA on a mini-module from a baseboard, so the baseboard would not get irradiated along with the mini module. This setup was physically superior to any setup created in the past. However, the algorithm implemented in fault detection and diagnosis was too slow to catch many intermittent faults. The program performing the test was running at 1 MHz while the Virtex II has the ability to run at 100 MHz [3]. The design was also optimized with Xilinx ISE software, meaning it was collapsed down into a structure different and smaller than intended. Therefore, even if the algorithm did detect any faults, there was a good chance that the detected fault would not be in the location reported.

Additionally, the most recent test detection algorithm ran through all possible combinations of test vectors instead of a minimized vector set with the same fault coverage. This created an extra 488 test vectors that had to be tested based on a nine-input design. Therefore, the chances of an intermittent fault slipping going undetected increased dramatically. The latest research also focused on a fault recovery technique known as triple-design triple-modular redundancy (TDTMR). This method used three different styles of adders with the same inputs. The outputs were compared with a voter creating a single point of failure. As it turned out, the only faults that were successfully located were at the voter [3].

Table 3. Virtex-5 CLBs and FFs Fluence to Upset [15]

Fluence to Upset (p/cm^2)	Number of Errors	Upset Signature	Number of Bit Flops	Recovery Method
4.3E9	1	LED 2 Partially On	1029	Re-program FPGA
9.1E9	1	LED 6 On	1177	Re-program FPGA
6.4E9	1	LED 7 On	1542	Re-program FPGA
8.2E9	1	LED 2 Partially On	303	Re-program FPGA
1.1E9	1	LED 2 On	701	Re-program FPGA
1.2E9	1	LED 3 On	280	Re-program FPGA
1.7E9	1	LED 3 On	1135	Re-program FPGA
1.7E9	1	LED 4 On	133	Re-program FPGA
2.0E9	1	LED 5 On	580	Re-program FPGA
2.2E9	1	LED 2 On	486	Re-program FPGA

2.4.3 Recent Detection Methodologies.

A couple of SRAM-based SEU detection methods have been constructed involving functional blocks and memory cells. In [15], multiple sets of counters, multipliers and Block Random Access Memory (BRAM) are implemented onto a Virtex 5 FPGA. If the outputs within any set of these elements do not agree with each other, a LED lights up, indicating a SEU. This allowed somewhat of a real-time detection method, but the in-depth results had to be retrieved after the experiments. The FPGA was irradiated with

Table 4. Virtex-5 BRAM Fluences to Upset [15]

Fluence to Upset (p/cm^2)	Number of Errors	Upset Signature	Recovery Method
1.85E9	2	Single Bit Flip in Separate BRAM Blocks	Re-program FPGA
2.56E9	1	Single Bit Flip	Re-program FPGA
6.19E9	1	Single Bit Flip	Re-program FPGA

various fluences of proton radiation and faults were detected with each method. Each fault was recoverable by FPGA reconfiguration. The results are shown in Tables 3 and 4. These are some of the most useful SEU data to date.

2.4.4 Fault Mitigation Strategies.

As noted in the research from [15], reprogramming the FPGA resolved the SEU issues. Studies in [16] focus on providing geographically separate areas of the FPGA in which applications experiencing faults may automatically relocate to a fault-free area of the device. A series of latches was used as the logic and statistical injected faults were inserted to analyze the effectiveness of the reconfiguration execution. Efforts in [17] also implement an automatic reconfiguration algorithm. This methodology goes one step further in keeping track of which areas of the FPGA are repeatedly getting faults. They claim that this distinguishes between SEUs and TIDs because they state that TIDs are not recoverable. However, many articles, including [18] suggest otherwise. Nonetheless, this study is at the cutting edge as it provides a smarter reconfiguration algorithm and has been tested with real radiation.

III. Methodology

The goal of this research is to combine some of the previous efforts by characterizing the effects of different types of radiation on integrated circuits in greater detail and in real-time. In order to achieve this, models of the perceived effects of radiation must be developed; an architecture must be designed and implemented to the FPGA that will detect different types of radiation-induced faults, along with their location, amount, and duration; and this architecture, implemented on a FPGA; and it must be exposed to a radiation-like environment while the fault data is reported and analyzed.

3.1 Radiation Effects on FPGA Applications

The fault detection system in this research is designed to detect SEU and TID faults, modeled as stuck-at faults and delay faults respectively. Although radiation causes other types of faults, these are the two that are the most predominant and easiest to model.

3.1.1 SEU Effects.

The proposed fault model suggests that stuck-at faults can occur in memory cells and along any signal on the FPGA's routed logic. The stuck-at fault model is based on the assumption that SEU faults consist of routing logic that is stuck-at a value of either ‘1’ or ‘0’. These values commonly represent an open or short circuit. However, in terms of radiation damage, they more accurately indicate a localized ionization that causes a CMOS device to temporarily make a connection to ground or the power supply, depending on the transistor affected. When a SEU is present in an FPGA application, the result or output may be incorrect depending on the input. A fault must be activated and propagated through an application for a bad output to be detectable [7].

The ability to expose all possible faults requires a combination of inputs that will activate and propagate each fault. One specific fault can be detected by analyzing the combination of inputs, with the corresponding incorrect outputs. Multiple faults are more difficult to locate since their effects on the output can alter the effect on the output. Specific combinations of faults present may be impossible to detect depending on the size and complexity of the application. Not only do SEUs impact logic applications, they can also affect memory. Memory faults may or may not be difficult to detect, depending on the severity of the glitch they inflict on the output. In order to properly characterize the effect of an SEU on an FPGA, the number of faults must be tabulated along with their location and duration for various types of radiation.

3.1.2 TID Effects.

TIDs are most commonly witnessed as delay faults. Delay faults are observed when all of the input data do not quite make it through a series of logic gates before the output is recorded. This causes a faulty output that can be difficult to properly diagnose, especially if SEUs and TIDs are being tested for simultaneously. A separate delay test must be implemented to track a signal that traverses through all of the logic blocks. The delay-causing effects of a TID can mimic those of heat-induced slowdown. Therefore, in order to single out a TID, slowdown must occur over a long period of time under radiation without being exposed to extraneous heat. In order to properly characterize the effect of a TID fault on a FPGA, the slowdown of the device, the location of the affected area, the duration, and the temperature must be noted.

3.2 Active Sensor Network

In order to effectively and efficiently detect, categorize, and characterize faults on a FPGA, a specialized system is constructed instead of developing input/output analysis fault detection to a functional circuit design. Traditionally, a functional design such as an adder is implemented with a fault detection algorithm based on inputs that generate faulty outputs. However, this method usually requires analysis of a large number of input vectors depending on the size of the design. In this research, a relatively large circuit design is created with only three inputs and 25 outputs that requires only two input vectors to detect and diagnose all possible faults. The active sensor network is a standalone system with a specialized structure and algorithm designed to collect and report any fault data. It is designed to run continuously and provide the aforementioned characterization information when any faults are detected.

3.2.1 Structure.

The network consists of a series of three-input/output inverter blocks, or sensors, while three signals traverse a section of the FPGA, from sensor to sensor. The three signals are used to detect the different types of faults. Two of the signals are complementary and static, designed to expose SEUs stuck at one and zero. The third signal alternates between one and zero every clock cycle and is designed to measure delay by observing how far the signal gets through the network when the critical value is recorded. The signals originate as a three-bit input vector which is generated and sent to the first sensor. From there, the vector traverses a section of the FPGA, from sensor to sensor. Each bit of the vector is inverted when it passes through a sensor (i.e. vector "100" becomes "011" after it passes through a sensor). The output vector of each sensor not only feeds to the next sensor, but is also sent to an analyzer to determine if it has the correct value.

15 sensor networks are placed over the entire area of the FPGA, each with 29 sensors. Ease of reporting, operating frequency, switching characteristics, and resource availability are factors in determining these amounts,   Four of these sensors act as delay

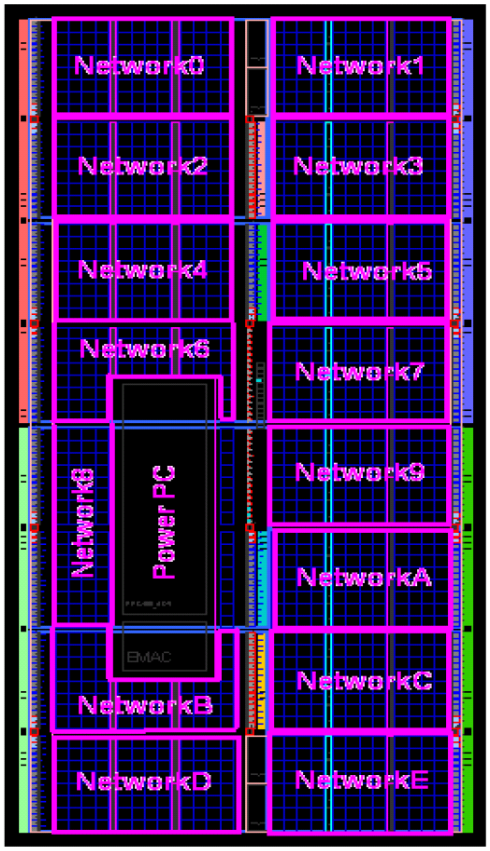


Figure 8. Active Sensor Network Layout on the FPGA

buffers whose outputs are not monitored. This will be discussed further in the next section. The sensors are geographically grouped by network onto distinct areas of the FPGA with ISE Planahead software. The sensors are spaced as evenly as possible within the constraints of the FPGA. Figure 8 shows how the FPGA is divided into the 15 networks in Planahead.

Figure 9 illustrates how the sensors are spread across each network. The red lines represent a high signal, the blue lines represent a low signal, the purple lines represent a signal that alternates between high to low. The black lines represent three, 25-bit vectors comprised of the results of each sensor output that are sent to the analyzer. The sensor layout in Figure 9 is simplified for clarity. The actual signal path traverses back and

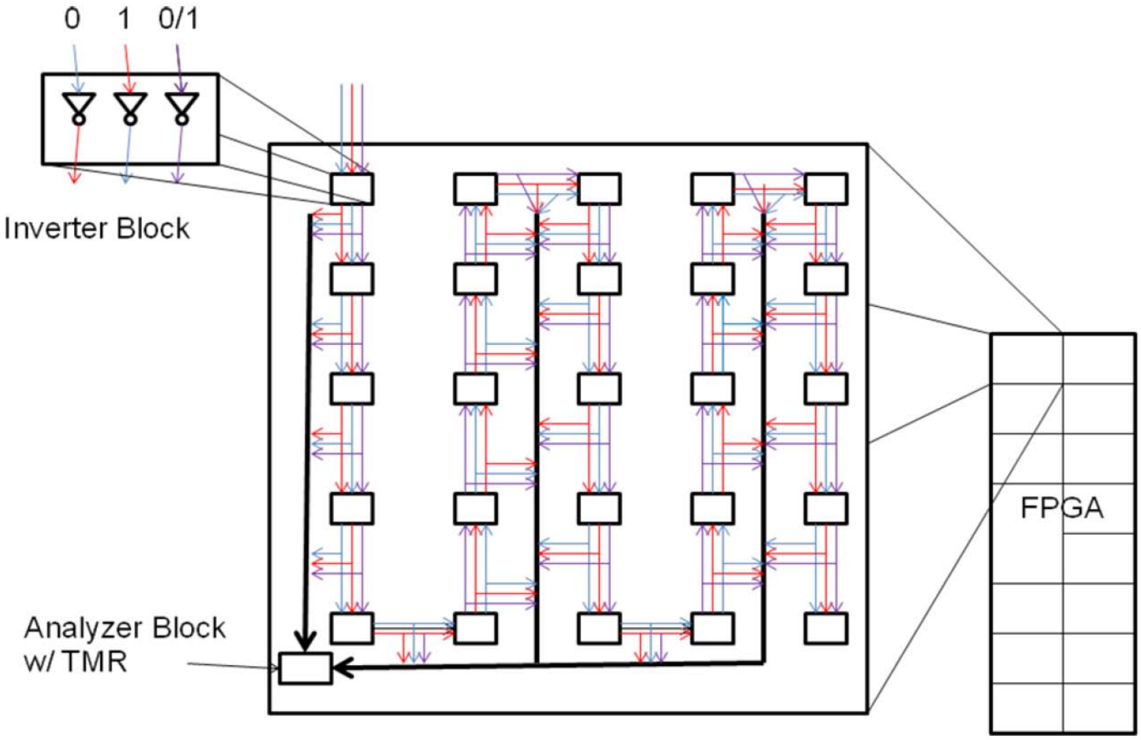


Figure 9. Active Sensor Network Configuration

forth across the FPGA to increase coverage while maintaining an approximate distance between each sensor. More details on the sensor layout are presented in the next section.

3.2.2 Algorithm.

The fault detection and characterization algorithm is developed structurally in VHDL. The two main components are the analyzer and reporter. The analyzers for each network detect fault information for their respective networks, while the reporter compiles the information from every network into one fault code for the entire FPGA.

3.2.2.1 Network Fault Analyzer.

Instead of monitoring inputs and outputs of an entire logic system, the outputs of each sensor are fed into an analyzer that monitors their validity. The analyzer detects any discrepancies between the expected outputs and the actual outputs of the sensors. An initial snapshot of all the sensor outputs is examined at first to determine which of the three signal paths (if any) are incorrect. A faulty static signal indicates the presence of a SEU. If the static signals are correct but the alternating signal is faulty, a TID or delay fault is present. SEUs are sought after first because they will also compromise the TID detection. Therefore, if a SEU is present, TIDs will not be tested for within a network so no false TIDs will be reported. Once the type of fault is determined, a two-bit error designator is generated and sent to the reporter.

It is simple to detect errors from the static signals because the outputs never change under normal operating conditions. The analyzer is designed to compare sensor outputs to expected values every time there is a change in the outputs in addition to every rising edge of the clock signal. Therefore, the analyzer should be able to detect SEU faults that last only a matter of nanoseconds depending on the switching speed of the sensor. Once a fault is detected, the outputs of all the sensors are examined to determine how many faults are present. Since one bad sensor output can alter the expected values of the remaining sensors in the chain, the analyzer compares the output of each sensor to the output of the preceding sensor. This way, a faulty sensor at the beginning of the chain will be reported as one failing sensor instead of several.

Once a SEU is detected from a sensor, that sensor is labeled faulty until a reset signal is received from the reporter. The number of faulty sensors is added up into one value to send to the reporter. Additionally, the analyzer keeps a tally of clock cycles in which a fault is present in the network. This value is incremented when a faulty value is passed to the analyzer and the clock signal switches from low to high. The tally keeps track of the duration of existing faults in the network and is also only reset when the reset signal is received from the reporter. When a SEU is detected, the network analyzer sends the fault count and duration count to the reporter.

TIDs and delay detection is more involved since the dynamic signal changes every clock cycle. There is an inherent delay from sensor to sensor that must be accounted for. This delay is attributed to the time it takes for the signal to get processed by the inverter and the time it takes the signal to get passed to the next inverter. Figure 9 illustrates the timing relationships between the outputs of the sensors in which the delay can be seen. Unlike SEU detection, the analyzer is programmed to capture the sensor output values on the falling edge of the clock. This gives the signal a half clock cycle to reach the end of the sensor network. A clock period of 40 nanoseconds is selected after several dozen simulations involving adjustments of clock speed, number of sensors, and distance between sensors. If the signal does not pass through the last sensor within 20 nanoseconds, a TID error is reported.

Ideally, under normal operating conditions (radiation-free at room temperature) the signal will reach the end of the network with very little slack left until the falling edge. This way, smaller amounts of heat or TID-induced delay will be reported. Figure 10 shows how this delay accumulates from sensor to sensor, up to the falling edge of the clock. Not all of the networks posses the same amount of inherent delay. This is because not all networks have the same shape, so the sensors could not be placed in the exact same orientation in each network. Hence, some network paths have shorter distances between sensors than others. Therefore, the analyzer is catered to the network with the most inherent delay. The rest of the networks have slightly more slack since all the

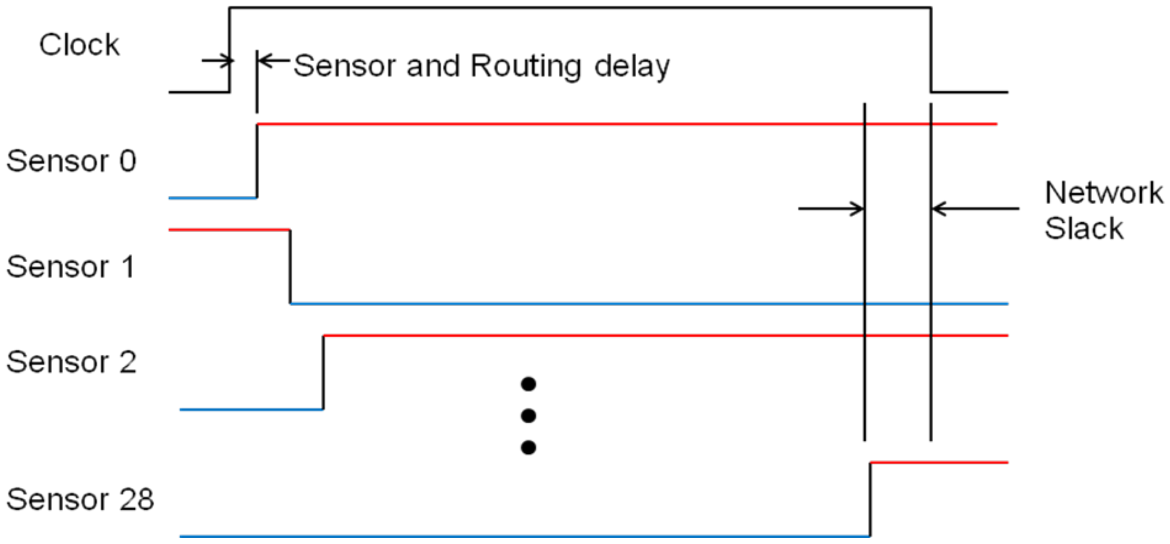


Figure 10. Sensor Output Signal Timing Relationships and Delay Accumulation

networks are driven by the same clock.

To properly detect faults in the alternating signal, a simple counter (incrementing every clock cycle) is devised. The analyzer uses the Least Significant Bit (LSB) of the counter to determine if the sensor outputs of the alternating signal are correct. For example, when the LSB of the counter is zero, the value of the alternating signal from Sensor 0 should be zero; Sensor 1's output should be one, and so on. When the LSB of the counter is one, the opposite values are expected (Sensor 0 outputs a one, Sensor 1 outputs a one, and so on). If a faulty output is discovered, the corresponding error code is sent to the reporter followed by the comparison of individual sensor outputs, similar to the SEU detection. The analyzer compares the outputs of consecutive sensors, starting at the end of the chain to make sure their values are different.

The first two consecutive sensor outputs from the end of the chain with the same value is the furthest point that the signal gets when the clock signal falls. The percent slowdown can be calculated from this point. It is expected that the first delay will be detected by the last sensor in the chain. In this situation, the percent slowdown is estimated to be 3.4 percent since there are 29 sensors (1/29). This equates to a delay of about 690 picoseconds (3.4% of 20 ns). As the delay gets worse, sensors earlier in the chain will begin to detect the fault. The position of the detecting sensors is sent to the reporter. Similarly to SEU detection, a duration counter is tabulated while delay is present and sent to the reporter.

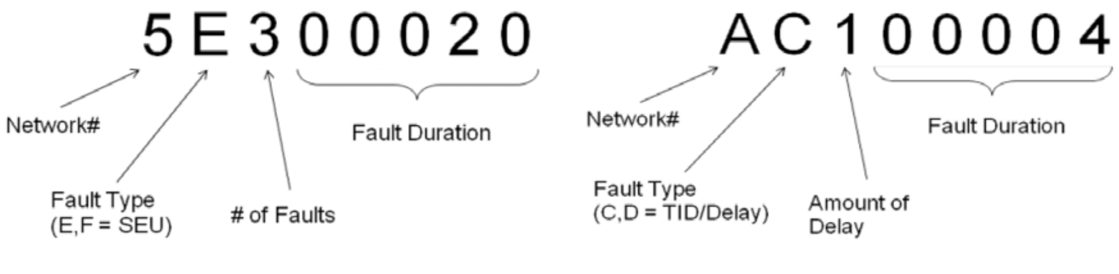
3.2.2.2 System Fault Reporter.

The purpose of the reporter is to organize all the fault data and present it in a timely, easily comprehensive manor. The reporter is limited by the amount of information that can be sent to the hyperterminal at a time. An RS232 serial cable connects the FPGA board to a COM port of a laptop. For this application the hyperterminal can transfer data at a maximum baud rate of 230 kilobits per second. Owing to this constraint, the reporter is designed to output one 32-bit fault code at a time. The reporter gathers the fault data from all of the network analyzers and compiles 32-bit fault codes for each of them. If no faults are present in a network, the code is all zeros.

The fault code describes the network containing faults, the type of fault detected, the number of errors within that network, and the number of clock cycles the error has been present since the last report. In the event of a TID or delay fault, the output will indicate how far the signal got to the end of the network before a result is recorded instead of the total number of faults in the network. The 32-bit code is converted to eight hexadecimal characters when it is sent to the hyperterminal for easier interpretation. The first character is the network designator from ‘0’ to ‘E’. The second character describes the type of fault present. An ‘E’ or ‘F’ indicates a SEU while ‘C’ or ‘D’ indicates a TID/delay. The last bit of the error designator carries over into the value of the following character, representing the number of faults or amount of delay since five bits are needed to cover the range of 1 - 25.

The last five characters are reserved for the fault duration, providing a maximum of 1,048,576 clock cycles (42 milliseconds). Such a long period of time is needed because any given network may have to wait for the rest of the networks to report its data, including transfer time, before the fault code can be read. This ensures that a greater percentage of faults are accounted for. This is important because one limitation of the algorithm is the inability for the reporter to log fault data for a network while that network's data is being sent to the hyper-terminal. During this process, a network's fault code ceases to update so a steady value may be sent off. Unfortunately, this means approximately 6.3% of the time, faults may go undetected. Once a network's fault data has been successfully sent to hyper-terminal, the reporter sends a reset signal to that network so all the error flags, fault counts and durations are set back to zero for the process to repeat. An example of a SEU fault (three SEUs in Network 5 lasting for 32 clock cycles) report output is shown in Figure 11a and a TID example (3.4% slowdown in Network A lasting for four clock cycles) in Figure 11b.

The power PC controls which networks get reported and when. It pulls data from each network one at a time, sequentially, as fast as possible. Data is pulled from Network 0 first, followed by Network 1, and so on up to Network E. After that, the reporter sends the value of the counter signal if any errors are present on the FPGA at the time before



(a) **(b)**

Figure 11. (a) Sample Report Format for a SEU (b) For a TID

the process repeats. The counter serves as a timestamp to indicate when the faults occur. The value of the counter will always be sent every five minutes as a time-check and operational check. Every 15 minutes, the counter resets itself to keep the bit size of the counter down while continuing to provide time-checks.

3.3 Block Memory Bit-Flip Detection

As mentioned in the previous sub-section, the SEU detection portion of the sensor network is designed to simulate memory bits. The total design in this research also tests for bit flips in actual Block Random Access Memory (BRAM). At the beginning of the program, the Power PC writes eight kilobytes of data to BRAM in an alternating "01010..." pattern. The purpose of the alternating pattern is to check for stuck-at-one and stuck-at-zero faults evenly. During fault detection, these values are read 32 bits at a time. If the value read from BRAM is not the original alternating pattern, its value is printed out to hyper-terminal with an "Invalid BRAM" statement. Descriptive fault data is not included with the error statement for two reasons: it is unknown which area of the FPGA specific bits are written to and there is no clock associated with the BRAM so the duration is not known.

3.4 Test Setup

The Xilinx Virtex 4 FX 12 Evaluation Kit is used as the device under test (DUT) for this research. Figure 12 shows the board with the FPGA in the center, power and JTAG connections on the top and serial port on the left. This board was chosen for this research for its ease of use and good value. For many of the experiments, the metal heat spreader lid is removed from the FPGA to observe more faults. The experiments discussed in this section are the injected fault test, a series of thermal tests, and optical flash test.

3.4.1 Injected Fault Test.

Before field testing, the system must validated by simulating faults to avoid damaging boards without useful results. In order to do this, the design code includes three fault injection sites in each network. When activated, these faults are automatically injected at the 200 ns point. This method serves as an initial check to make sure the system is operating correctly. An injected fault consists of a hardwired ‘1’ or ‘0’ into a sensor instead of the output from the previous sensor. A variety of fault combinations (SEU and TID) are injected in each network to make sure each network is catching every fault.

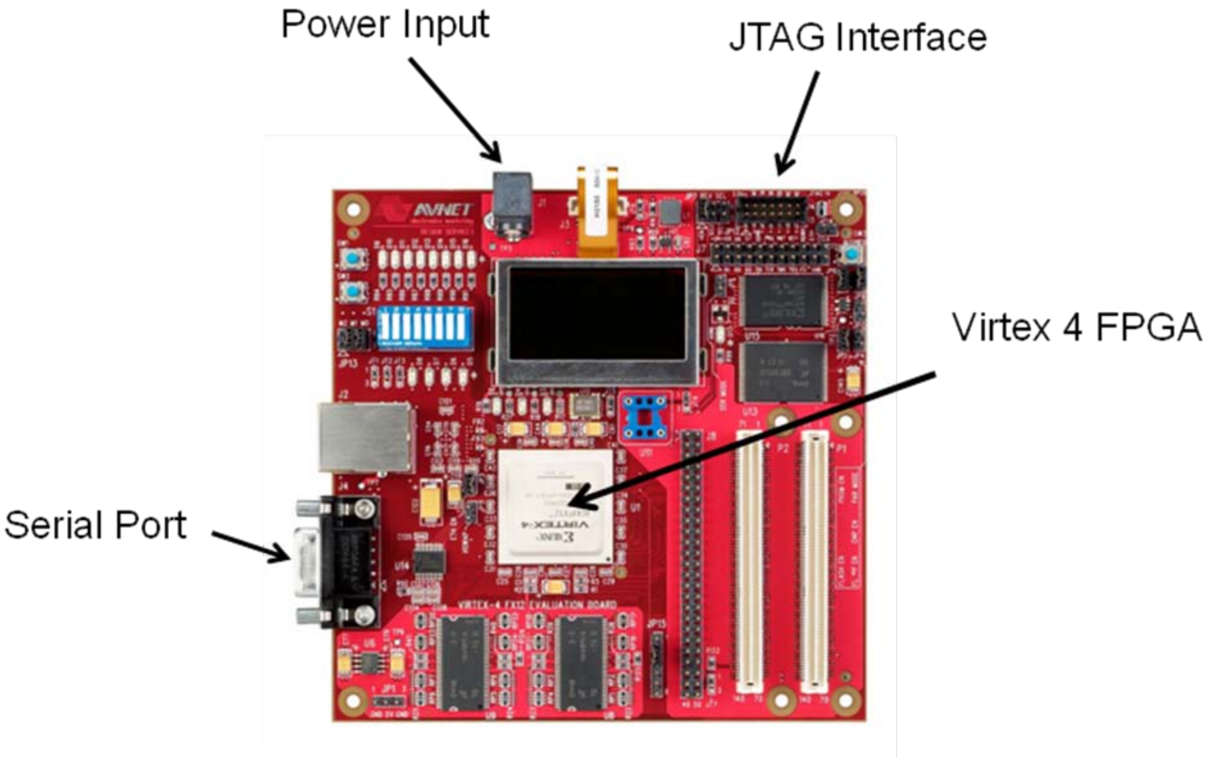


Figure 12. DUT, Virtex-4 Evaluation Kit [19]

3.4.2 Thermal Testing.

A convenient, low-cost method of testing is executed by subjecting the DUT to a number of high temperature tests. The temperature increase is proven to cause delay by shifting the threshold voltage and increasing leakage current [20], similarly to TID effects. The thermal tests conducted in this research are the temperature chamber test, heat gun test, optical laser test, and soldering iron test.

3.4.2.1 Temperature Chamber.

The first test features the use of a heat chamber that subjects the FPGA to high temperatures (up to 110 ^{0}C). A Watlow temperature chamber is used for this experiment. This chamber has a temperature range of -70^{0}C to 180^{0}C with variable rates of change. The DUT is placed inside the chamber connected to a RS232 serial cable, FPGA programmer, and power from an Agilent triple output DC power supply. These connection cables are fed through an insulated hole in the side of the chamber. The DUT is cooled down to -10^{0}C before it heats up to 110^{0}C and finally cools back down to room temperature. Fault data is recorded throughout the experiment.

3.4.2.2 Heat Gun Test.

This test is applied to analyze the effect of instantaneous, high-temperature presence on the outer portion of the FPGA. This is designed to be a destructive test, heating the DUT until it fails while collecting data. A heat gun is set to 649^{0}C and is aimed at the DUT from one inch away for this test.

3.4.2.3 Optical Laser Test.

The optical laser test is conducted based on research from [21] in which flash memory bits are erased by focusing a 50 mW laser on a group of memory cells for several minutes. In this experiment, the same procedure from [3] is followed with a couple of exceptions. Since no faults were detected in [3], only 50 mW lasers are used. The pinhole is also left out in this experiment, as the diameter of the laser beam is much smaller than the area of one of the sensor networks implemented on the FPGA. The first iteration of the test focuses the laser on the upper left corner (Network 0) of the de-lidded FPGA from 12 inches away in an attempt to induce and record a fault in that region. Further iterations include the use of up to three lasers focused on the some corner of the DUT from as close as three inches away.

3.4.2.3 Soldering Iron Test.

The soldering iron test is a more intense method of infusing a localized heating element. In this test, a 316^{0}C soldering iron is applied to a corner of the FPGA while fault data is monitored. Each of the four corners are tested to verify if the system reports the correct region affected by the heat.

3.4.3 Optical Flash Test.

The optical flash test is also setup following the procedures from [3] in an attempt to record bit flips caused by Electrical Magnetic Interference (EMI). Both an unmodified and a de-lidded FPGA are exposed to a flashes ranging from 18.75 W to 600 W from distances ranging from three to nine inches away.

3.5 Impact

This research provides insight of how different types of radiation can affect a FPGA device. Ideally, some clues can be revealed on ways to prevent radiation-induced with FPGA design techniques. For example, there may be an area of the device that is less prone to faults. This fault detection system could also be interleaved with a primary application that may detect faults before they start impacting the main system. Possibly the most valuable prevention method of this research is the slowdown notification. If the system is reporting a 12% slowdown, the operator can decrease the clock-speed by 12% to adjust for the slower performance before larger errors occur in the main system.

IV. Results and Analysis

Many interesting design properties of the Xilinx Virtex-4 are revealed during the making of the fault characterization system. They are presented in this chapter along with the results and analysis of the fault injection, thermal, optical, and EM experiments.

4.1 Design

The placement and routing of the system produces some interesting results that affect the rest of the experiments. The two main concerns of these findings affect the physical layout of the design and the timing of the design.

4.1.1 Layout.

Although the hardware is established structurally in VHDL with the “keep” attribute to prevent optimization, the actual implemented design is substantially different. The biggest difference between the perceived structure and the implemented design is the basic sensor component. The portion of the sensor that inverts the alternating TID detection signal is clearly represented by a LUT in Planahead. However, the portion of the sensor that inverts the static SEU detection signal is not assigned to any type of device whatsoever on the FPGA map. In fact, only sensors nine through 24 list this inverting component in the selectable architecture. They can be confined to a region on the FPGA but ca not be assigned to a specific device like the alternating signal inverters. The absence of the static signal inverters in sensors zero through eight may suggest that they were optimized. This should not be the case, however, since all of the sensors were coded the same way.

4.1.2 Timing.

Timing is another factor that must be accounted for after placement and routing. As mentioned in the previous chapter, several trials are run to determine how many sensors to implement in each network and the proper frequency for the system. During these trials, a timing profile is produced to more effectively relate the delay fault readout to the actual slowdown of a network by finding the average slack between the switching of the last sensor in the chain to the falling edge of the clock signal. Although Xilinx provides some switching characteristics in [22], the switching time of the LUTs being used is not provided. Additionally, the time required for the signal to travel between sensors is unknown. Therefore, the amount of added delay caused by one sensor and the trace length associated with it need to be calculated. This is accomplished by fixing 35 sensors to specific locations in every network, logging the delay readout, and removing two sensors at a time until no errors are reported. The extra, non-reporting sensors are placed, in pairs, between Sensors 0 and 1 and between Sensors 1 and 2 the basic detection logic does not require changing.

The distance between sensors is made as constant as possible within the confines of logic that is already placed. A general sensor path for a network is illustrated in Figure 13. The red dots are the sensors fixed to a location with their sensor number beside them in red. The blue dots are unfixed, non-sensor logic blocks. The signal travels from Sensor 0 to Sensor 24. The first four reporting sensors have two sensors between each of them labeled with an ‘a’ or ‘b’ suffix. The extra sensors basically require the signal to travel all the way to the next block and back before it can proceed, hypothetically tripling the amount of time required for the signal to pass through one main sensor and get to the next main sensor. This methodology is based on the assumption that the switching time and distance are the only two factors affecting the time it takes to pass from one sensor to another.

While performing the analysis, it is noted that the removal of sensors produces a wide range of results from network to network and trial to trial. Some networks show a reported delay decrease of three, while other networks actually report a greater delay when two sensors are removed. This inconsistency is likely due to the rerouting that takes place when the number of sensors is altered. Each time the design is implemented, only the fixed sensors retain their position. The logic that makes up the rest of the system (i.e. the analyzers and reporter) is routed differently each time. While this would not seem to make much of a difference, clearly it does have an impact on the behavior on the

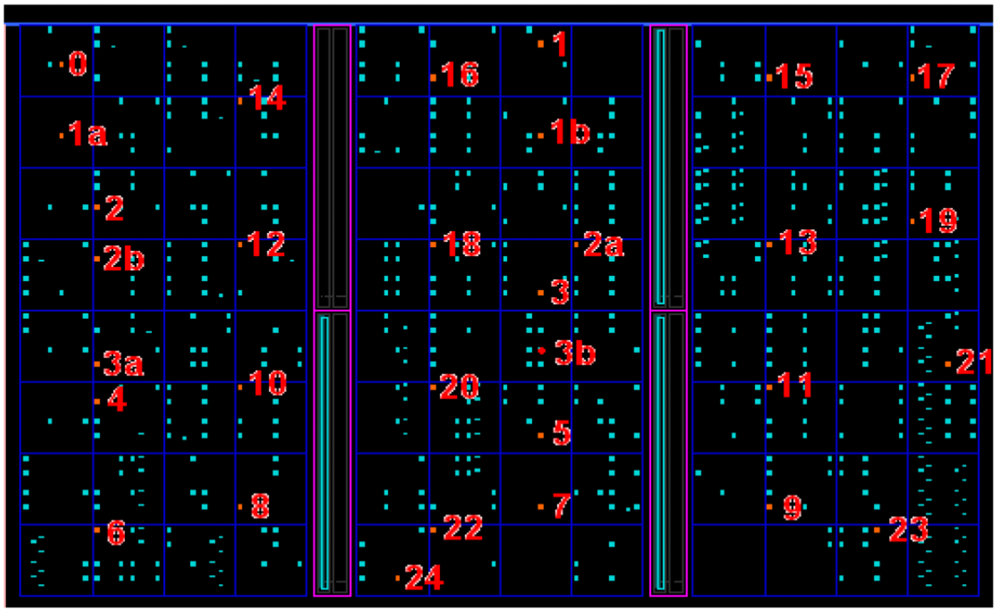


Figure 13. Fixed Sensor Layout for Delay Characterization with Labeled Sensors

results produced by the system. Fixing all of the logic components of the system may produce more consistent results, but would also involve individually placing thousands of CLBs.

The measured result from these trials is the average decrease in reported delay value per sensor removed. The expected value of this figure is one since one less sensor should allow the signal to reach one more sensor along the chain before the clock signal falls. The actual figure is less than one however. With a sample size of 33 delay values, a standard deviation of 0.33, and a variance of 0.11, the average decrease in delay value per sensor removed is 0.77. This, along with the wide range of results between trials proves the assumption previously stated is incorrect and that there are many more variables affecting the delay than just sensor switching speed and separation distance. To add to the uncertainty of this problem, not every network reported data for each of the trials. This is very noticeable in the trials with the maximum amount of extra sensors. This anomaly will be discussed further in the fault injection results section.

Despite the varying results of the delay trials, a baseline can still be established to quantify delay based on the reported value given by the algorithm. Equations (1) and (2) are used to calculate sensor delay (SD) and slack (SL) where HCP represents a half clock period (20 ns) and S is the number of sensors. These equations are based on the assumption that every sensor takes an equal amount of time to receive a signal and pass it to the next sensor. No delay is reported from the system with 29 sensors. Applying S_{29} to Equation (1) gives a SD_{MAX} of 0.69 ns. However, the 31-sensor system yields delay values of one in the majority of the networks. Owing to this, the SD_{MIN} is 0.65 ns. Applying SD_{MIN} and S_{29} to Equation (2) produces a SL_{MAX} of 1.15 ns.

$$SD = HCP/S \tag{1}$$

$$SL = HCP - SD * S \tag{2}$$

From this point, an average decrease in reported delay of one is needed to make the system error free, while utilizing the entire 20 ns for the signal to reach the final sensor. However, two sensors must be removed, decreasing the reported delay value by 1.54 according to the data collected earlier (1.54 = 0.77 * 2). Therefore, a system with zero slack should have an equivalent of 29.7 sensors (31 minus 1/0.7 sensors). Applying these new values to Equation (1) yields a SD_{AVG} of 0.67 ns. This average fits exactly in the middle of the previously calculated range. The resulting SL_{AVG} on a 29-sensor network from Equation (2) is 0.57 ns. Therefore, Percent Slowdown can be calculated ± 3% with Equation (3) with DV being the reported delay value.

$$PS = (20 - (0.67 * DV + 0.57))/20 \tag{3}$$

4.2 Injected Fault Test

The injected fault tests are a good baseline to check for proper system functionality. However, not all faults can be simulated due to limitations of the Power PC synthesis. These limitations along with the results of the injected fault test are discussed in this section.

4.2.1 Injected Fault Limitations.

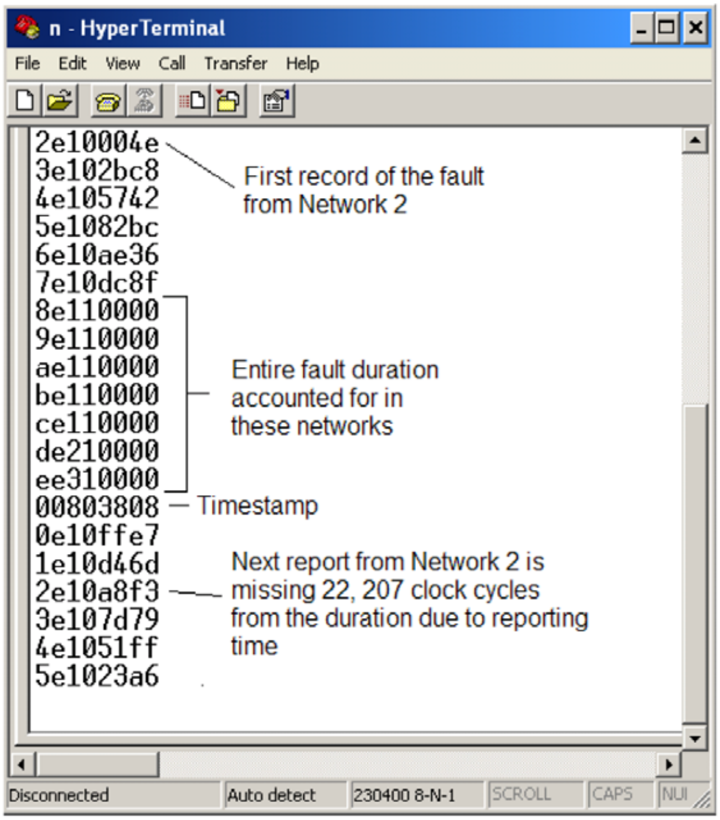
The fault injection component of the system is designed to hard-code high or low signals into three areas of each network to emulate SEUs. Various lengths of fault presence are tested. However, the injected faults are driven by the counter so they can be injected when the counter value changes. Since the counter changes every 40 ns, the shortest fault length testable is also 40 ns. Additionally, the Power PC does not recognize “wait” statements, which happen to be the basis of the injected delay fault. A sensor is designed to pass its value to the next sensor 10 ns after receiving an input when a delay fault is injected. Since the Power PC cannot relate the clock frequency to a specified period of time in the hardware, delay faults cannot be modeled.

4.2.2 Analysis.

The system is successfully able to detect injected stuck-at faults of various durations down to the shortest of 40 ns. The expected fault numbers and durations are also recorded. For faults lasting longer than one reporting cycle, a lapse of duration coverage is observed. For example, Figure 14 shows a report of 14 faults injected to each of the 14 networks for 65,536 clock cycles (2.6 ms), all stating at the same time. Network 2 is the first to report the fault, 78 clock cycles after it is injected. The next time Network 2 reports, the duration is 43,251 clock cycles equating to 22,207 clock cycles where the fault is present but its duration is unaccounted for. Therefore, the timing and location of the fault can affect the reported duration (Networks 8 – E in this example record the entire fault duration). One aspect of the fault injection test is unexplainable, however, as most test runs have one or two networks do not report their faults. This anomaly is not explainable at this time as every fault is designed exactly the same and there is no recognizable pattern of which network will not report.

4.3 Temperature Chamber Test Results

Three attempts of the thermal test are conducted. The first two produce results that are not very valuable. The thermal chamber is heated up to the rated temperature of the FPGA at 90^{0}C where data collection was ceased. Only two networks experienced delay of just six percent. While the detection of delay is a good start, there is not much to quantify from these two test runs. However, from these runs, a correlation between current consumption and failures is observed. Errors appear to start being reported when the current consumption is greater than 476 mA. This relationship becomes the basis of



```
2e10004e
3e102bc8
4e105742
5e1082bc
6e10ae36
7e10dc8f
8e110000
9e110000
ae110000
be110000
ce110000
de210000
ee310000
00803808
0e10ffe7
1e10d46d
2e10a8f3
3e107d79
4e1051ff
5e1023a6
```

First record of the fault from Network 2

Entire fault duration accounted for in these networks

Timestamp

Next report from Network 2 is missing 22, 207 clock cycles from the duration due to reporting time

Figure 14. Hyperterminal Report of Fault Injection Test

the third test.

In the third test, the chamber is heated up to 110^0C while the ambient temperature and the current consumption are recorded every minute. Once again, the errors begin once the current reaches 476 mA. The current consumption more accurately reflects the FPGAs core temperature and represents the increase in current required to counter the increase in leakage current. The core temperature is not calculated since the current is easier and more practical to obtain with the power supply. The relationship between faults and current cannot be observed in the hyperterminal during the test since the outputs are printing out too fast to read because they are being reported in real time. The comparison is made after the experiment by comparing the current/time profile with the timestamps of the faults reported.

In this attempt, six of the networks report delay. The most slowdown reported is 10% in Networks B, D, and E. Just as in the first two experiments, Network D experiences delay longer than any other network. A map of the FPGAs affected regions based on the thermal experimental data is illustrated in Figure 15. The green blocks represent the networks reporting six percent delay while the red blocks represent the networks reporting 9.9 percent delay. Most of the affected networks reside on the lower half of the FPGA.

Figure 16 depicts a graph of the system current draw versus time and temperature of the experiment. The percent slowdown of Network D is profiled here. The blue region indicates a detected delay fault, green indicates a six percent slowdown, and red indicates a 9.9 percent slowdown. The delays detected without a slowdown are those that appear in the report as a delay fault with a null delay value. This happens when the initial analysis of all of the sensor networks uncovers a fault, but the individual sensor output comparisons are still reporting no faults. The relationship between current and delay is taken one step further as each network starts to experience delay on the positive slope of the graph at the same current value that is stops experiencing delay on the negative slope within seven mA. Even more convincing, is the degree of delay also shares the same

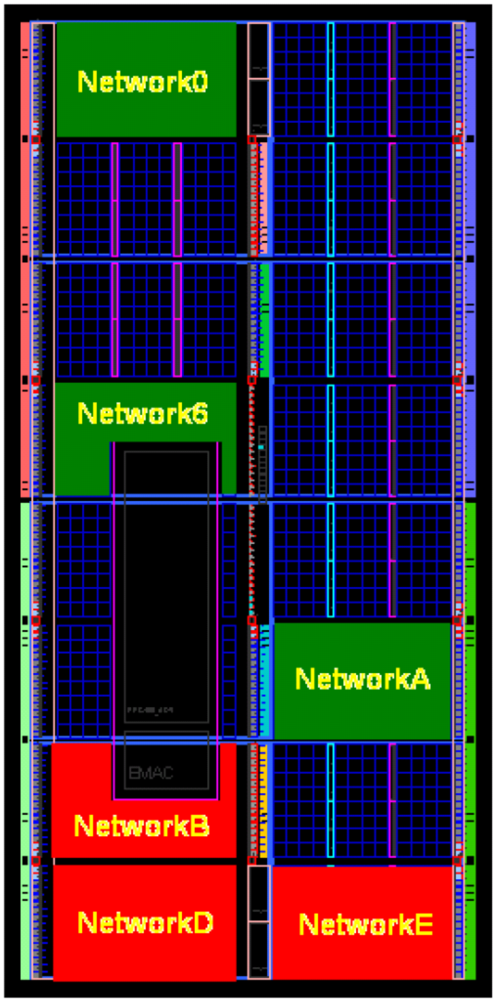


Figure 15. Thermal Test Delay Distribution

current relationship on the positive and negative slopes of the current/time line.

The difference in network slowdown can be attributed to a number of factors. Part of the intent of this research is to identify regions of the FPGA that may be more vulnerable to specific types of faults. However, the results from the design analysis indicate that the difference in slowdown can be due to a difference in slack caused by wide variations in network implementation. This may be the case for the six networks that detected faults at different currents, but does not explain why the rest of the networks did not detect any delay at all. In order for one network to report a 10% slowdown and an adjacent network reporting no delay, the difference in slack would have to be at least 1.3 ns. While this difference is possible, it exceeds the theoretical maximum slack calculated in the

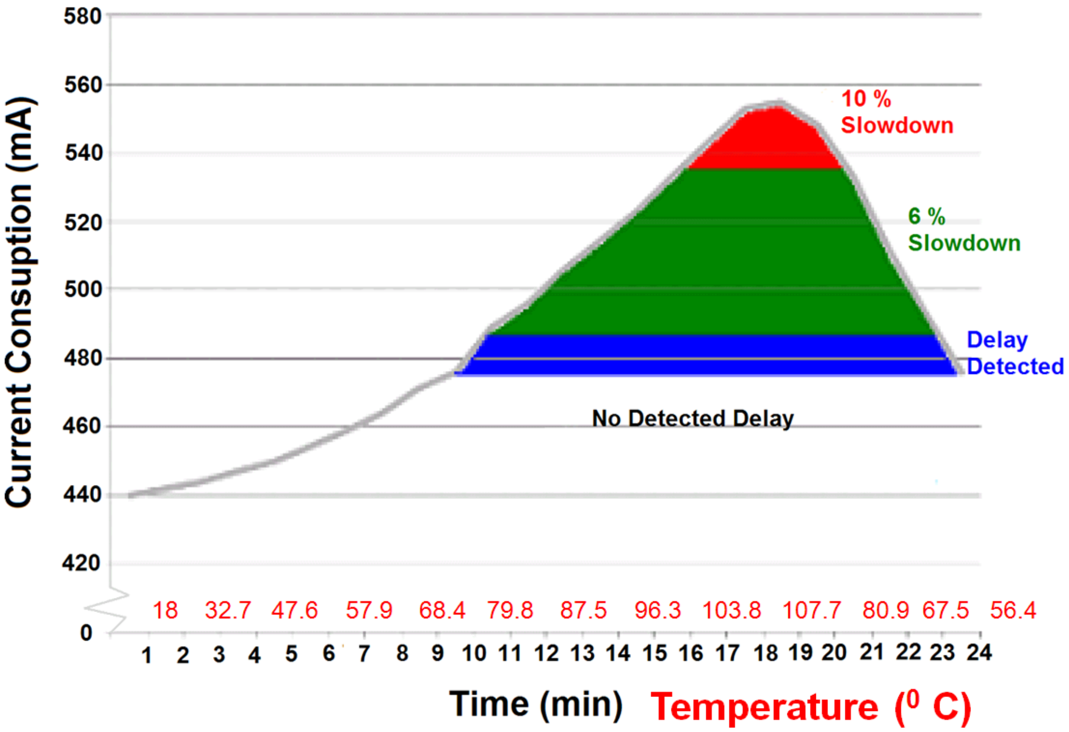


Figure 14. Degree of Delay for Network D Based on Current Consumption

previous section and is unlikely. The region of the FPGA may correlate with slowdown vulnerability, but several variables need to be addressed, such as non-reporting networks, non-sensor logic usage and clock slowdown before such a claim can be made.

4.4 Heat Gun Results

The heat gun experiment is designed to take the temperature chamber one step further by exposing the DUT to conditions outside of its rated temperature range. The extreme heat quickly penetrates the FPGA, causing delay faults within seconds. All but two of the networks report delay. Since the minimum reported slowdown is 16.3%, it is assumed that these two networks experience delay, even though it is not reported. Network 0 suffers the worst slowdown of 20.7%. A graph of Network 0's percent

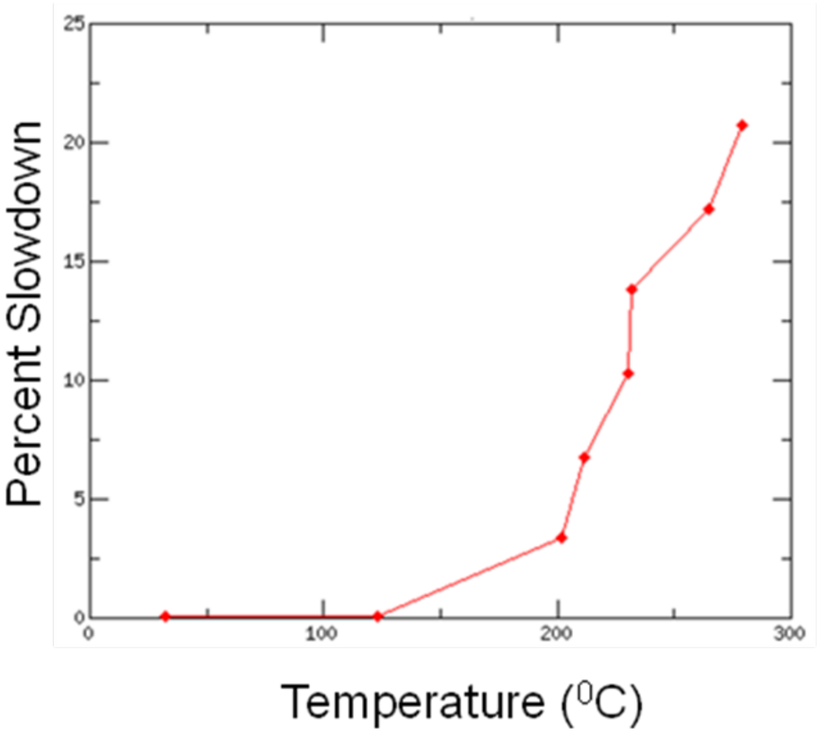


Figure 15. Percent Slowdown vs. Temperature from Heat Gun Test

slowdown over time is shown in Figure 17. The maximum current draw reaches four amps when the device fails. Despite reaching the point of failure, the board remains operational after it is cooled down and reprogrammed. The current draw at room temperature after this test has risen to 506 mA, indicating some permanent damage caused. Delay was the only fault detected in this experiment.

4.5 Optical Laser Results

With a quicker, more sensitive design that incorporates BRAM, the optical laser test from [3] is attempted again to record a bit flip generated by a laser as was accomplished in [21] as well as recording the location and duration of the bit flip. Unfortunately, no faults are recorded. Even three 50 mW lasers focused onto the same point on the FPGA did not induce a fault of any kind. The current draw is also monitored in this experiment. After an hour of each level of laser intensity, the maximum change in current is only two mA; nowhere near the current increase noted to cause a delay fault in the temperature chamber. The lack of results in this experiment can likely be due to the construction of the FPGA. The impact of a laser would have to penetrate the layers of oxide and copper to reach the memory cells and logic blocks since the Virtex 4 is a flip chip design. In an attempt to assist the laser with a more direct route to the transistors on the FPGA, the laser beam is aimed at an angle toward the side edge of the FPGA and even on the capacitors on the other side of the board. After an hour of each attempt, it is determined that the CLBs and BRAM cells of the Virtex-4 are not affected by laser exposure.

4.6 Soldering Iron Results

The soldering iron test uncovers yet another aspect to consider involving the construction of the FPGA. The expected result is to record faults in different regions when different corners of the FPGA are attacked by localized heat; in which case Networks 0, 1, D, and E would report delay for each of the corner attacks. This result is not recorded. Instead, the same networks report delay each time the soldering iron is applied. When the corners on the left side of the FPGA are heated, Network 3 is the first to report delay while Network 6 reports first when the right side is heated. This indicates that the location of the network has some impact. However, the same four networks (3, 6, 7, and D) report delay each time, regardless of the location being heated, indicating that there is another factor involved. The maximum slowdown recorded in these test runs is only six percent, and the faults disappear almost immediately after the soldering iron is removed.

These unexpected results are likely due to the unaccounted for third dimension of the FPGA. [24] describes the FPGA as a stack of up to 22 layers consisting of silica substrate between two plates of copper in which the traces and pads are etched out from. These layers are stacked on each other, separated with insulating substrate. Additionally, there are planes of copper amongst the layers serving as power distribution areas. With this information in mind, it is very likely that a localized heat source could penetrate to the first of these layers where the heat is dissipated over the entire two dimensional plane of the FPGA. At this point, the networks that are routed closest to the surface will experience delay before those closer to the board.

Table 5. Optical Flash Soft Error Points

Intensity (W)	Soft Error Distance (cm)	Logic SEU Observed?	BRAM SEU Observed?	TID Observed?
18.5	None	No	No	No
35	7.5	No	No	No
70	15	No	No	No
140	15	No	No	No
280	15	No	No	No
560	22.5	No	No	No

The layers of the FPGA are connected with metal vias. However, the majority of the localized, relatively weak heat is dissipated by the power plates. Planahead does not account for the depth of the FPGA when allowing the user to place and route logic blocks. It can therefore be assumed that the networks reporting delay in this particular experiment are placed toward the surface of the die. This finding creates an inconsistency in modeling TID faults with heating experiments. While the affects on leakage current and threshold voltage remain, the gradual ionizing properties would not transfer across the FPGA like heat does.

4.7 Optical Flash Results

Again, repeating experiments in [3] with a quicker, more robust system, the optical flash test is conducted. However, once again, the desired results from this experiment are not achieved as no fault data is recorded. Tests conducted with the heatspreader protecting the FPGA proves completely resistant to the Electrical Magnetic Interference (EMI) created even from the most powerful flash of 600W from as close as three inches away.

No faults of any kind are detected, and the system continues to operate unaffected. When the heatspreader is removed, the device's functionality is dependent upon the proximity and the intensity of the flash. Table 6 contains the distance at which each of the intensities tested causes the FPGA to crash.

V. Conclusion

Overall, some interesting finds are uncovered in this research. The system created in this research shows potential, but has a lot of improvements to be made before it can become the all-encompassing fault detection and characterization tool it was intended to be. This section summarizes the contributions of this research, and some recommendations to future improvements that can be made to turn this system into a useful tool for FPGA usage in radiation-prone environments.

5.1 Contributions

While a few organizations have created fault detection methods that either determine fault types, amount, or location, this research combines the two into one system. It also provides fault data that is not being accounted for in other methodologies such as fault duration and delay. Even though there are already automated fault reconfiguration systems being developed now, the addition of the methodology presented in this research will make them more robust and more effective.

5.2 Future Work

There are several improvements and applications this design can be used for in future projects. More effort needs to be devoted to the system design for it to become more useful in further experimentation and being implemented as a radiation detection system in operational uses.

5.2.1 Design.

A better understanding of the design properties will help improve this system immensely. The delay and routing properties are somewhat of a mystery to most designers that do not have access to in-depth fabrication layouts. Understanding the FPGA construction will make it easier to turn this system into a finely-tuned delay detection system.

5.2.2 Experimentation.

This system can be exposed to several radiation sources for fault characterization. As mentioned in Chapter 2, SEUs can be induced by proton, neutron, heavy ion, and alpha exposure. Preferably, the radiation could be centralized onto a small area of the FPGA so the location to be directly linked to a radiation-induced fault. A mask could be made to shield all of the FPGA except the area to be irradiated. The thickness of the shield should be thick enough to provide the right amount of stopping power for the selected source at the selected energy. Protons and neutrons make the most sense for this testing as they are more readily available. While alphas also fit into this category also, their energy level from common sources, such as Americium, is only around 5 MeV. An alpha at this energy will penetrate 23 micrometers into silicon, not enough to cause a SEU.

Meanwhile a true TID experiment could be conducted to compare the results to the thermal experiment from this research. It would be interesting to see if the same current usage to delay relationship exists with a true TID. Gamma, x-ray, or electron radiation would be a good candidate for these tests. The fault location methodology would not be tested, however, since the exposure of these sources is so uniform. A mask would not work in this case since a very thick and dense material would be needed that would be too difficult to punch a small hole through. Although, an unmasked DUT would more accurately simulate radiation exposure in space.

5.2.3 Applications.

As mentioned previously, this sensor network can be implemented in conjunction with other applications on newer boards as a early warning system in the event of radiation threats. Future iterations of this system can implement automated reconfigurations of the routed logic to avoid areas of the FPGA experiencing faults. Additionally, an automated system clock can be controlled by the delay monitor, slowing down by the percentage of slowdown detected. This system could be incorporated with the efforts described in [17], making it more accurate and robust. After all, after the faults are characterized, the next step is to prevent, mitigate and correct them.

Bibliography

[1] Xilinx Website, FPGA vs ASIC, Inc., Xilinx, February 2010 <http://www.xilinx.com/company/gettingstarted/fpgavsasic.htm>

[2] C. Carmichael, *Triple Module Redundancy Design Techniques* for Virtex FPGAs, xAPP197 (v1.0), Xilinx Corp., 2001.

[3] Naber, Nathan P. *Real Time Fault Detection and Diagnostics Using FPGA-Based Architectures*. MS Thesis, AFIT/GCEENG/10-04. Graduate School of Engineering & Management, Air Force Institute of Technology, Wright-Patterson AFB OH, March 2010.

[4] F. Lima, C. Carmichael, J. Fabula, R. Padovani, R. Reis, "A Fault Injection Analysis of Virtex FPGA TMR Design Methodology," *RADEC*, 2001.

[5] Lofru Technologies Website, Application Development, April 2010 <http://lofru.com/Solutions/ApplicationDevelopment.html>

[6] K.A. LaBel, "Radiation Effects & Analysis". *NASA*. Sep 2009 <http://radhome.gsfc.nasa.gov/top.htm>.

[7] Bushnell, M.L. and V.D. Agrawal. Essentials of ElectronicTesting for Digital, Memory, and Mixed-Signal VLSI Circuits. Boston: Kluwer Academic, 2000.

[8] C. Claeys, E. Simoen, Radiation Effects in Advanced Semiconductor Materials and Devices. Berlin: Springer-Verlad, 2002.

[9] J. Petrosky, Radiation Effects on Electronic Devices: Theory, Modeling and Experiment. NENG660 Course Notes, Air Force Institute of Technology, 2007, n.d.

[10] T.P. Ma, P.V. Dressendorfer, Ionizing Radiation Effects in MOS Devices and Circuits. New York: Wiley Interscience Publications, 1989.

[11] J. Scarpulla, A. Yarbrough. "The Effects of Ionizing Radiation on Space Electronics" *Crosslink* vol. 4, no. 2, June 2003, pp. 15-19.

[12] Holmes-Siedle, Andrew and Len Adams, Handbook of Radiation Effects. Oxford: Oxford University Press, 2002.

[13] Wang, Weizhong, “Sense Amplifier Basesd RADHARD Flip Flop Design”, *IEEE Transactions on Nuclear Science*, vol. 51, no. 6, December 2004.

[14] Quinn, Heather. “A Test Methodology for Determining Space Readiness of Xilinx SRAM-Based FPGA Devices and Designs.*”* *IEEE Transactions on Instruments and Measurement*, vol. 58, Oct. 2009, pp. 3380-3395.

[15] Hiemstra, David. *Single Event Upset Characterization of the Virtex-5 Field Programmable Gate Array Using Proton Irradiation*. Brampton, Ontario: MDA, July 2010

[16] Zick, Kenneth. *On-Line Characterization and Reconfiguration for Single Event Upset Variations.* Langley, VA: NASA Langley Research Center GSRP Fellowship.

[17] Bolchini, Christina. “Fault Classification for SRAM-Based FPGAs in the Space Environment for Fault Mitigation,*”* *IEEE Embedded Systems Letters*, vol.2 no.4, Dec. 2010, pp. 107-110.

[18] Wang, J.J. “Total Ionizing Dose Test Report,” Report No. 03T-RT54SX32S-T25JS004, March 12, 2003

[19] *Avnet Inc.* Xilinx Virtex-4 FX12 Evaluation Kit User Guide. 2005.

[20] Fang, David. *Self-Timed Thermally-Aware Circuits.* Cornell, NY: Computer Systems Laboratory Electrical and Computer Engineering, Cornell University.

[21] Skorobogatov, Sergei. *Local Heating Attacks on Flash Memory Devices.* Cambridge, UK: Computer Laboratory, University of Cambridge.

[22] *Xilinx.* Virtex-4 FPGA Data Sheet: DC and Switching Characteristics. 2009.

[23] *Xilinx.* Virtex-4 FPGA Packaging and Pinout Specification. 2008.

[24] *Xilinx.* Virtex-4 FPGA PCB Designer's Guide. 2008.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 074-0188*

1. REPORT DATE *(DD-MM-YYYY)* 24-03-2011	**2. REPORT TYPE** **Master's Thesis**	**3. DATES COVERED** *(From – To)* Oct 2009- Mar 2011

4. TITLE AND SUBTITLE

Radiation Induced Fault Detection, Diagnosis, and Characterization on FPGAs

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Getz, Thomas, B, III

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GE/ENG/11-12

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Dr. Robert L. Herkoltz
Program Manager - Information Operations and Security
Air Force Office of Scientific Research (AFOSR/RSL)
875 N. Randolph Street, Suite 325, Room 3112
Arlington, VA 22203-1768
(703) 696-6565; robert.herkoltz@afosr.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

AFOSR/RSL

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

The development of Field Programmable Gate Arrays (FPGAs) has been a great achievement in the world of microelectronics. One of these devices can be programmed to replace the need for thousands of individual specialized devices. Despite their great versatility, FPGAs are still extremely vulnerable to radiation from cosmic waves in space. Extensive research has been conducted to examine how radiation disrupts FPGAs. This research incorporates and enhances current methods of radiation detection. A design is created that has the ability to detect flipped bits and delay errors caused by radiation along with their location, amount and duration. All of this is accomplished and reported in real time. The design requires more testing, but once that is done this system can be incorporated with FPGA reconfiguration methods that automatically place application logic away from failing errors of the FPGA. This system has great potential to become a valuable tool in fault mitigation.

15. SUBJECT TERMS

FPGA, radiation induced faults, single event upset, total ionizing dose

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Dr. Yong C. Kim (ENG)
REPORT U	**ABSTRACT** U	**c. THIS PAGE** U	UU	158	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 785-3636, x4620; Yong.Kim@afit.edu