University of Kentucky
UKnowledge

University of Kentucky Master's Theses

Graduate School

2008

SINGLE EVENT UPSET DETECTION IN FIELD PROGRAMMABLE GATE ARRAYS

Shadab Gopinath Ambat
University of Kentucky, shadabambat1@gmail.com

Right click to open a feedback form in a new tab to let us know how this document benefits you.

Recommended Citation

Ambat, Shadab Gopinath, "SINGLE EVENT UPSET DETECTION IN FIELD PROGRAMMABLE GATE ARRAYS" (2008). *University of Kentucky Master's Theses*. 511.
https://uknowledge.uky.edu/gradschool_theses/511

This Thesis is brought to you for free and open access by the Graduate School at UKnowledge. It has been accepted for inclusion in University of Kentucky Master's Theses by an authorized administrator of UKnowledge. For more information, please contact UKnowledge@lsv.uky.edu.

Abstract of Thesis

SINGLE EVENT UPSET DETECTION IN FIELD PROGRAMMABLE GATE ARRAYS

The high-radiation environment in space can lead to anomalies in normal satellite operation. A major cause of concern to spacecraft-designers is the single event upset (SEU). SEUs can result in deviations from expected component behavior and are capable of causing irreversible damage to hardware. In particular, Field Programmable Gate Arrays (FPGAs) are known to be highly susceptible to SEUs. Radiation-hardened versions of such devices are associated with an increase in power consumption and cost in addition to being technologically inferior when compared to contemporary commercial-off-the-shelf (COTS) parts. This thesis consequently aims at exploring the option of using COTS FPGAs in satellite payloads. A framework is developed, allowing the SEU susceptibility of such a device to be studied. SEU testing is carried out in a software-simulated fault environment using a set of Java classes called JBits. A radiation detector module, to measure the radiation backdrop of the device, is also envisioned as part of the final design implementation.

KEYWORDS: Radiation, Geiger tube, Scintillator, Semiconductor Detector, Single Event Upset (SEU), Field Programmable Gate Array (FPGA), JTAG, JBits, XHWIF

Shadab Gopinath Ambat

02/29/2008

SINGLE EVENT UPSET DETECTION IN FIELD PROGRAMMABLE GATE ARRAYS

By

Shadab Gopinath Ambat

Dr. James E. Lumpp Jr.
Director of Thesis

Dr. Yu Ming Zhang
Director Graduate Studies

02/29/2008

RULES FOR THE USE OF THESIS

Unpublished theses submitted for the Master's degree and deposited in the University of Kentucky Library are as a rule open for inspection, but are to be used only with due regard to the rights of the authors. Bibliographical references may be noted, but quotations or summaries of parts may be published only with the usual scholarly acknowledgements.

Extensive copying or publication of the dissertation in whole or in part also requires the consent of the Dean of the Graduate School of the University of Kentucky.

A library that borrows this project for use by its patrons is expected to secure the signature of each user.

Name	Date

THESIS

Shadab Gopinath Ambat

The Graduate School
University of Kentucky
2008

SINGLE EVENT UPSET DETECTION IN FIELD PROGRAMMABLE GATE ARRAYS

THESIS

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in the College of Engineering
at the University of Kentucky

By

Shadab Gopinath Ambat

Lexington, KY

Director: Dr. James E. Lumpp Jr., Professor of Electrical Engineering

Lexington, KY

2008

MASTER'S THESIS RELEASE

I authorize the University of Kentucky
Libraries to reproduce this thesis in
whole or in part for purposes of research.

Signed: Shadab Gopinath Ambat

Date: 02/29/2008

To my parents...

Acknowledgements

First of all I would like to thank my advisor, Dr. James E. Lumpp, Jr., for his patience and guidance throughout this work and for allowing me to be a part of the IDEA Lab. The knowledge and experience I gained while working under him will be invaluable. I would like to thank Dr. William R. Dieter for being a part of my graduate committee and for teaching a very interesting Real-Time and Embedded Systems course. It acted as the necessary catalyst in my desire to work in the area. I am also thankful to Dr. Todd J. Hastings for serving in my committee and for teaching an excellent course on Solid-State Electronics. Although I did not get a chance to pursue the field further, the experience was certainly rewarding.

Thanks to Gordon Hollingworth and Miguel Silva for their previous work with JBits and also to Gonçalo Furtado for his prompt help. They provided some form of assistance in an otherwise support-less community.

I would like to thank Yehya, Samir, and Rama for the time and effort they have spent in checking the thesis for any mistakes and for the valuable feedback they provided. Last, but not least, I would like to thank my mother and my friends. Their everlasting support contributed in more ways than one and made this all possible.

Table of Contents

Acknowledgements .... **iv**

Table of Contents .... **v**

List of Figures .... **vii**

List of Tables .... **ix**

1 Introduction **1**
- 1.1 Background 1
  - 1.1.1 Field Programmable Gate Arrays (FPGAs) 2
  - 1.1.2 The Single Event Upset (SEU) Problem 4
- 1.2 Objective 7
- 1.3 Thesis Outline 8

2 System Overview **9**
- 2.1 Introduction 9
- 2.2 The CubeSat Standard 9
- 2.3 System Overview 10
  - 2.3.1 Detector Subsection 11
  - 2.3.2 SEU Subsection 11
  - 2.3.3 Detection Unit CPU 12
  - 2.3.4 Flight Computer 12

3 Radiation Detector Options **13**
- 3.1 Introduction 13
- 3.2 Space Radiation Environment 13
  - 3.2.1 The Van Allen Radiation Belts 13
  - 3.2.2 Cosmic Rays 17
  - 3.2.3 Solar Wind 18
  - 3.2.4 Solar Flares and Coronal Mass Ejections 20
- 3.3 Radiation Detector Types 21
  - 3.3.1 Geiger-Müller (GM) Tube 21
  - 3.3.2 Scintillators 24
  - 3.3.3 Semiconductor Detectors 28

4 Single Event Upsets and Their Effects on FPGAs **34**
- 4.1 Introduction 34
- 4.2 Single Event Effects 34
- 4.3 Other Radiation Effects 35
  - 4.3.1 Total Ionizing Dose (TID) Effects 35

4.3.2 Spacecraft Charging 36
4.4 SEU Mechanism 37
4.5 Factors Inducing SEUs 38
4.5.1 Trapped Particles 38
4.5.2 Galactic Cosmic Ray (GCR) Particles 39
4.5.3 Particles from Solar Flares 39
4.5.4 Atmospheric Neutrons 40
4.5.5 Radioactive Materials 40
4.6 Effects of SEUs on FPGAs 40
4.6.1 Upset in a Basic SRAM Cell 40
4.6.2 SEU Configuration Error Modes 42
4.7 SEU Mitigation Techniques 46
4.7.1 Technology-Based Techniques 46
4.7.2 Design-Based Techniques 47

5 Implementation of the XHWIF Interface 52
5.1 Introduction 52
5.2 JBits 52
5.3 Virtex-II Evaluation Board 54
5.4 The Joint Test Action Group (JTAG) Interface 56
5.5 XHWIF Implementation for the Virtex II Board 59
5.5.1 The JTAG Scan Chain 59
5.5.2 Host PC-Board Communication Interface 61
5.5.3 The Java Native Methods 62
5.5.4 The XHWIF Interface 68

6 Testing Methodology 69
6.1 Introduction 69
6.2 SEU Detection Logic 69
6.3 Test Procedure 72
6.3.1 Phase I: Fault-Free Operation 72
6.3.2 Phase II: Fault Injection 75
6.4 Results 79

7 Conclusion 81
7.1 Summary 81
7.2 System Implementation 82
7.3 Future Directions 83

Appendix A: Virtex II JTAG Registers and Supported Instructions 85

Appendix B: Virtex II Selected Configuration Specifics 88

Glossary 94

References 96

Vita 101

List of Figures

Figure 1: A Commercial Pancake-Type Geiger-Müller Probe 1
Figure 2: FPGA Block Diagram [31]........ 2
Figure 3: CLB Slice for the Virtex II [31] 3
Figure 4: The Virtex 5 FPGA 4
Figure 5: KySat1 10
Figure 6: System Block Diagram........ 11
Figure 7: The Van Allen Radiation Belts 14
Figure 8: The Explorer I Satellite Launching on a Redstone-Jupiter C Rocket [9]........ 14
Figure 9: The Solar Wind [10]........ 19
Figure 10: Solar Flare (Left) and a Coronal Mass Ejection (Right) [10]........ 20
Figure 11: Geiger-Müller Tube Circuit Diagram........ 22
Figure 12: Different Types of Geiger-Müller Tubes. Thin Wall (Hot Dog-Type) (Left), End-Window (Center) and Pancake Style (Right) [51] 23
Figure 13: Typical Scintillator Detector Arrangement 25
Figure 14: P-N Diode Detector Circuit........ 29
Figure 15: Si PIN Diodes [22] 30
Figure 16: RADFET Operation 31
Figure 17: A 4-Transistor RADFET Chip with Bonded Wires to read the Dose Measurements [25]........ 32
Figure 18: The South Atlantic Anomaly [29]........ 39
Figure 19: 6-Transistor SRAM Storage Cell 41
Figure 20: PIP Failure Modes........ 42
Figure 21: An Example of an LUT Error........ 43
Figure 22: A MUX Defect 44
Figure 23: A Simplified IOB Element [35] 45
Figure 24: An Error in an IOB Element [35]........ 45
Figure 25: TMR Block Diagram........ 47
Figure 26: Implementation of TMR for State-Dependent Logic [36] 48
Figure 27: An Example of Temporal Redundancy 49
Figure 28: A Temporal Sampling Latch [37] 49
Figure 29: Configuration Frames in the Virtex II [38] 51
Figure 30: JBits Execution........ 53
Figure 31: The Virtex-II Evaluation Board 54
Figure 32: Boundary-Scan Architecture 56
Figure 33: TAP Controller State Diagram [38] 58
Figure 34: Boundary-Scan Chain on the Virtex II Board........ 60
Figure 35: Parallel Cable III [43]........ 60
Figure 36: Parallel Cable III Schematic [43] 61
Figure 37: Flowchart for Fetching Device IDs........ 63
Figure 38: FPGA Configuration Flowchart........ 65
Figure 39: Flowchart showing a Readback Sequence 67

Figure 40: SEU Detection Logic.. 70
Figure 41: Screenshot of the Java Test Program 71
Figure 42: Files used for Bitstream Verification [38]... 74
Figure 43: A Detailed Look at the CLB Slice (Top Half) [31]... 76
Figure 44: Error File Structure.. 77
Figure 45: Proposed System Implementation.. 82
Figure 46: Packet Header Types.. 88
Figure 47: The STAT Register 92

List of Tables

Table 1: Comparison between Radiation-Hardened and Commercial FPGAs........................ 6
Table 2: Comparison between Commonly-Used Scintillator Materials................................ 27
Table 3: Board Configuration Mode Settings.. 55
Table 4: Fault-Injection Test Results.. 78
Table 5: Virtex II JTAG Instructions... 85
Table 6: JTAG Registers for the Virtex II 86
Table 7: Virtex II Configuration Register Set... 89
Table 8: CMD Register Codes 91
Table 9: Description of STAT Register Bits... 92

Chapter 1

Introduction

1.1 Background

Advancement in space technology has prompted a significant effort to be dedicated towards understanding and characterizing the effects of ionizing radiation. Apart from the hazards such radiation poses to humans, the errors that it induces in electronic devices and circuits can lead to abnormalities in critical spacecraft modules. Such effects have been observed in the past in endeavors like Voyager-1 and the Hubble Space Telescope [1] [55] and have been the motivation for several projects aimed at preventing similar occurrences in future.

Figure 1: A Commercial Pancake-Type Geiger-Müller Probe

Radiation in space can arise from sources such as cosmic rays, the Van Allen radiation belts, solar phenomena etc. and can be measured by means of sensors like Geiger-Müller (GM) tubes or semiconductor detectors. These devices can detect different types of radiation and can also measure attributes such as the dose amount or particle energy levels. Figure 1 shows a commercial radiation detector from United Nuclear™ that is

capable of measuring alpha, beta and gamma radiation. The sensing element consists of a pancake-type GM detector.

1.1.1 Field Programmable Gate Arrays (FPGAs)

FPGAs are semiconductor devices that can be user-programmed with a logic function. The function can range from common structures such as arithmetic logic units or microprocessors to custom application-specific logic. Their inherent re-programmability feature has been fully exploited for prototyping purposes. The FPGA architecture, that makes all this possible, consists of two key functional blocks – the configurable logic block (CLB) and a switch matrix that forms the interconnections between them. Some devices can also contain additional blocks such as an Input Output Block (IOB) that acts as an interface between the device and the package pins. The block diagram of a typical FPGA (Virtex™ II from Xilinx™) is shown in Figure 2 [31].

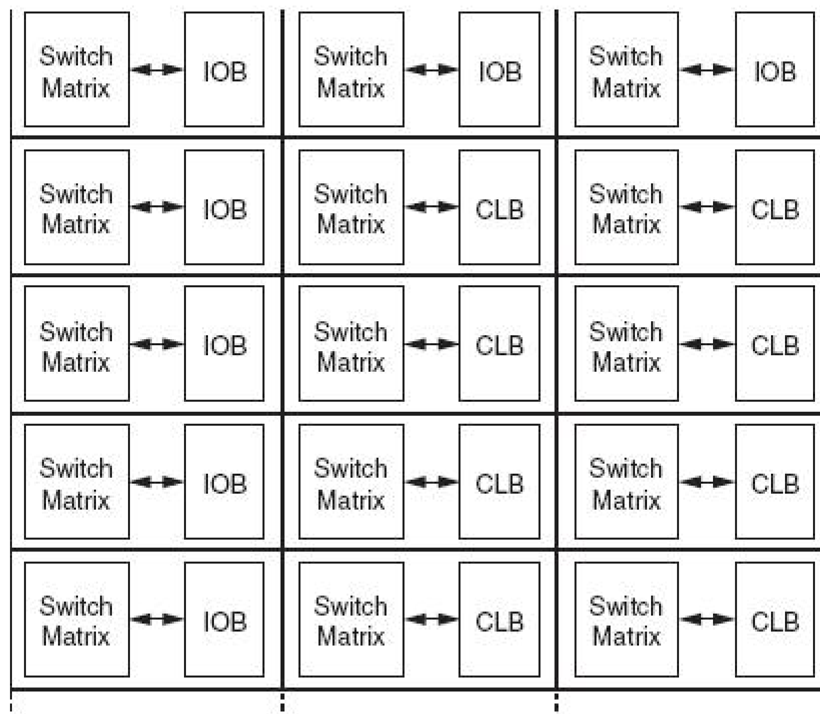


Figure 2: FPGA Block Diagram [31]

The CLB is the heart of the FPGA, and it normally consists of look-up tables (LUTs) and other low-level logic elements such as logic gates, storage elements, multiplexers

(MUXs) etc. The CLB in case of a Virtex II FPGA for instance, consists of 2 tri-state buffers and 4 similar slices with each slice containing a pair each of 4-input function generators, carry logic, multiplexers, storage elements and a few logic gates, shown in Figure 3 [31]. The function generators can be configured as either one of a 4-input LUT, a 16-bit distributed SelectRAM, or a 16-bit shift register. CLBs are arranged in the form of an array with the programmable interconnects (switch matrix) providing the necessary connections between different blocks based on the implemented function. Other than these, devices can also contain blocks to facilitate design development such as random access memory (RAM) structures, multipliers, clock managers or sometimes even high-end structures such as microprocessors.

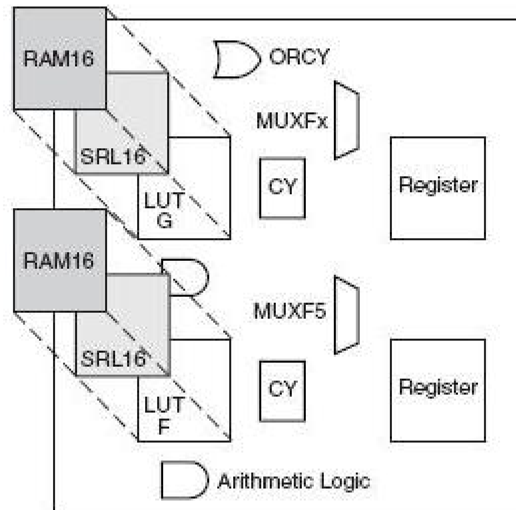


Figure 3: CLB Slice for the Virtex II [31]

The flow one usually follows in order to program an FPGA with a desired logic design can be divided into three steps. The design is first created using a hardware description language such as Verilog or VHDL. This is then utilized in generating a low-level device specific bitstream file. The final step is to download the bitstream onto the configuration memory of the FPGA, which individually defines the behavior of the CLBs and corresponding interconnects needed to implement the design. The configuration memory can be based on static RAM (SRAM), flash or antifuse structures. Other than the

configuration memory, the device also consists of user memory which is used to store design-related data such as register states.

FPGAs have gained popularity in recent years particularly in aerospace applications. One of the major reasons being the prospect of performing post-launch design optimizations or changes in spacecraft objectives. Present line of commercial FPGAs are capable of integrating powerful embedded processors and several common intellectual property cores that provide a complete system-on-chip solution. FPGAs that contain dedicated multiplier blocks are particularly suited as co-processors for computation-intensive applications such as digital signal processing. A good example is the Virtex 5 which is the latest in the Virtex series of FPGAs. The device shown in Figure 4 is built on a 65 nm process and includes 288 dedicated DSP blocks, each block consisting of a 25 bit x 18 bit multiplier and a 48 bit adder/subtractor/accumulator. Other features include 4 embedded Ethernet MAC blocks capable of implementing 1000 Base-X (Gigabit Ethernet) implementation, and a dedicated block that provides PCI-Express functionality [3].

Figure 4: The Virtex 5 FPGA

1.1.2 The Single Event Upset (SEU) Problem

A particularly prominent effect of radiation pertaining to electronics is the SEU, identified as a change in state occurring when a high-energy particle collides against a sensitive node of a micro-electronic device. These changes in voltage levels can in-turn lead to potentially destructive events such as latch-ups. Recent advancements in the

semiconductor industry have led to the fabrication of highly scaled devices. These devices exhibit an increased sensitivity to SEUs due to a reduced feature size and a proportional increase in device density.

In the case of programmable logic devices such as FPGAs, SEU effects can be much more severe. Since FPGAs utilize a configuration memory array to define the logic function, an SEU occurring in a single bit in the array can lead to an unexpected alteration of the original design. The SRAM structure in particular, that commonly forms the configuration memory of commercial-off-the-shelf (COTS) FPGAs, is known to be highly susceptible to SEUs [2]. Hence certain accommodations need to be made in order to use these devices for space applications.The industry solution to SEUs is radiation-hardening. These enhanced devices are basically derived from similar commercial versions with the underlying difference usually lying in the manufacturing technology or in the implementation of redundant logic to counter SEU events. The downsides of using such radiation-hardened devices are:

- These devices consume more power
- Are more expensive
- Are typically slower than the commercial version
- Have a reduced effective design area

Moreover, while hardening can reduce the occurrence of SEUs, the device is not completely immune to them. This argument particularly holds true for rad-hardened versions based on the SRAM architecture of COTS technologies [5]. To elucidate the tradeoffs involved with hardening, a comparison is provided in Table 1 between a rad-hardened Virtex II QPro FPGA (XQR2V1000) and its commercial counterpart, the Virtex II FPGA (XC2V1000). The first significant point to note is that the QPro was released more than 3 years later with no significant design changes. To provide a better basis for assessment the Virtex 4 device, released in the same year as the QPro, is also included in the comparison.

Table 1: Comparison between Radiation-Hardened and Commercial FPGAs

	Virtex II QPro	**Virtex II**	**Virtex 4**
Device and Package Type	XQR2V1000-4 BG575[1]	XC2V1000-6 BG575	XC4VLX25-12 FF668[2]
Release Date	May, 2004	Jan, 2001 [4]	Sep, 2004
Process Technology	150 nm	150 nm	90 nm
Bitstream size	3,753,432 bits	4,082,592 bits	7,819,520 bits
Core Power Supply Voltage (V_{CCINT})	1.5 V	1.5 V	1.2 V
Quiescent V_{CCINT} Supply Current (I_{CCINTQ})	100 mA (Typical) 500 mA (Max)	12 mA (Typical) 250 mA (Max)	77 mA (Typical) N/A (Max)
DCM[3] Max Input Clock Frequency (using DLL[4])	360 MHz	450 MHz	500 MHz
Available I/Os	328	328	448
Size	31 mm x 31 mm	31 mm x 31 mm	27 mm x 27 mm
Price[5]	$4,091.00	$446.00	$462.50

Notes:

1. BG – Standard Ball Grid Array (BGA)
2. FF – Flip-Chip Fine-Pitch BGA
3. DCM – Digital Clock Manager
4. DLL – Delay-Locked Loop
5. Prices from NuHorizons™ and Avnet™

Though built to be structurally similar to the Virtex II, the QPro still shows some discrepancies, such as a smaller bitstream size and higher power consumption. A higher speed grade was used for the Virtex II (-6) to emphasize the fact that the QPro has only one speed option (-4). Also notice how the QPro compares to the Virtex 4 that has more than twice the number of configuration bits translating to more logic resources. It should be mentioned here that the XC4VLX25 is one of the lower end versions of the Virtex 4. Finally the rad-hardened device is nearly 10 times more expensive making it unsuitable

for low-budget projects. Thus several applications might find it more practical to use COTS FPGAs.

The application of this work is in determining the feasibility of using these devices in spacecraft systems. Although the associated benefits of COTS devices make them attractive options, their higher vulnerability to SEUs makes it necessary to incorporate fault recovery techniques. The mechanism employed here makes use of the FPGA readback/reconfiguration features to achieve this. Also, in our case these functions will be implemented by a microcontroller hence the unit that manages the process needs to be deployable on an embedded platform. For this purpose a set of Java classes known as the JBits Application Programming Interface (API) was selected. JBits provides the capability of designing and/or modifying circuits for Xilinx Virtex devices. Apart from inheriting the advantages of Java, a useful feature of JBits is that it can be used to examine and even configure individual FPGA CLBs. The main requirement to run JBits is a Java Virtual Machine (JVM). Several implementations of the JVM have been specifically customized for microcontrollers and could therefore be used to execute it. Santiago Leon from Virginia Polytechnic Institute and State University was able to successfully run JBits on a LEON RISC processor through the Waba JVM [7].

1.2 Objective

The motivation behind this work is to incorporate a COTS FPGA in a satellite payload to determine its behavior in high-radiation space environments, i.e. its exhibited SEU rate. A radiation detector was also included in the design so as to be able to quantify the SEU counts based on the surrounding radiation levels.

The main aim of this thesis is to develop the framework for the major sections of the payload and thereby aid in the final implementation. The first section namely the radiation detector module requires the selection of a suitable sensor. GM tubes as well as other popular sensor options will be examined, describing their operating mechanisms and typical characteristics. The second section is the SEU module that will monitor the

FPGA for SEU instances and consequently remove them by applying mitigation techniques. The JBits package chosen herein plays two major roles in this work. The first, and what originally prompted its selection, is in simulating the SEUs. Since the cost for using a cyclotron facility is particularly high, the faults are injected by directly modifying the bitstream and performing a run-time reconfiguration. The second function is in identifying and correcting any occurring SEUs in the bitstream. This is done by performing a readback whenever the logic on the FPGA detects an SEU, recording the bits that were affected, and reconfiguring the device with the correct bitstream. A test environment was developed for this intent that made use of the Xilinx Virtex-II Evaluation Board from Avnet™, populated with a Virtex-II FPGA. Details on the setup will be given in a later chapter.

1.3 Thesis Outline

Chapter 2 of this thesis introduces the CubeSat standard along with the Kentucky Satellite (KySat) project. An overview of the proposed system for the final design is also given in a later section. Chapter 3 describes the major radiation sources that are likely to affect the payload and examines several commonly used radiation detectors. Chapter 4 explains SEU concepts and various factors that can generate them. The effects that SEUs tend to have on FPGAs are then presented along with a list of applicable mitigation strategies. JBits makes use of the Xilinx HardWare InterFace (XHWIF) to communicate with the board. Chapter 5 is thus dedicated to explaining the process used in creating this interface and the components involved therein. The testing methodology that was followed to generate and consequently detect faults in the FPGA bitstream is then provided in Chapter 6. Finally, Chapter 7 summarizes the thesis along with suggestions for future improvements.

Chapter 2

System Overview

2.1 Introduction

The final payload implementation is expected to be carried out for a CubeSat class satellite. This chapter describes these satellites along with their main features. This is followed by a description of the overall system.

2.2 The CubeSat Standard

The CubeSat standard was designed at Stanford University and California Polytechnic State University (Cal Poly), and defines electrical, structural, operational, and testing specifications for a 1 kg, 10 cm cube-shaped satellite dubbed as a CubeSat [40]. Alternate designs utilizing double (2 kg) or even triple (3 kg) configurations can also be used. The launch is managed by Cal Poly by means of a standardized deployment system known as the Poly Picosatellite Orbital Deployer (P-POD). The P-POD can carry up to three single CubeSats and acts as the interface between the launch vehicle and the CubeSats. Its main purpose is to ease integration into any launch vehicle. This allows the CubeSats to be deployed as secondary or tertiary payloads at a lower cost.

CubeSats are typically launched in a Low-Earth Orbit and have become a popular low-cost option particularly for university missions. The community consists of over 80 universities and corporations world-wide with several companies providing off-the-shelf components to facilitate CubeSat designs.

A possible candidate for carrying the payload described in this work is the Kentucky Satellite (KySat). KySat is a state-wide effort by universities in Kentucky, to provide rapid access to space for small payloads. KySat1 is planned as the first of many launches

expected to have more ambitious payloads and spacecraft bus capabilities. The payload on KySat1 consists of a VGA camera and an experimental S-Band radio for high-bandwidth communication. It is expected to be launched in 2008 and while further details on its orbit are not available at present, certain passage over Kentucky rules out orbits with inclinations less than 40 degrees. The active lifespan of the satellite is expected to be 18 months, though it will remain in orbit for approx. 10-15 years [40].

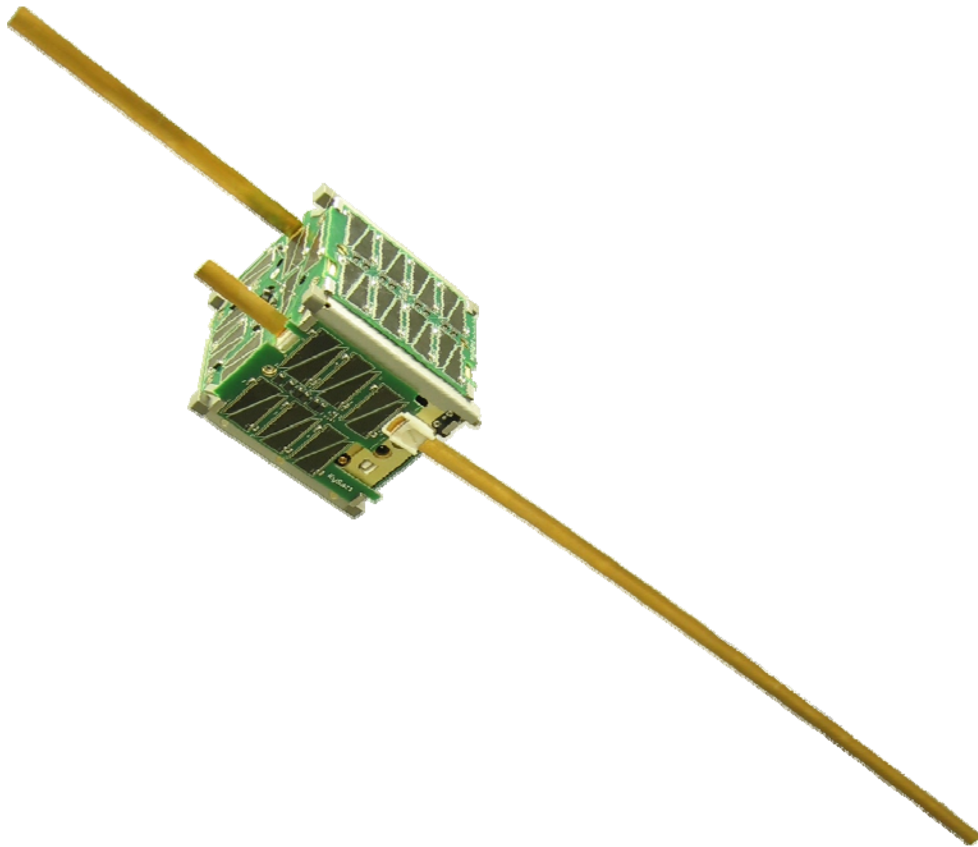


Figure 5: KySat1

2.3 System Overview

A basic block diagram for the envisioned system is given in Figure 6. A discussion of the individual sections along with the expected dataflow between them ensues.

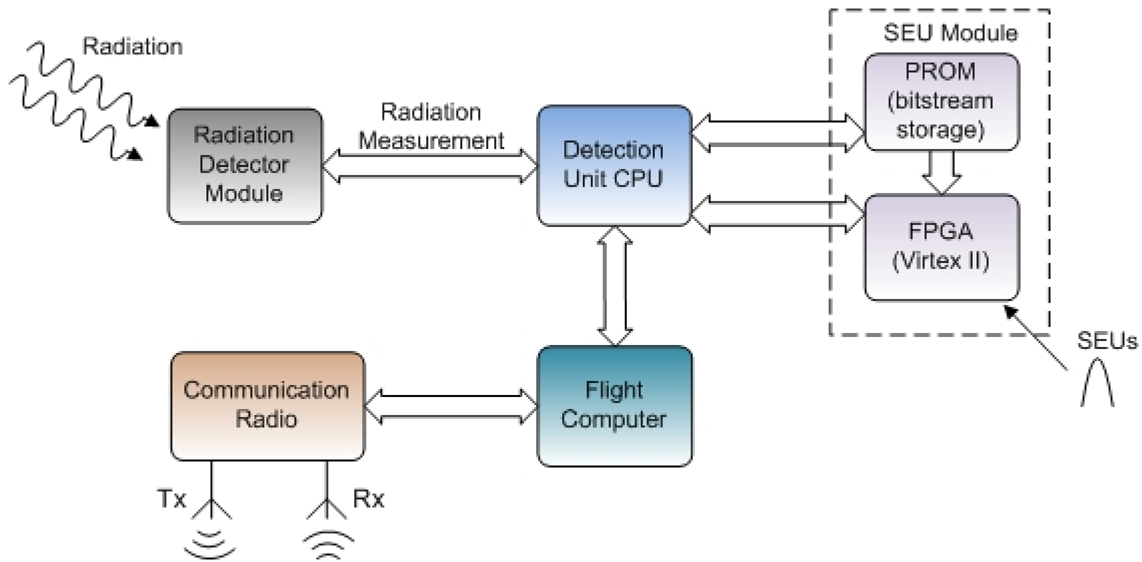


Figure 6: System Block Diagram

2.3.1 Detector Subsection

This subsection will consist of one of the radiation detectors that will be discussed later on. Each of the sensors has their own strengths and the selection will need to be based on optimality towards fulfillment of mission goals. For instance, GM tubes provide a simple and robust solution, in addition to being cost-effective, while solid-state detectors such as PIN diodes offer a much better resolution and are also able to provide a measure of the incident particle energy.

2.3.2 SEU Subsection

The device under test (DUT) for the SEU module will consist of a Virtex II FPGA. A Virtex II Pro was initially considered but it contained several enhanced features that were deemed unnecessary for our current requirements. Another reason that had prompted this change was that JBits has not yet been implemented for this series. The DUT will be programmed with a logic function to detect SEU events. The simplest approach consists of two identical logic slices that are fed with the same test pattern. Their outputs are then constantly compared for mismatches occurring due to SEUs.

An additional PROM will also be required to store the bitstreams needed for configuration and SEU detection/correction. FPGA configuration can either be

completely managed by the microcontroller, or it can alternatively be configured directly from the PROM. The second method can be faster especially if done via a parallel SelectMAP interface, and can be employed during power-up or in cases when power cycling is required to correct the errors.

2.3.3 Detection Unit CPU

A principal requirement for the microcontroller selected as the CPU for the SEU and Detector Subsections is that it should have sufficient memory to be able to run JBits. This includes memory for the JVM implementation and the underlying real-time operating system. Part of the code can also reside in the PROM. The primary functions of the microcontroller will be to generate a measure of the radiation based on the sensor output, and to detect and count the number of SEU events, optionally recording the bits that were affected. These can then be transmitted to the ground-station as data packets along with a timestamp.

2.3.4 Flight Computer

This block will be the CPU for the main payload of the satellite and will control most of the system operations. These will for example include tasks such as management of the power system, communication radio, etc. For KySat1, the power system consists of a solar cell array as the primary power source along with a backup battery supply. Communication with the ground-station is established through a UHF and VHF antenna for transmitting and receiving respectively. More information on the implementation can be found in [40]. The data on the radio com-link from our perspective will typically consist of:

- Commands from the ground-station.
- Periodic or on-demand updates on the SEU count.
- Periodic or on-demand transmission of the measured particle count and/or absorbed radiation dose.
- Periodic transmission of system status information.

Chapter 3

Radiation Detector Options

3.1 Introduction

This chapter will study the most common radiation detectors describing their concept of operation and features. However, before proceeding it is essential to be aware of the radiation setting that the payload is to be deployed in.

3.2 Space Radiation Environment

The radiation environment in space is affected by several elements and the type of orbit, altitude, and inclination of the payload decide which of these play an important role during its active time in space. Apart from their function in detector dose measurements, these sources are also important from an SEU perspective as will be discussed in the following chapter. The payload in our case is expected to be launched at a Low Earth Orbit (LEO), defined by NASA as an orbit that extends from 80 km – 2,000 km above the earth's surface [46]. Other common orbits include:

- **Medium Earth Orbit (MEO)**: Orbits that lie between 2,000 km to 35,786 km.
- **Geosynchronous Orbit (GEO)**: Orbits at an altitude of 35,786 km having a period equal to the earth's rotational period.
- **Highly Elliptical Orbit (HEO)**: Orbits with a low perigee of approx. 1,000 km and an apogee of 35,786 km. Their inclination is typically between 50 and 70 degrees.

3.2.1 The Van Allen Radiation Belts

The Van Allen Belts are composed of two toroidal bands of energetic particles trapped due to the earth's magnetosphere, shown in Figure 7. The inner belt extends from 1,000

km – 10,000 km and consists mainly of high-energy protons while the outer belt, made predominantly from trapped electrons believed to have been injected from the solar wind, lies at an altitude of about 15,000 km – 30, 000 km [8]. The gap between the two belts, designated as the 'safe zone' or the 'Van Allen Belt slot', has a much lower particle count and is particularly significant to satellites having MEOs. The zone is created by particle interactions with radio waves generated from lightning. This process reduces the particle energy, eventually resulting in their harmlessly dissipating in the earth's atmosphere.

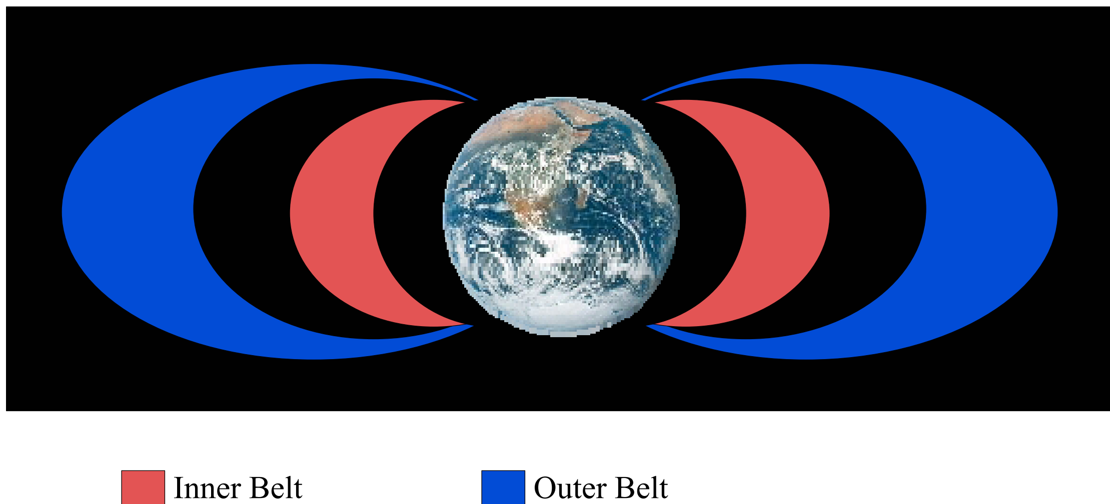


Figure 7: The Van Allen Radiation Belts

Figure 8: The Explorer I Satellite Launching on a Redstone-Jupiter C Rocket [9]

The inner belt was first discovered by Explorer I and later confirmed by Explorer III. Both missions were launched by NASA in 1958 under Dr. James Van Allen. The outer belt was later discovered in the same year by Pioneer III [47]. All three missions used a Geiger-Müller (GM) tube as the radiation detector.

Though the earth's atmosphere restricts the belts from directly affecting surface life, their effects are still visible. The Aurorae Borealis (Northern Lights) for instance are caused due to the trapped particles (electrons) entering the earth’s atmosphere through the geomagnetic tail, which is the region of the belts closest to the earth. The electrons impart energy through collisions to atmospheric elements that in-turn release the energy as light. Emission from oxygen for example is green, while nitrogen emits a reddish glow.

Radioactive neutrons generated by collision of cosmic rays with the atmosphere, decay and produce protons as by-products forming the inner belt. The AE-8/AP-8 Trapped Particle Flux Maps from NASA [11] estimate the particle energies in the belts to be 0.04 MeV – 7 MeV for electrons and 0.1 MeV – 400 MeV for protons. These flux maps have been the standard used by spacecraft designers for several decades now. Known limitations exist however and thus a new AE-9/AP-9 model is currently being developed by NASA and several other partners. The new model will include ions and is expected to cover particles with a broader energy range [44].

The particle fluxes can fluctuate due to effects from solar events such as flares or from man-made causes (nuclear explosions). These can at times even introduce artificial radiation belts. A classic example of one such incident and the damage caused therein is Starfish Prime, a nuclear test conducted by the United States on July 9, 1962. The test, occurring at an altitude of 400 km, inserted an artificial belt of high-energy electrons (up to 7 MeV) into the inner Van Allen belt. Effects of the belt were observable for five years, nearly eight years in some areas [16]. Several satellites failed as a result of the unexpected increase in radiation [12]:

- **Ariel I (US/UK)**

Ariel I, launched in April 1962, was a satellite designed by NASA to carry six British payloads to study the relationship between the ionosphere and solar radiation. Four days after the explosion its operation became irregular due to the accelerated degradation of its solar cells caused by the electrons from the belt. Other effects included overloading of the cosmic ray particle counter circuitry and a consequent weakening of the detector [13].

- **Transit 4B (US)**

Transit 4B was a US Navy navigation satellite launched in November 1961. It stopped transmitting 25 days following the test due to a sharp decrease in solar cell output.

- **TRAAC (US)**

The Transit Research and Attitude Control (TRAAC) satellite was launched on the same launch vehicle as the Transit 4B to test gravity-gradient stabilization in Transit satellites. It also shared the same fate as the Transit 4B, failing to transmit 38 days after the explosion due to solar cell degradation.

- **Injun I (US)**

Built to study the natural and artificial radiation belts, auroras, and other geophysical phenomena, Injun I was launched in June 1961. Its active transmission ended in March 1963, longer than the other 3 satellites, owing to a power supply designed to withstand higher degradation [14].

- **Cosmos V (Russia)**

It was launched in May 1962, to study the upper layers of the earth's atmosphere, and was used to collect data from the artificial belt for a period of 4 months. Only indirect information is provided on the power sources of the satellite.

- **Telstar I (US)**

Telstar I was the first active, direct relay communications satellite, and was launched in July 1962 on the day following the Starfish blast. It was designed by Bell Laboratories and funded by AT&T. An increase in radiation intensity caused from

Starfish and a Russian nuclear test in October 1962 led to the failure of some transistors in the command system and it stopped responding in November 1962. Although a workaround managed to reactivate the satellite, the transmitter could only handle intermittent transmissions before finally failing on February 21, 1963 [15].

There are numerous instances of satellite failures induced by the radiation belts and those mentioned above only demonstrated the ones that were specifically due to Starfish. An upside of the Van Allen belts is that they provide some level of protection against solar winds and cosmic rays by limiting the number of particle collisions instigated by them in the atmosphere.

3.2.2 Cosmic Rays

Cosmic rays consist of energetic particles that originate from outside the earth's atmosphere. They were discovered in 1912 by Victor Hess. He measured the background ionization rate on a balloon and found that it increased with altitude, concluding that this was due to radiation from sources outside the atmosphere, later named 'cosmic rays'. Cosmic rays comprise:

- Galactic Cosmic Rays (GCRs): They consist of particles originating from outside the solar system.
- Anomalous Cosmic Rays (ACRs): These are particles originating from within the interstellar space in the heliosheath.
- Solar Energetic Particles: They are generated from solar events such as solar flares.

This section will only deal with GCRs, since ACR particles have comparatively lower energies and speeds. Solar energetic particles will be studied in the following sections.

GCRs are composed of approximately 90% (hydrogen) protons, 9% alpha particles (helium) and 1% ionized nuclei of various elements. About 1% of cosmic rays particles consist of electrons. Ions of elements having an atomic number (Z) $1 \leq Z \leq 92$ are known

to be present in these rays [17]. Still, most of the ions are from lighter elements such as hydrogen, helium, and to a lesser degree carbon and oxygen. Particles from heavier elements (Z>25) such as iron are comparatively rare but nonetheless present. Despite their lower fluxes, when compared to trapped particles, GCRs still possess high penetrating properties due to their energy levels and are hence hostile towards electronics. Particles can have energies ranging in hundreds of GeV and some have energies well over that range. In fact the highest energy that was ever recorded for a single cosmic ray particle was in 1991, by the Fly's Eye cosmic ray detector of the University of Utah, and was approx. $3x10^{20}$ eV [18]. Particles with such high energies are believed to originate from outside the Milky Way as opposed to the majority of GCRs.

GCRs are energized while they are traveling through interstellar matter. On top of getting ionized during this passage, the particles are also accelerated several times, probably by supernova remnants to nearly the velocity of light [48]. Constant deflections from magnetic fields scatter the particles evenly in all directions. This effectively makes it impossible to determine the direction of their origin. Particles that are not fully ionized before they reach the earth's magnetosphere have a higher penetrating power. The earth's magnetic field can provide some degree of protection against GCRs to spacecrafts depending on their inclination and altitude. The poles, however, are vulnerable on account of the shape of the field lines. Like the trapped radiation belts, GCRs are also affected by solar cycles with the particle level being lowest during solar maximum, and reaching climax during solar minimum.

3.2.3 Solar Wind

The solar wind is the plasma of charged particles, namely electrons, protons and heavy ions that originate from the Sun's corona. Its existence was first hinted at by the fact that comet tails always pointed away from the Sun irrespective of the comet's direction of motion. Ludwig Biermann from Germany suggested that this was due to a stream of particles emitted by the Sun or 'solar corpuscular radiation' [49]. In 1958, Eugene Parker of the University of Chicago while studying the coronal processes coined the term 'solar wind' for this flow [50]. The extremely hot temperature of the Sun continuously heats the

particles in the corona up to a point where they are capable of escaping from the Sun's gravitational pull. The high temperature results in a regulated plasma stream that travels radially outward along the Sun's magnetic field lines. The Sun's rotation keeps the flow of the wind continuous. Figure 9 shows the motion of solar wind through interplanetary space.

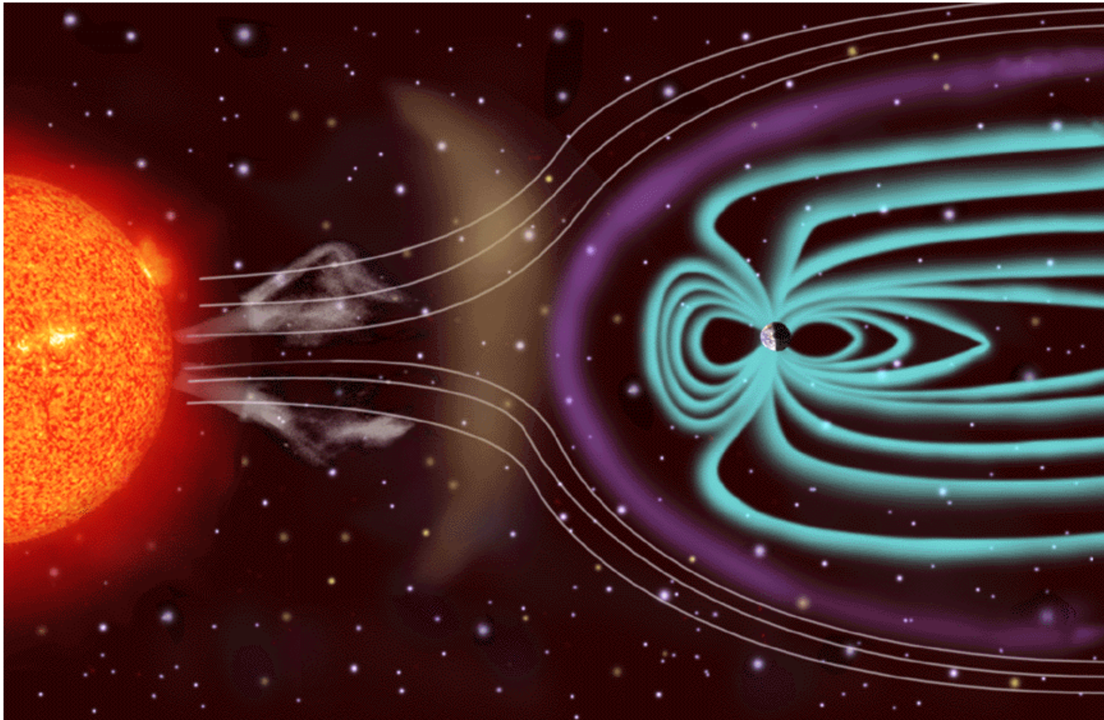


Figure 9: The Solar Wind [10]

The composition of the solar wind resembles the element distribution of the Sun with approx. 95% protons and the remaining 5% formed by helium and lesser doses of oxygen and other elements. An equal number of electrons, also present, balance out the positive charge. The particles are discharged by the Sun in all directions at energies of approx. 0.5 – 2 KeV/nucleon. They have extremely high speeds ranging from 300 km/s over streamers to 900 km/s over coronal holes (dark regions), with an average of about 400 km/s (nearly 10^6 mph) [10]. Solar events such as coronal mass ejections and solar flares cause disturbances in the wind and can result in an increase in charged particle density and energy levels through geomagnetic storms. Apart from interfering with radio signals

and an increased probability of satellite charging, an increased drag from such storms can result in LEO satellites prematurely re-entering the atmosphere.

3.2.4 Solar Flares and Coronal Mass Ejections

Solar flares are tremendous explosions occurring on the surface of the Sun. They are caused due to the abrupt release of energy in magnetically active regions around sunspots. The energy discharge is accompanied by emission of accelerated protons, electrons and heavy ion particles. The ions accelerated in this manner become the Solar Energetic Particles mentioned earlier. The frequency of flare events coincides with the solar cycle. The solar cycle is a major factor influencing space weather and has a period of approx. 11 years, divided into 7 years of solar maximum and 4 years of solar minimum. Increase in sunspots during solar maximum consequently result in a higher number of flare events than during solar minimum. Prominent flares can generate magnetic storms on Earth as discussed previously.

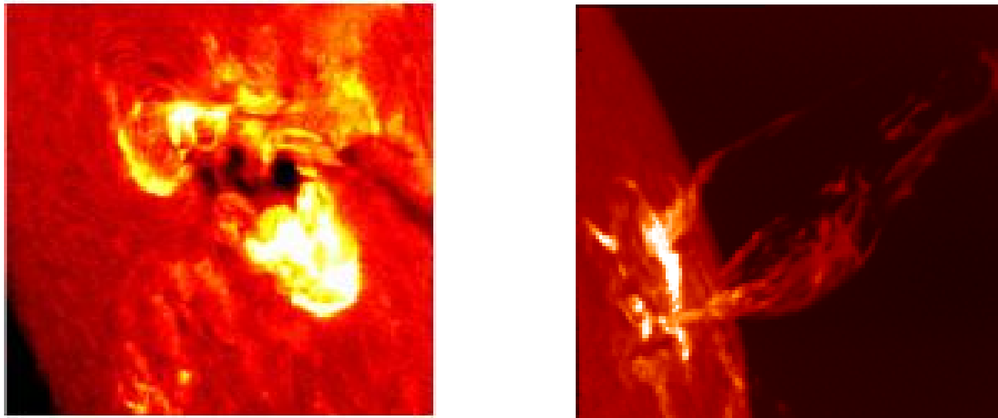


Figure 10: Solar Flare (Left) and a Coronal Mass Ejection (Right) [10]

Another solar phenomenon that can generate magnetic storms and energetic particles is a Coronal Mass Ejection (CME). CMEs are huge bubbles of gas bound with intense magnetic field lines that are ejected from the Sun. The shock waves are capable of accelerating the particles to millions of km/hr. CMEs can disrupt the Earth's magnetosphere and are believed to be responsible for major storms on Earth. They are usually associated with flare events but are known to occur even in their absence.

3.3 Radiation Detector Types

Long-term exposure to radiation can be detrimental to a device. Monitoring the radiation using radiation detectors can therefore give an indication as to the expected device lifespan. The following units can be used to express the measure of radiation:

- **Roentgen (R)**: One Roentgen is defined as the amount of ionization required to produce 1 electrostatic unit of charge in 1 cubic cm of air at standard temperature and pressure.
- **Radiation absorbed dose (rad)**: Rad is a measure of the dose absorbed by a material. 1 rad is equivalent to the dose required for 0.01 Joule of energy to be absorbed per kg of matter. (The SI unit for this term is Gray (Gy). 1 Gy = 100 rads)

Several types of radiation sensors are utilized in spacecraft designs. Some of them are more suited at measuring certain types of radiation than others. Commercial detectors are mainly utilized to detect radiation from 3 types of sources:

1. Alpha Particles: They are positive particles that are emitted during alpha decay and consist of 2 protons and 2 neutrons (helium nuclei).
2. Beta Particles: They are basically high energy electrons or positrons usually produced from beta decay.
3. Gamma Rays: These are photons having the highest energy and frequency (typically 10^{20} Hz and above) in the electromagnetic spectrum.

Some sensors can also be used to detect non-ionizing particles such as neutrons. This section discusses the different sensors that are used in radiation measurement, along with their innate strengths and weaknesses.

3.3.1 Geiger-Müller (GM) Tube

The GM tube was used in the Explorer missions to measure the radiation in the Van Allen belts and is still widely used today. It was invented in 1908 by Hans Geiger as a

device to detect alpha particles. In 1928, he improved the model with Walther Müller, extending the detection to all types of ionizing radiation. The GM tube is a kind of a gas-filled radiation detector. Other detectors falling under this category are ionization chambers and proportional counters. The pulses produced by the tube are usually counted by a microcontroller or a CMOS counter to quantify the surrounding radiation environment. Figure 11 shows a typical GM circuit.

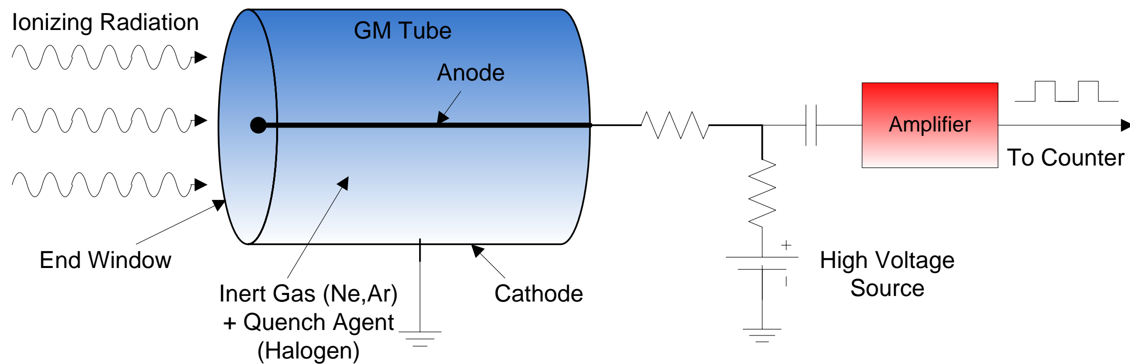


Figure 11: Geiger-Müller Tube Circuit Diagram

The outer surface of the tube is made of a conducting material like stainless steel that acts as the cathode. The anode consists of a thin metal wire lying along the central axis of the tube. The tube is filled with an inert gas such as neon or argon [20]. The tube shown here is an end-window type with the window usually being manufactured from materials such as mica, mylar, glass, or thin metals. Mica and glass are the most common types, with mica being more sensitive and hence providing alpha particle detection but also being more delicate. When radiation passes through the window, it ionizes the gas inside the tube causing pairs of electrons and positive ions to be generated. The circuit is completed when the electrons travel towards the anode and the ions towards the cathode. Since a detectable signal is required, the anode is kept at a high potential of typically 500 – 900 V. This provides sufficient acceleration to the electrons to generate more electron-ion pairs through collisions, resulting in an avalanche effect and thereby producing a large pulse at the output. To prevent a continuous discharge a process called *quenching* is employed. This involves either providing extra circuitry, or filling up the tube with a quenching gas such as alcohol vapor or a halogen. The second method is fairly effective

and is generally preferred. Also, halogen gas is normally used since it has a longer lifetime than alcohol, which gradually gets broken down due to the discharge process. Quenching correctly associates each pulse with a single particle. Depending on the pulse amplitude, an amplifier might be necessary before feeding the pulses to a microprocessor or other counting device. Current limiting resistors are usually needed when connecting the high-voltage source; moreover a small capacitor blocks the output from the high-voltage source while transmitting the pulses. Additional components might include a collimator that is used to prevent the tube from saturating in intense radiation regions by limiting its field of vision [6]. Though being relatively easy to use, a known downside that these detectors have is that they cannot discriminate between particles of different energies. A fixed-amplitude output pulse is generated as long as the particle energy is above the threshold for the tube. This can be countered by using highly sensitive detectors such as scintillators that are capable of producing pulses of different amplitudes based on incident particle energy.

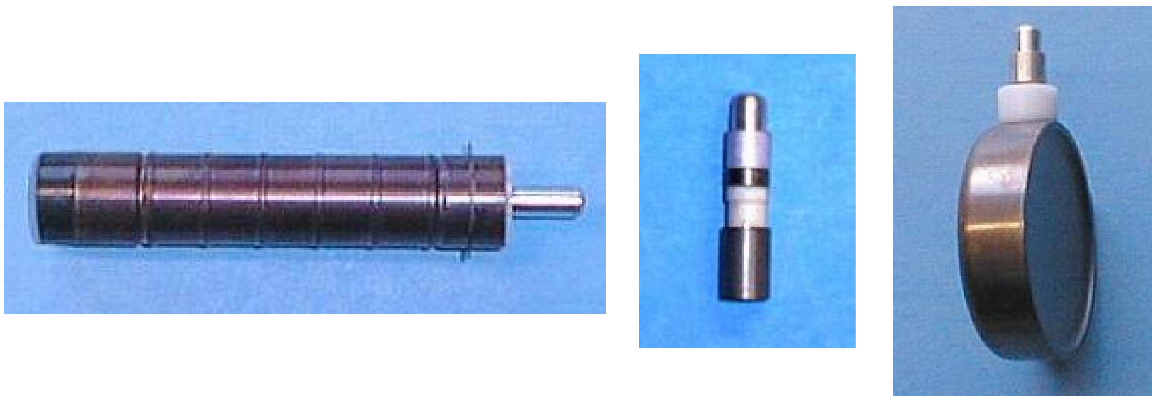


Figure 12: Different Types of Geiger-Müller Tubes. Thin Wall (Hot Dog-Type) (Left), End-Window (Center) and Pancake Style (Right) [51]

GM tubes are robust, cost-effective, and are frequently used for general-purpose radiation detection. They can even be modified to detect neutrons by filling the tube with boron trifluoride gas or providing an inner lining of boron. The interaction with boron nuclei result in alpha particle generation which can consequently be detected. GM tubes can be fairly small and light-weight and can come in different configurations. Figure 12 [51],

shows 3 kinds of GM tubes manufactured by LND, INC™ and arranged from left to right by increasing order of sensitivity.

- **Thin Wall (Hot Dog-Type)**: These are mainly used to detect beta and gamma radiation. They are tough and inexpensive but usually have a lower sensitivity and are unable to detect alpha particles.
- **End-Window**: They are more sensitive than the hot dog type, and can detect alpha radiation due to the addition of the mica/glass window. They can also come in reasonably small sizes. However, they are slightly more expensive and the window is rather delicate.
- **Pancake Style**: The large mica window gives them a much better sensitivity than the other two types particularly to low energy gamma photons. However they are bulkier and can be quite heavy (the model shown in the Figure is more than 20 times heavier than the end-window type). Another shortcoming is that since the window occupies nearly one entire side of the tube, they can be very fragile.

Overall the end-window type offers a good balance in terms of cost, sensitivity and size. The model shown in the Figure is also the same one used in the MEROPE project [6]. It is 35 mm long, with an 8.7 mm diameter and weighs only 6 grams. MEROPE was a CubeSat developed at Montana State University that aimed at reproducing the Explorer I mission, i.e. measure the radiation in the Van Allen belts using a Geiger tube. It was set for launch on July 26, 2006 but unfortunately the Dnepr Soviet missile it was riding on crashed shortly after launch. The efficacy of the design in measuring the radiation in the belts could therefore not be determined. However, GM tubes have been successfully used in numerous missions and as such are an especially viable option for our model.

3.3.2 Scintillators

Scintillation is the term used for a flash of light produced in a phosphor when it absorbs a photon or an ionizing particle. This is the basic principle on which scintillators work. When a charged particle or energetic photon enters the scintillation material, it transfers

its kinetic energy to the medium by exciting atomic electrons to higher energy levels. This energy is released as light in the visible or ultra-violet spectrum by the electron when it returns to a lower energy state. A photoelectric transducer such as a photomultiplier tube (PMT) then converts this to an electrical pulse. This is illustrated by a typical diagram of a scintillation counter in Figure 13 [20], showing how these detectors would normally be used for measurement applications. The scintillator output is optically coupled to a PMT through a light tube. Since the light emitted by the scintillator is proportional to the energy of the incident particle, they differ from GM tubes in that they can not only detect an ionizing particle but also provide a measure of its energy. The pulses from the PMT are provided to a multi-channel analyzer that records the pulse amplitude and a counter then provides a measure of the particle count.

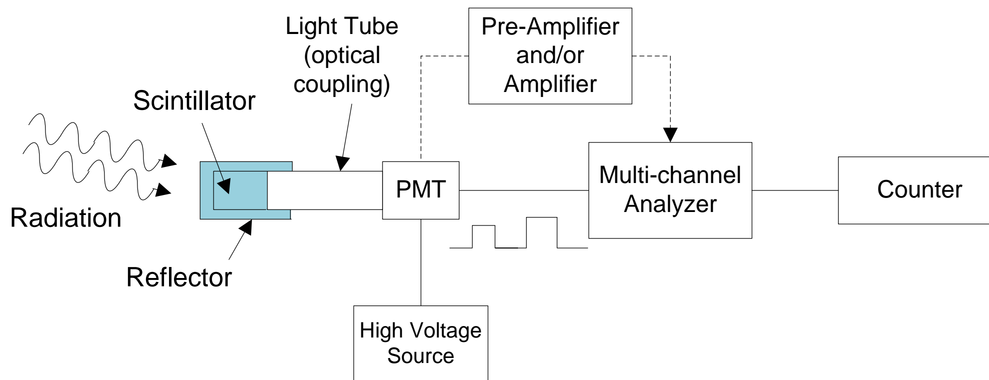


Figure 13: Typical Scintillator Detector Arrangement

Measurement of gamma radiation is an indirect process that requires the generation of charged particles. Gamma photons interacting with the atoms of the material can either directly or indirectly, through Compton scattering or photoelectric processes, produce electrons or positrons that are then detected through the normal process.

The sensing element can be of different materials namely liquids, crystalline solids, or plastics. The main requirements are that the material should be transparent to its own emitted radiation and that it should be efficient in energy conversion. The materials that are most commonly used can be divided into two categories:

1. **Organic Materials**

   Organic materials are characterized as having a fast response due to short luminescence decay times in order of a few nanoseconds. They are however not very efficient. Materials in this category can be further classified into 3 types:

   - **Liquids:** The sensing material in this group consists of a primary scintillator (solvent), a fluor and sometimes an additional emulsifier [52]. The fluor acts as a wave shifter, absorbing the light emitted by the solvent (usually in the UV spectrum) and shifting it to a different wavelength to ease detection. The entire mixture is known as a 'cocktail'. Materials such as p-phenylene oxide, p-terphenyl and benzene can be used as solvents while the fluor includes substances such as 2-methoxyethanol, toluene etc. Liquid scintillation counting has an inherent disadvantage in that in order to detect radiation the radioactive sample needs to be dissolved in the cocktail.

   - **Plastics:** Plastic scintillators have a similar material configuration consisting of a polymer base such as polystyrene or polyvinyl toluene and a fluor. They are inexpensive, robust and are capable of measuring α, β, γ or fast neutron particle radiations. They are however susceptible to degradation.

   - **Crystals:** Naphthalene was one of the earlier organic crystals to be used as scintillators. Most current scintillators though are made of anthracene or stilbene crystals. Anthracene is known to have the best efficiency among all organic scintillator materials [53]. Nonetheless, it is difficult to purify and its efficiency is still just one-third of NaI(Tl).

2. **Inorganic Crystals**

   Inorganic crystals are widely used these days since they have excellent efficiencies due to higher densities and high-Z values of the constituent elements. Most of them however have slow response times, limiting their effectiveness in high-flux environments. Alkali halide salts such as sodium iodide (NaI) or cesium iodide (CsI) are the most commonly used materials. Impurities called 'activators' are sometimes added to improve efficiency of light emission usually in the form of thallium (Tl) to

create NaI(Tl) and CsI(Tl) respectively. Glass is also sometimes used to build scintillators. NaI(Tl) has a high efficiency and a near linear energy response, but it is hygroscopic and vulnerable to mechanical shock [54].

A comparison between several common scintillator materials is listed in Table 2 [26]. NaI(Tl) has the highest efficiency but its decay time is 2 orders of magnitude more than that of organic materials, hence the material selection needs to be dependent on the implementing application.

Table 2: Comparison between Commonly-Used Scintillator Materials

Properties	**NaI(Tl)**	**CsI(T1)**	**Glass**	**Liquid**	**Plastic**
Density (g/cm^3)	3.67	4.51	~3.5	0.9	1.03
Relative Light Output	1.00	0.45	0.15	0.4	0.25
Radiation Length (cm)	2.59	1.85	4	~45	40
Luminescence (nm)	410	530	395	425	400
Decay Time (ns)	230	1000	100	2	2
Refractive index	1.85	1.80	1.55	1.50	1.58
Hygroscopic	Yes	Slightly	No	No	No

The main advantages of scintillators over GM tubes are a higher sensitivity and the ability to discriminate between different energy levels. On the other hand, they are significantly more expensive and their operation is relatively more complex. In terms of resolution, semiconductor detectors are far superior to either of the two.

3.3.3 Semiconductor Detectors

Semiconductor or solid-state detectors operate on a similar mechanism as gaseous ionization detectors in that they rely on particle interactions for the detection process. They can provide excellent resolution and response times and can be rather compact. However, they have a lower sensitivity when compared to scintillators and are as yet not as widespread as GM tubes. They can also be fairly expensive. Different types of these detectors exist with the basic operation remaining the same. The major ones are examined below.

3.3.3.1 Intrinsic Semiconductors

This is the simplest arrangement for the device to act as a detector, and it essentially consists of a pure semiconductor crystal (typically silicon (Si) or germanium (Ge)). When a charged particle passes through the device, the energy it loses in its path is used in the generation of electron-hole pairs. The detectors are normally placed between two electrodes with an applied electric potential. This prevents the charge carriers from recombining and their resultant drift due to the field creates an output pulse. Since the energy required to produce an electron-hole pair is a function of the band-gap energy of the material E_G and is constant (E_G = 3.6 eV (Si) and E_G = 2.9 eV (Ge)), the number of pairs generated gives a direct measure of the energy of the incident particle. Hence the amplitude of the output pulse is proportional to the particle energy. Moreover the energy required to generate the charge carriers is significantly less when compared to gaseous detectors (~30 eV), which results in a better energy resolution.

A known complexity can arise if residual impurities are present in the Si/Ge crystals since this results in a low resistivity [19]. This inhibits it from detecting low currents generated from smaller particle concentrations. Nonetheless, improvements in modern techniques have in recent years been able to provide highly refined semiconductor crystals. High-Purity Ge (HPGe) detectors are typically used these days mainly for measuring gamma radiation, but they are susceptible to noise due to thermal carrier generation and need to be cooled at cryogenic temperatures. Another solution is to use doping. A process known as drifting is used to produce Lithium (Li) drifted silicon

known as Si(Li) and likewise Ge(Li) for germanium. Li drifting helps provide impurity compensation [20] but Ge(Li) still needs to be cooled at liquid nitrogen temperatures to be able to provide functional gamma radiation measurements. Compound structures such as GaAs (Gallium Arsenide) and CdZnTe (Cadmium Zinc Telluride) are attractive future options since they can be operated at approx. room temperatures.

3.3.3.2 P-N / PIN Diodes

The detector in this configuration consists of a reverse-biased p-n junction diode normally in an OFF state as shown in Figure 14 [20]. When an energetic charged particle is incident on the diode, it provides sufficient energy to generate electron-hole pairs, reducing the width of the depletion region and eventually resulting in a current pulse. This is provided to an external counting/pulse height analyzing circuit. These detectors are normally not used to measure high energy particles since the depletion region is not very thick. They measure energies in the range of approx. 1.5 MeV for electrons, 20 MeV for protons, 80 MeV for alpha particles and only low energy gamma photons. The detector frequently consists of multiple elements arranged in the form of a strip.

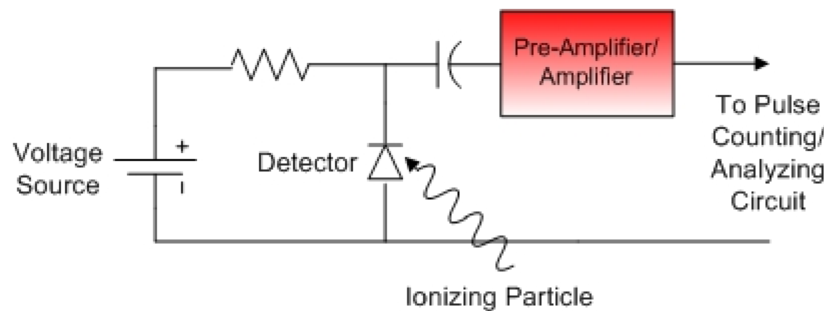


Figure 14: P-N Diode Detector Circuit

The addition of an extra intrinsic region between the p-n junction gives a p-i-n (PIN) diode that has a better response. A larger depletion region also extends its range of detection to higher energies. PIN diodes are sometimes used to detect neutron particles by using a converter such as a polyethylene radiator [21]. The neutrons collide with the radiator resulting in the generation of recoil protons with energies proportional to the incident neutrons. These protons can then be normally detected by the diode. Figure 15 [22] shows Si PIN diodes that are used to measure beam intensities at the European

Synchrotron Radiation Facility. They are custom-manufactured by Canberra-Eurisys™, having a sensitive area of 3 mm x 8 mm (width x height) and a fairly low leakage current (≤7 nA).

Figure 15: Si PIN Diodes [22]

PIN diodes have been used as rad-detectors in several space missions, some of which are listed below:

- **GeneSat-1:** A NASA CubeSat project to study genetic changes in bacteria during spaceflight.
- **Galileo In-Orbit Validation Element-A (GIOVE-A):** Developed in UK (University of Surrey), its goals included acquiring the use of the frequencies allocated by the International Telecommunications Union for Galileo and monitoring the radiation environment of the orbits planned for the Galileo constellation.
- **CORONAS-F:** It was a solar observatory launched in 2001 as a joint effort between several countries including Russia, Ukraine, UK and the US.

3.3.3.3 RADFET

Radiation Sensitive FETs (RADFETs) are inherently enhancement p-channel MOSFETs that provide a measure of the total accumulated dose in a material. In this case, the region monitoring the measurement is the gate of the MOSFET. When radiation is incident on the device, charge carriers are generated in the gate oxide region as seen in Figure 16 [23]. Some of these are lost due to recombination. Out of the free carriers, the electrons are transported out of the oxide. The less mobile holes on the other hand travel towards the substrate, several of them getting trapped in the process especially near the region of the oxide-substrate interface. The net increase in the positive charge of the device results in a relative change (ΔV_T) in the threshold voltage V_T of the MOSFET. ΔV_T increases as a function of the radiation dose, and hence the total dose absorbed by the device can be determined.

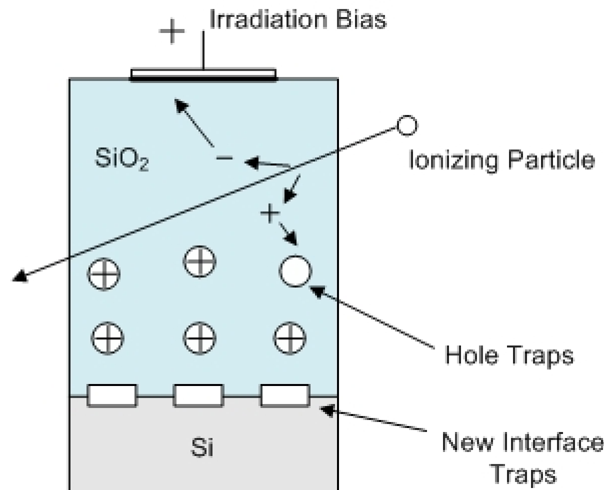


Figure 16: RADFET Operation

The simplest approach to using the RADFET in dose measurement would be to operate it in saturation. The output voltage would then consist of V_T and an additional component due to the bias current. The RADFET is typically operated in 2 modes – active and passive. In the active mode, the device is subjected to a positive gate bias which limits carrier recombination, thus increasing the sensitivity of the device. Devices in this mode are commonly used in radiotherapy applications. In the passive mode, the device has a lower sensitivity since it is operated at zero gate bias. This however allows large dose

measurements to be made, ideal for space dosimetry. Another advantage of this mode is its low power consumption. The response for both modes can be represented as [24]:

$$\Delta V_T = 0.04\ D\ t_{ox}^2\ f \text{ (active mode)}$$
$$\Delta V_T \sim 0.0022\ D^{0.4}\ t_{ox}^2 \text{ (passive mode)}$$

Where t_{ox} is the oxide thickness (µm), f is the fraction of generated holes that escape recombination which is a function of the applied gate bias, and D is the dose (rad). Since the first 2 are known device parameters, D can be determined.

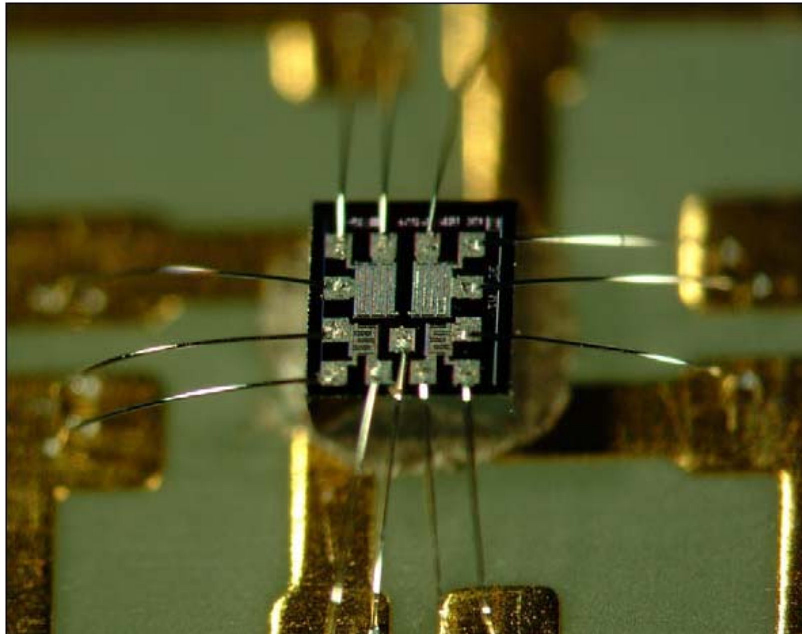


Figure 17: A 4-Transistor RADFET Chip with Bonded Wires to read the Dose Measurements [25]

Although their resolution is limited and not as good as some other detectors, RADFETs are highly suited for high-dose space environments. RADFETs can be used as single elements or in integrated chips containing several transistors of different sensitivities, to differentiate between different radiation sources. Such a configuration is shown in Figure 17 [25] which consists of 4 transistors manufactured by REM Oxford™, UK. 2 of the

transistors have sensitivities in the rad range and the other 2 in the Krad range. This is obtained by using a t_{ox} of 0.95 μm and 0.13 μm respectively.

RADFETs are often times used with other detectors such as PIN diodes to give an appraisal of the total dose. The GIOVE-A mentioned previously consisted of this arrangement wherein two RADFETs were used for dosimetry, while a PIN diode was used to measure proton fluxes. As with the other detector types mentioned earlier, RADFETs can also be modified to increase their receptivity to neutrons, specifically by enhancing the gate region with a polymeric or boron loaded substance.

Chapter 4

Single Event Upsets and Their Effects on FPGAs

4.1 Introduction

This chapter will take a look at single event phenomena including hard and soft errors. A detailed explanation of SEUs including their generation will be presented before observing their consequences on FPGAs.

4.2 Single Event Effects

This section defines the various effects one might encounter due to single event occurrences resulting from particles such as protons or heavy ions ($Z \geq 2$). These are based on the definitions provided by the Joint Electron Device Engineering Council (JEDEC) association [27].

- **Single Event Effects (SEEs)**: Any observable or measurable change in state or performance occurring in a microelectronic device, component or system that can be digital, analog or optical, resulting from a single energetic particle strike. SEEs comprise SEUs and several other effects defined below.
- **Single Event Upset (SEU)**: A soft error resulting from a transient signal induced by a single energetic particle strike.
- **Single Event Transient (SET)**: A transitory voltage spike (change of state) at a node of an integrated circuit resulting from single particle collisions.
- **Single Event Latch-up (SEL)**: An abnormal high-current state caused when single energetic particles pass through sensitive regions of the device, resulting in (in most cases) loss of device functionality.
- **Single Event Burnout (SEB)**: An event caused by a single particle collision inducing a localized high-current state in a device, and resulting in its destruction.

- **Single Event Functional Interrupt (SEFI)**: A loss of normal functionality of a device that unlike SELs, does not require power cycling to restore device operability or result in permanent damage as in SEBs.
- **Single Event Gate Rupture (SEGR)**: An event where a single energetic particle causes a breakdown in a MOSFET and a consequent formation of a conducting path through its gate oxide.
- **Single Event Hard Error (SHE)**: An irreversible change in operation from a single particle event normally resulting from a permanent damage to device elements. These are usually caused by events such as destructive latch-ups or a gate rupture.

This work was mainly conceived towards studying the SEU. This effect will therefore be the object of our focus in the current chapter as well as during the development of the test platform discussed at a later stage.

4.3 Other Radiation Effects

Besides SEEs, radiation can result in other abnormalities in microelectronic devices. The two prominent ones are reviewed in this section.

4.3.1 Total Ionizing Dose (TID) Effects

TID effects are a result of cumulative charge deposited in the device material mainly in the form of trapped charge in the insulating layers. The absorbed dose is a function of orbit parameters namely time, inclination and altitude. This is a long-term radiation effect unlike SEEs, and arises primarily from electrons, protons and resulting secondary radiation in the form of bremsstrahlung photons. The electrons and protons mainly originate from the radiation belts, and from solar events (flares).

TID can lead to device failure and its effects are seen as increased leakage currents, timing skews or threshold shifts. The RADFET, explained in Chapter 2, utilizes these very TID induced parametric shifts specifically in the threshold to provide a measure of the total dose absorbed by a device. Incidentally, the satellites that experienced solar cell

damage from the Starfish nuclear blast had failed due to intensified TID effects. TID damage can be partially reduced by shielding that provides some protection against electrons and low-energy protons or by enhanced manufacturing processes, as are employed for several rad-hardened ICs. Rad-hardened devices for space applications are in general expected to withstand a TID of 100-300 krad. The Virtex II QPro has a TID tolerance of 200 krad (Si).

4.3.2 Spacecraft Charging

Spacecraft charging is defined as the buildup of charge in the spacecraft. The discharge that follows as a result can cause structural damage to spacecraft components or can cause functional abnormalities. Depending on the location at which it occurs, this can be divided into [28]:

- **Surface Charging**: Occurs when the spacecraft surface is charged relative to the surrounding plasma. This can either be absolute, where the charge distribution over the entire surface is uniform with respect to the ambient plasma, or differential, wherein different areas of the satellite are charged to different potentials.
- **Deep Dielectric Charging:** Occurs due to charge buildup inside the spacecraft, mainly due to high-energy electrons causing charge deposition in dielectric materials.

The plasma environment, high-energy electrons from the radiation belts or the photoelectrons generated due to solar photons, any of these can act as charging sources. Damage is mainly caused through arc discharges, which occur when the field buildup, resulting from differential or dielectric charging, go beyond the material breakdown voltage. The currents associated with arcing are particularly high and can result in permanent component damage through burnouts.

4.4 SEU Mechanism

SEUs are inherently soft errors and as such non-destructive. In several cases however they can lead to SEFIs or potentially destructive errors like SELs. The SEU process is initiated when an ionized particle enters the micro-electronic material. The particle interacts with the material atoms during its passage, losing its energy in the process. The amount of energy transferred is given in terms of a Linear Energy Transfer (LET). An LET is defined as 'the amount of energy lost per unit path length by a charged particle as it traverses a material' [27]. Since the energy lost is proportional to the material density, LET is usually expressed in units of MeV-cm^2/mg. The energy transfer, results in the creation of electron-hole pairs that drift in the presence of an electric field (funneling), generating a current pulse that induces transient charges. If this occurs in the vicinity of a sensitive node, the amount of charge deposited at the node determines the occurrence of an SEU. In other words, an SEU will ensue if the deposited charge exceeds the critical charge (Q_{crit}) for the node i.e. the minimum charge that produces a soft error. Q_{crit} is a function of the feature size, specifically $Q_{crit} \alpha L^2$ for a device feature size of L x L [28]. Hence feature size scaling in modern chips has also increased their vulnerability to SEUs.

While heavy ions generate SEUs by direct ionization, protons typically do so through nuclear reactions. These can be in the form of:

1. **Elastic Collisions:** Energy is transferred to a recoil (Si) nucleus.
2. **Spallation:** Target nucleus is split into lighter particles and a heavy recoil nuclei ion.

Secondary particles from either of these reactions can then cause SEUs by the charge deposition process explained previously. In case of spallation, the angle of incidence of the original particle also need to be considered, as the charge deposition is higher at certain angles. Given that neutrons are devoid of any charge, they rely entirely on either of these two processes for generating SEUs.

4.5 Factors Inducing SEUs

This section will discuss SEUs from the perspective of the various radiation particles that produce them. The primary sources for these particles are the factors regulating the space radiation weather that were mentioned in Chapter 2.

4.5.1 Trapped Particles

The trapped particle population consists of protons, electrons and heavy ions trapped in the earth's magnetosphere. However, due to the fact that electrons do not generate SEUs [17] and trapped heavy ions do not posses sufficient penetration power to do so, we can disregard them when dealing with SEUs.

The high-energy protons in the inner radiation belt are expected be the principal factor governing SEU events for LEO satellites that encounter them several times a day. Shielding against them provides only limited protection due to the weight restrictions imposed by satellites. High density memory structures in particular, like SRAMs, exhibit low threshold LETs (LET_{th}) to protons. The particle population is affected by the solar cycle and events such as flares. The number of trapped protons decreases during solar maximum and increases during solar minimum. Flares can inject additional protons in the belt, and can sometimes even lead to the formation of additional belts thereby increasing the probability of SEU events.

A particular region of interest for LEOs is the South Atlantic Anomaly (SAA) shown in Figure 18 [29]. This region is located off the coast of Brazil at an altitude of approx. 500 km and consists of intense concentrations of protons from the inner belt. Its existence at that particular location arises due to the earth's magnetic field being offset from its center. This results in the geomagnetic field strength being weakest in that region, allowing particles from the inner belt to extend lower into the atmosphere. The map was taken at approx. 560 km. This anomaly has caused failures in many satellites. The concentrated environment can bring about an increase in SEU events, or even enhance TID effects. The Roentgen Satellite (ROSAT) that provided this map had to turn off 2 of its particle detectors while passing over this region to prevent them from getting damaged.

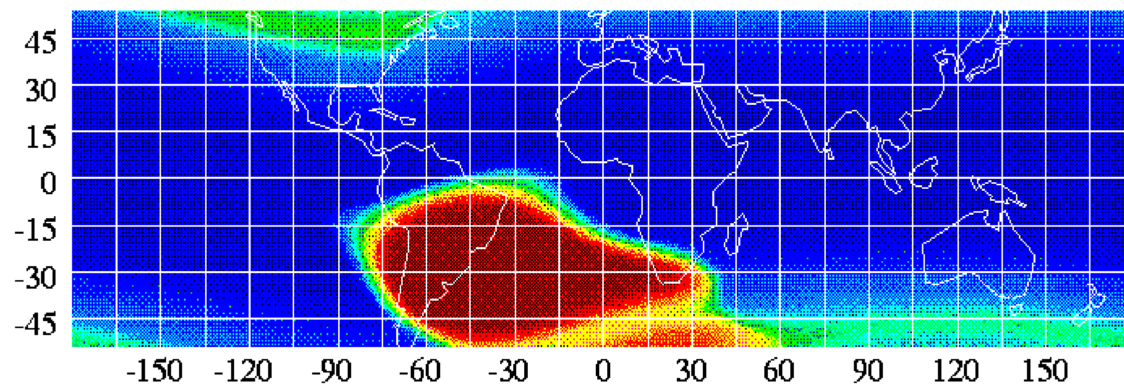


Figure 18: The South Atlantic Anomaly [29]

4.5.2 Galactic Cosmic Ray (GCR) Particles

Protons and heavy ions originating from GCRs are another major source of SEUs. They are of specific importance to high-altitude orbits like geostationary or highly-elliptical orbits. In case of LEOs the radiation belts provide some degree of shielding from their effects. The heavy ion component in GCRs is of particular significance and it contributes to SEU events by means of direct ionization. The ions as such usually possess high energy levels and can even result in multiple upsets. Their high penetrating properties, especially in case of partially ionized ions, render shielding against them ineffective. The particle fluxes exhibit a similar dependence to the solar cycle as trapped particles.

4.5.3 Particles from Solar Flares

Solar flares, though having a very low frequency of occurrence, are still important since the particles they produce can have significantly high energies. Proton energies may go up to a few 100 MeV while heavy ion particles can go to even higher energies ranging in 100s of GeV [17]. Heavy ion fluxes can in some cases be as much as four times higher than corresponding GCR levels. Another effect of flare events is that the geomagnetic storms they set off disrupt the earth's magnetosphere. This can facilitate penetration for lower energy cosmic particles hence heightening the chances of SEU events.

4.5.4 Atmospheric Neutrons

Neutrons are generally produced as secondary particles through cosmic ray spallation reactions in the atmosphere. They are mainly of concern for low (aircraft) altitude based systems, since SEUs in spacecraft are primarily from protons and heavy ions. Their energy levels typically range from 20 – 300 MeV [30] and some of them can be transported to ground levels. Neutrons dominate SEU events occurring at aircraft altitudes and are even known to affect electronics located at ground level. Apart from generating SEUs, neutrons can also cause displacement damage resulting in lattice defects in the device material. Shielding against neutrons is unfeasible since they are capable of penetrating several feet of concrete. The maximum neutron flux is at an altitude of about 15 km, known as the Pfotzer point.

4.5.5 Radioactive Materials

SEUs can sometimes result from radioactive impurities introduced in the microelectronic device during manufacturing. This is usually in the form of low-energy α-particles that are emitted from radioactive isotopes such as uranium-238, thorium-232 and their decay products. These substances are typically present as impurities in package molding compounds or in the chip metal. The α-decay can release particles with energies ranging in a few MeV and as such can generate a large number of charge carriers along their ionization path. This process is by and large independent of the altitude and is one of the sources of ground-based SEU events. Although current technology has improved the purity of packaging materials, trace amounts of such impurities are nonetheless present.

4.6 Effects of SEUs on FPGAs

4.6.1 Upset in a Basic SRAM Cell

SEUs are inherently transient, and might go undetected in several cases. But in storage circuits like latches and memory structures these events get stored. The SRAM structure that commonly acts as the configuration memory for FPGAs is based on the 6-transistor storage cell. However, the cell is known to be vulnerable to SEUs [32], consequently

compromising the functionality of the FPGA. Figure 19 shows a possibility of how an SEU might affect the cell.

The nMOS pass gates T5 and T6 are used to read/write values to the cell. For a write operation, *bit* is set to the desired logic, say ‘0’ and *~bit* to logic ‘1’. The pass transistors are then activated via the RD/WR Enable line and the value is latched in by the cross-coupled inverters. Similarly to read the state of the cell the pass gates are once again enabled.

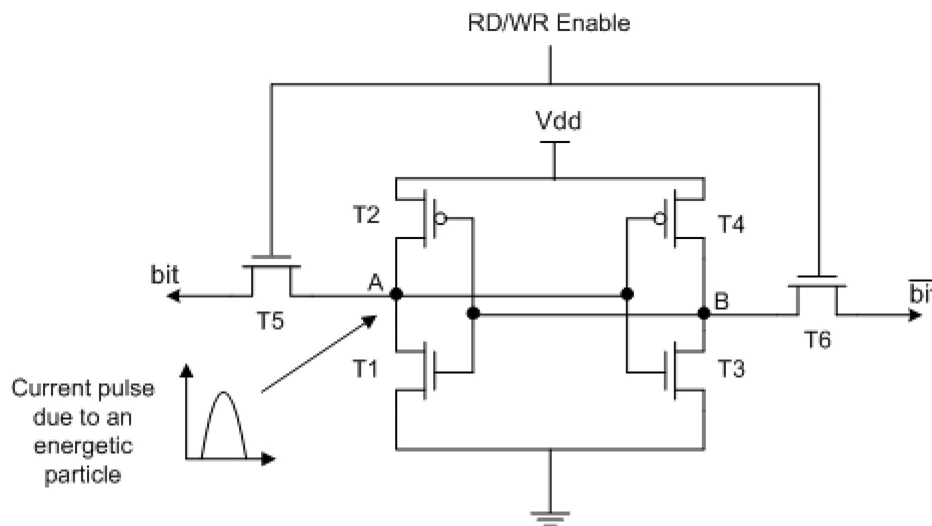


Figure 19: 6-Transistor SRAM Storage Cell

Now suppose that the cell holds a logic ‘0’ when a current pulse from a particle collision arrives at node A. The pulse travels to the gate of T3 and if its magnitude is high enough, is capable of turning on T3 thereby pulling node B to logic ‘0’. The positive feedback ensures that the change propagates to the other two transistors, eventually resulting in a logic ‘1’ getting stored in the cell. If the structure originally held a logic ‘1’, then a similar event occurring at node B would result in a logic reversal. It is important to note that the occurrence of such logic changes also depends on the design and a particle collision might not necessarily result in a bit-flip.

4.6.2 SEU Configuration Error Modes

As mentioned earlier, the FPGA memory is divided into user and configuration memory and an SEU can affect either of the two. The former case results in an undefined state getting latched in a register element. Though this might cause a temporary disruption in normal functionality, the original design remains unchanged. In the latter case however, bit-flips can cause an alteration of the logic and the affected area needs to be reprogrammed to remove the fault. Faults in this category can be either in the routing, logic resources or in the IOBs and will be further discussed.

4.6.2.1 Signal Routing Errors

The signal routing in case of SRAM-based FPGAs like the Virtex takes place through a component known as the programmable interconnect point (PIP). It is essentially a pass gate that controls routing between two wires in the device. SEUs can cause three types of failures in PIPs [33]:

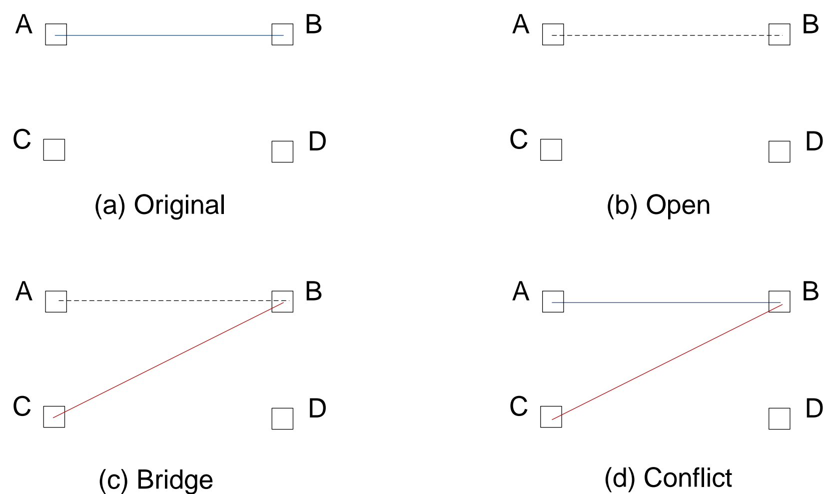


Figure 20: PIP Failure Modes

Open: Let the fault-free mode consist of node B being driven by node A as shown in Figure 20(a). In case of an *open* failure type, an SEU hit can cause the connection between A and B to be severed as shown. This might translate to a logic resource being effectively isolated from the design.

Bridge: In this scenario, the connection A-B is replaced by a new connection C-B. Node B is now driven by a different driver causing an unknown change from the expected implementation.

Conflict: The third type of failure involves a new link C-B being created while the original connection is still intact. The outcome is a signal contention due to creation of the wired-AND structure and an unknown logic value driving node B.

4.6.2.2 Errors in Logic Resources

Faults injected into CLB resources by SEU hits also fall into 3 categories based on the resource affected [34]:

LUT Defect: The fault is caused in an LUT residing in a CLB element. Since LUTs are usually used to implement logic gates and other combinatorial logic, this can cause a modification of the logic function. Figure 21 illustrates this by showing how a bit-flip can result in a 4-input OR gate implementation getting changed to a stuck-at-one fault.

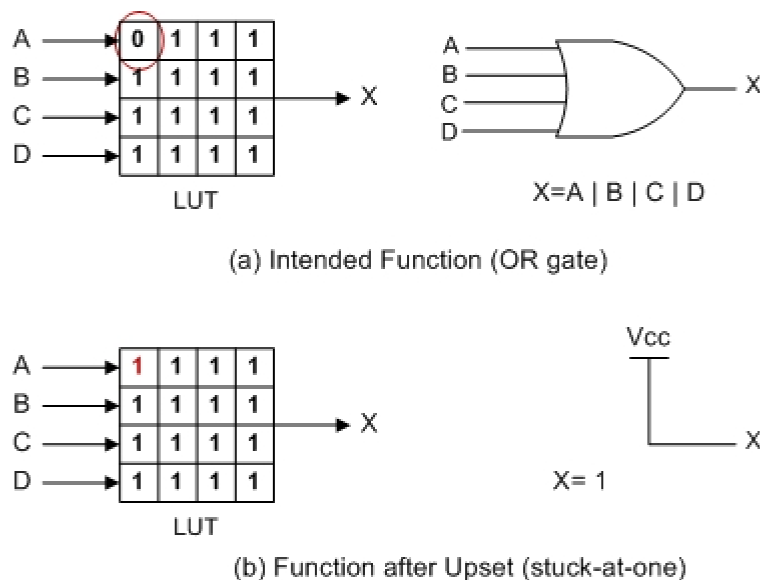


Figure 21: An Example of an LUT Error

MUX Defect: In this fault, the SEU affects the MUX select signal thereby resulting in a different signal getting routed to the output. This is shown in Figure 22, where the output was originally configured to pass input Y, but the upset causes X to be selected instead.

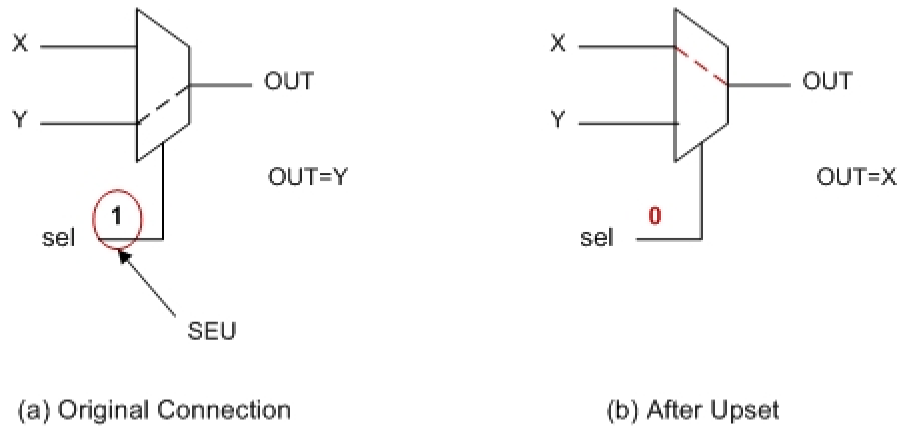


Figure 22: A MUX Defect

Initialization Defect: The configuration memory consists of bits that are associated with the CLB initialization that are used to define the behavior of the block resources. For instance, the storage elements can be configured as edge-triggered D-type flip-flops or as level-sensitive latches. Another option would be to program the function generator to behave as an LUT, RAM, or shift-register element (for more information see [31]). Upsets in these bits can therefore severely affect the design.

4.6.2.3 IOB Faults

IOBs are used to specify the operation mode for the package pins. A simplified diagram of a Virtex IOB is shown in Figure 23 [35]. A pin can be configured in either of three basic signal modes – input, output or inout (bi-directional tri-stated). Other configuration options for the IOB include selecting the I/O standard (LVCMOS, LVTTL etc.), signal terminations (pull-up, pull-down resistors), specifying the signal drive strength and configuring optional storage elements associated with each mode. Either of these features is capable of being altered by an SEU.

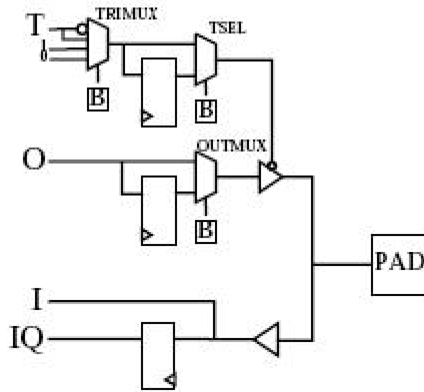
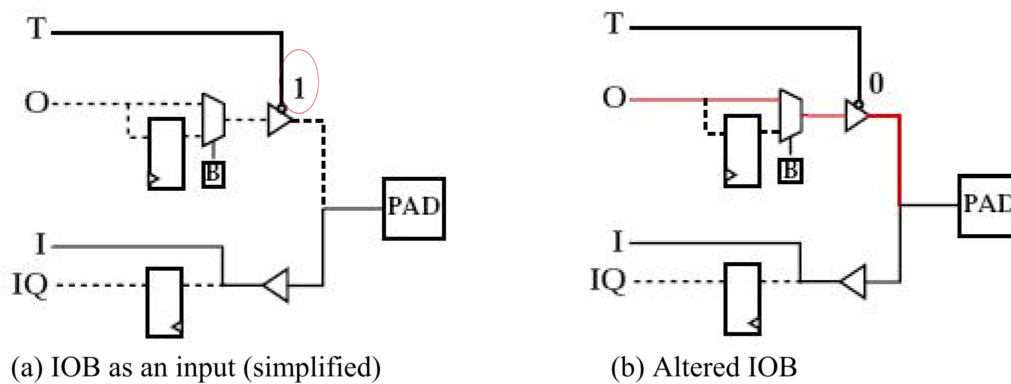


Figure 23: A Simplified IOB Element [35]

A particularly serious condition arises when an upset inverts the signal direction, as in an input signal getting transformed to an output, shown in Figure 24 [35]. The activation of the tri-state buffer controlling the output driver causes the output signal *O* to be routed to the same pad as the input. The likelihood of this kind of situation to transpire however is very rare and usually requires more than one upset. A test was conducted at the Los Alamos National Laboratory to study this for the case of single and two-bit upsets on a Virtex FPGA [35]. The FPGA consisted of 512 IOBs and a configuration size of 5,962,944 bits. They determined that at least two configuration upsets were required for the IOB failure to take place. Further, with the assumption of a uniform upset distribution, the probability of this event occurring was $2.9\text{x}10^{-11}$.



(a) IOB as an input (simplified) (b) Altered IOB

Figure 24: An Error in an IOB Element [35]

4.7 SEU Mitigation Techniques

Radiation-hardening against SEUs can be done in several ways. It can either be inherently built into the device during the manufacturing phase or be realized by making modifications to the design. The most commonly used approach incorporates redundant components that reduce chances of interruptions in the operation due to upsets. Correction of an upset following detection is usually desirable to prevent accumulation of errors especially in the case of FPGAs where multiple upsets can increase the likelihood of SEFIs. This section discusses different types of methods that are commonly employed to cope with the SEU problem. One must be aware nonetheless that an overhead is always involved, either in the form of cost, power consumption, or area.

4.7.1 Technology-Based Techniques

4.7.1.1 Silicon-on-Insulator (SOI)

In this method a different substrate consisting of an insulated SiO_2 layer over a silicon substrate is used in place of the conventional silicon bulk substrate. Bulk structures used when fabricating CMOS-based devices utilize wells to provide isolation between adjacent configurations. These however increase their vulnerability to SELs due creation of parasitic transistors. SOIs provide complete device isolation, thereby eliminating the use of such well structures, and hence reducing susceptibility to latch-ups. They can also provide some resistance against SEUs by reducing the charge collection area. Another common type of SOI is called Silicon-on-Sapphire (SOS). In this case a thin epitaxial Si layer is grown over a sapphire substrate. Both SOI and SOS improve device resistance against SELs and TID effects and can also reduce the number of SEU events, but the fabrication processes are known to be costly.

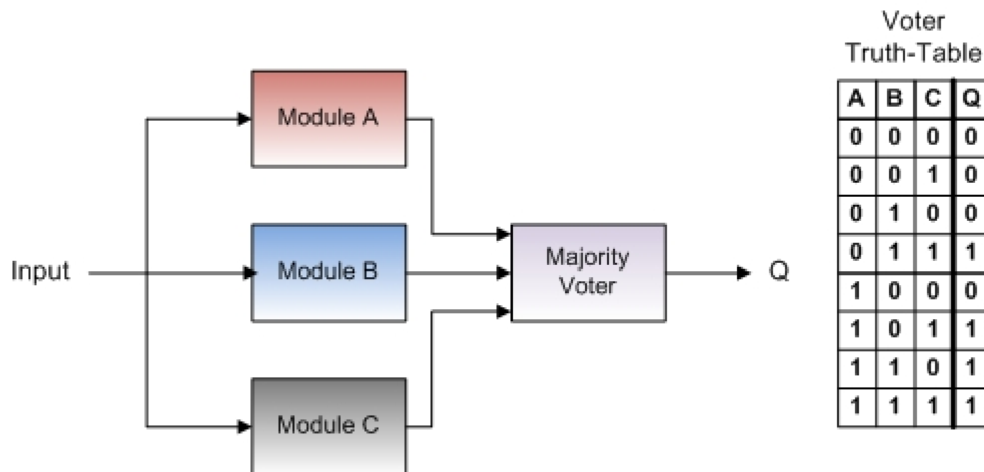
4.7.1.2 Epitaxial CMOS Process

This is a widely used technique, though not as advanced as SOI. An epitaxial layer is grown over a highly doped Si substrate. The low resistivity resulting from the highly doped region, limits its charge-collection thereby improving its immunity against SEUs. This arrangement offers limited SEL protection since the parasitic latch-up elements are still present unlike in SOIs.

4.7.2 Design-Based Techniques

4.7.2.1 Spatial Redundancy

A classic example of spatial redundancy is Triple Modular Redundancy (TMR). TMR is a widespread method and is applied to several domains other than SEU mitigation. It works on the principle of using three identical design blocks and then voting on their results to detect failures. An output is obtained as long as at least two of the blocks are functioning correctly. If a module encounters an error, it is detected by the voter and can then be corrected by reconfiguring the block. The module can represent a physical hardware resource such as a processor or a board or can be redundant logic on the same chip. An apparent weakness of the simple approach described in Figure 25 is that an upset in the voter or in the input cannot be countered. This can be resolved by triplicating the voter, and supplying the input from three identical computations through different paths.



A	B	C	Q
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Figure 25: TMR Block Diagram

A derivative of the TMR can be used to detect and correct SEUs for state-dependent logic such as counters and is shown in Figure 26 [36]. This scheme uses three voters and also protects the circuit against the accumulation of faults. Counters can be permanently knocked out of sequence by an SEU that affects its current state. Voter circuits only detect SEU events and fault-tolerance is provided by voting out the faulty sector.

However in case of counters, the affected slice needs to be restored to the correct state before an upset in another logic segment results in a faulty output. This is done in Figure 26 where a feedback path from the voter output refreshes the counter to the correct state. The clock signal is also triplicated to further improve its robustness. An evident tradeoff associated with TMR is the imposed area penalty (~3x).

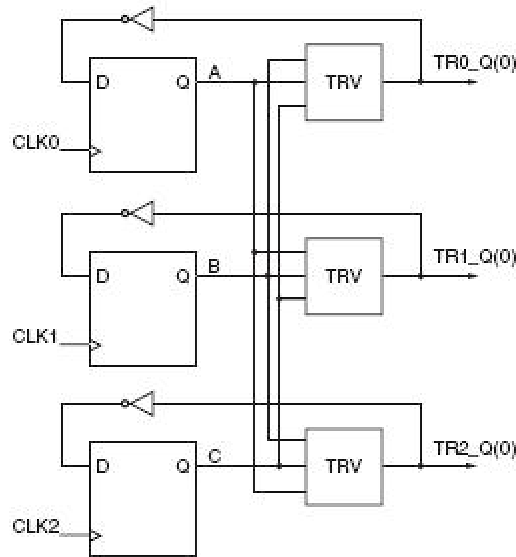


Figure 26: Implementation of TMR for State-Dependent Logic [36]

4.7.2.2 Temporal Redundancy

A variant of TMR provides redundancy in the time domain. This technique is generally used to offer mitigation against SETs in inputs or in combinatorial logic. It functions by sampling the signal at different time periods as shown in Figure 27. A delay element is added to CLK_A and is used to clock elements B and C. Assuming positive edge-triggered flip-flops, if an SET is present in the input during the rising edge of CLK_A, it gets latched in block A. However, the transient dies off before blocks B and C get affected and hence the error can be voted out. The time delay Δt is crucial for the proper operation of this circuit and must be chosen based on the maximum expected width of the SET. This scheme requires fewer elements than spatial TMR, but pays the price in terms of a lower speed, which is at least twice the anticipated SET pulse width.

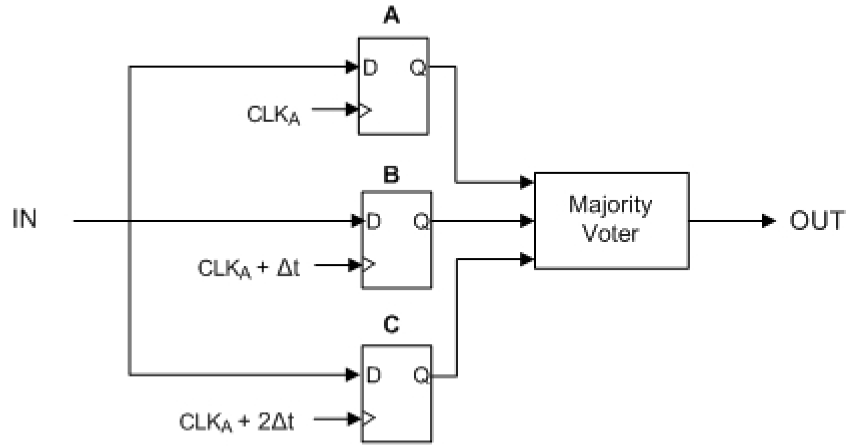


Figure 27: An Example of Temporal Redundancy

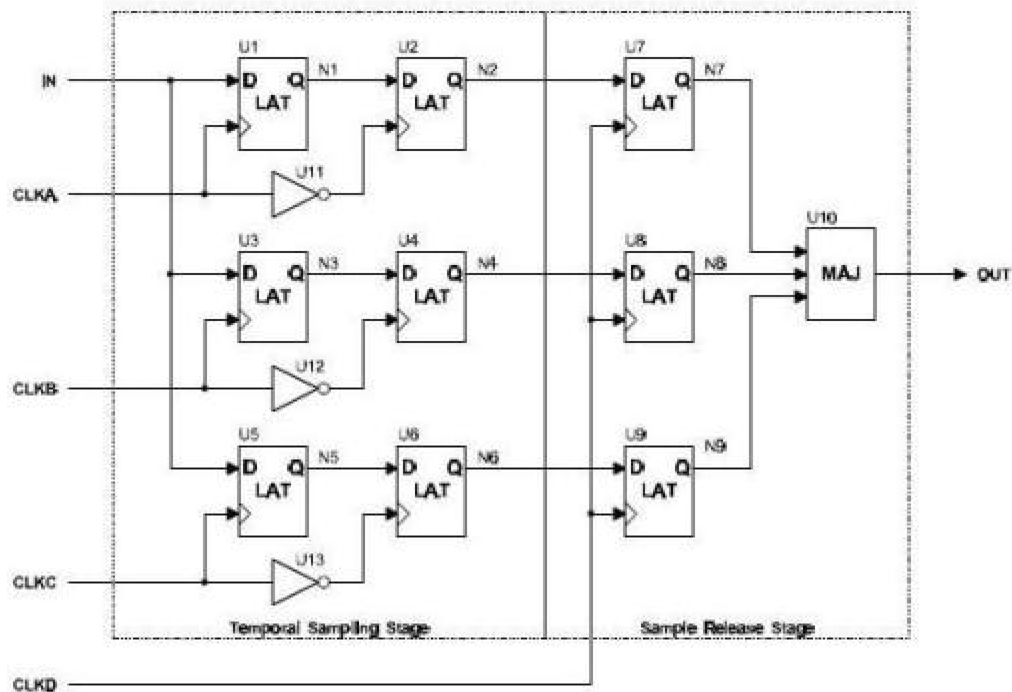


Figure 28: A Temporal Sampling Latch [37]

A different approach called the temporal sampling latch makes use of both temporal and spatial redundancy to provide better immunity against data SEU and SET events [37]. This design also provides protection against SETs in the clock signals. The circuit consists of nine level sensitive latches as shown in Figure 28. Clocks B through D are phase shifted versions of CLK_A providing the time-based redundancy, while the spatial

redundancy is provided by the three parallel paths. While an improvement from the previous methods, it also inherits weaknesses from both.

4.7.2.3 Scrubbing

Scrubbing is a technique used primarily in SRAM-based FPGAs to prevent accumulation of errors. This method uses inherent functions of these devices like readback/reconfiguration to provide SEU mitigation. This is done in two phases – detection and correction. Detection involves performing a readback operation that reads back the current data present in the configuration memory. A bit-by-bit comparison is then carried out between the readback file and the original (golden) bitstream to determine the number and position of the upsets. The device is then ‘scrubbed’ to remove the faults by reconfiguring the device with the golden configuration.

Several modifications to this flow can be made to speed up the process. The easiest way is to skip the detection phase altogether and scrub the device periodically at a predefined rate called the ‘scrub rate’. This has the added advantage of faster upset correction as well as a reduced overhead, since the detection process requires additional memory to perform the comparison. This method can be used if the only concern is to avoid accretion of faults.

In Virtex devices, the configuration memory is organized in the form of one-bit vertical frames as shown in Figure 29 [38]. These are the smallest addressable components for read/write operations involving the configuration memory. A scheme developed at the Los Alamos National Laboratory uses each of these frames to calculate a corresponding 16-bit CRC value [39]. These are then used during the comparison process to determine SEU occurrences. This methodology reduces the computation and memory required with respect to the normal scrubbing process. Many of the current FPGAs such as the Virtex II permit partial reconfiguration/readback. This gives the designer the option to scrub only affected frames instead of the entire device, greatly lowering the scrubbing duration.

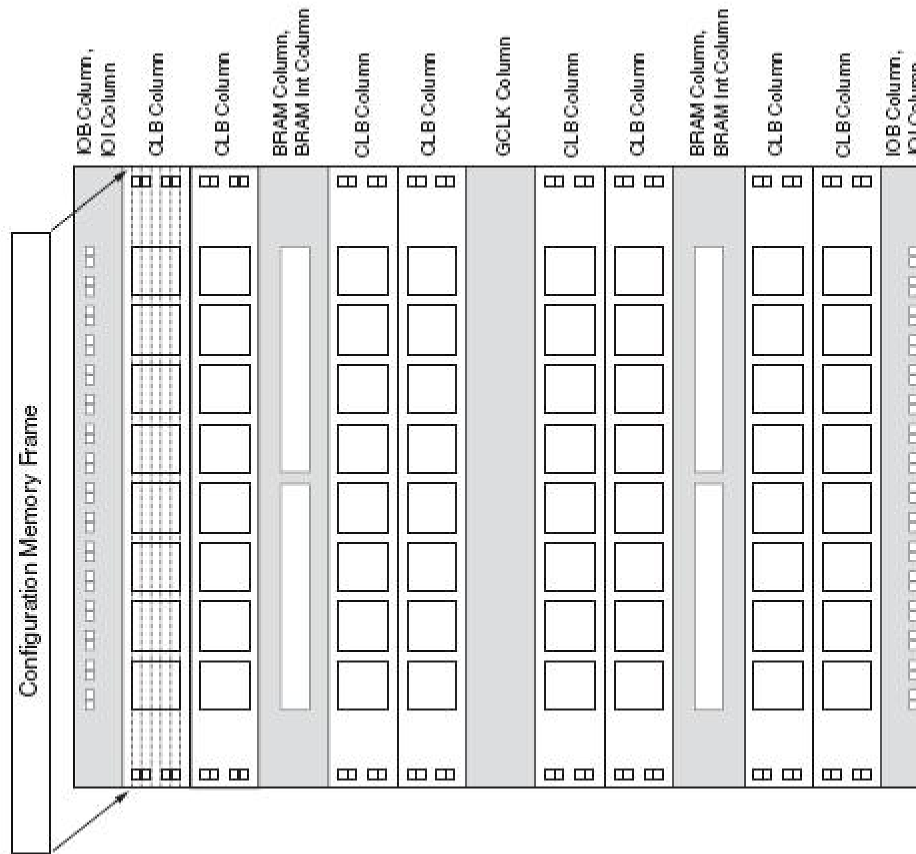


Figure 29: Configuration Frames in the Virtex II [38]

The scrub rate needs to be carefully selected depending on the estimated SEU rate. Ideally it should be high enough to correct a fault before the arrival of another upset. Based on the system requirements however, one might be willing to tolerate a few faults in exchange for a lower scrubbing overhead. Scrubbing is commonly used in conjunction with the other mitigation techniques discussed earlier, like TMR, to give a higher degree of immunity against SEUs.

Chapter 5

Implementation of the XHWIF Interface

5.1 Introduction

The purpose of this chapter is to describe the XHWIF interface that was developed for the Virtex II board used in this work. This interface is a key component of JBits since it makes it possible to use this tool with virtually any board that has a supported FPGA.

5.2 JBits

JBits is a suite of Java classes that provide an API into Xilinx FPGA bitstreams. It provides the lowest level access to the FPGA and can be used to configure CLBs as well as routing resources [41]. The bitstream is either generated from Xilinx tools such as Xilinx ISE or obtained directly from the device by initiating a readback. A major advantage of JBits is that the bitstream generation or modification is extremely fast, limited only by Java compilation times as opposed to the slow synthesis times of traditional design tools especially for large designs.

A typical design flow followed while using JBits is shown in Figure 30 [41]. Since the entire implementation is in Java, a simple Java-based development environment is used instead of the traditional CAD approach. JBits communicates with the development board through an interface known as the Xilinx HardWare InterFace (XHWIF). Since each board has a different setup, a custom XHWIF port needs to be supplied for each one. This is done through the Java Native Interface (JNI) in the form of a C/C++ program. The program essentially utilizes a known standard such as SelectMAP or Boundary-Scan to access the FPGA on the board. XHWIF provides a layer of abstraction that enables JBits applications to run on different development boards without modifications. Once a port is defined for a particular board, it can be used to connect to it locally or even remotely

through a TCP/IP based network. Other tools such as BoardScope also utilize the same XHWIF interface to connect to the hardware. BoardScope is a graphical debugging tool that can be used to probe and analyze FPGA resources.

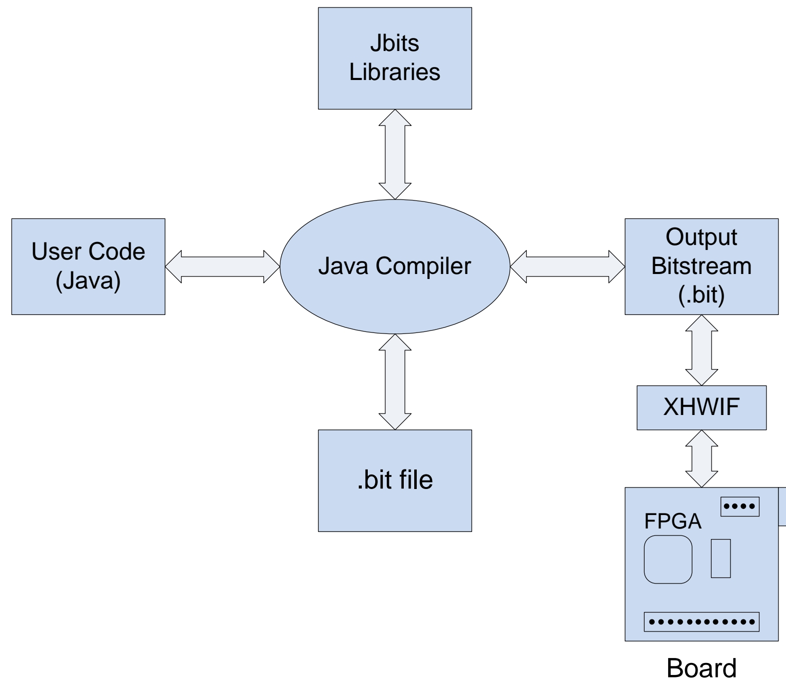


Figure 30: JBits Execution

JBits can be used to create new designs or modify existing ones. Both applications require an input file (.bit) to be specified, either in the form of a null bitstream or one synthesized from a complete design. The resources are viewed in the form of a 2-dimensional array of CLBs, each of which is referenced through a row and column. Settings for individual elements within the CLBs such as the LUTs and flip-flops along with their respective routing can then be configured to form logic functions.

Previous versions of JBits provided support for the Xilinx XC4000 and Virtex families while the current release, JBits3 was developed for the Virtex II. JBits was originally conceived as a tool to support dynamic reconfiguration for run-time reconfiguration (RTR) based systems, an option that was not supported by the normal HDL or schematic capture based flows [41]. This has provided the means to develop powerful fault recovery

mechanisms. Design changes can be made on-the-fly to create redundant logic or specify alternate routing as a workaround for faulty or permanently damaged areas on the device. The IDEA lab here at the University of Kentucky has been actively involved in the area of dependable and fault-tolerant computing. Hence this function of JBits is particularly appealing and needs to be further investigated in future.

5.3 Virtex-II Evaluation Board

The board used for this work is the Virtex-II Evaluation Board manufactured by Avnet. Figure 31 shows the board and highlights the elements utilized for this work, namely [42]:

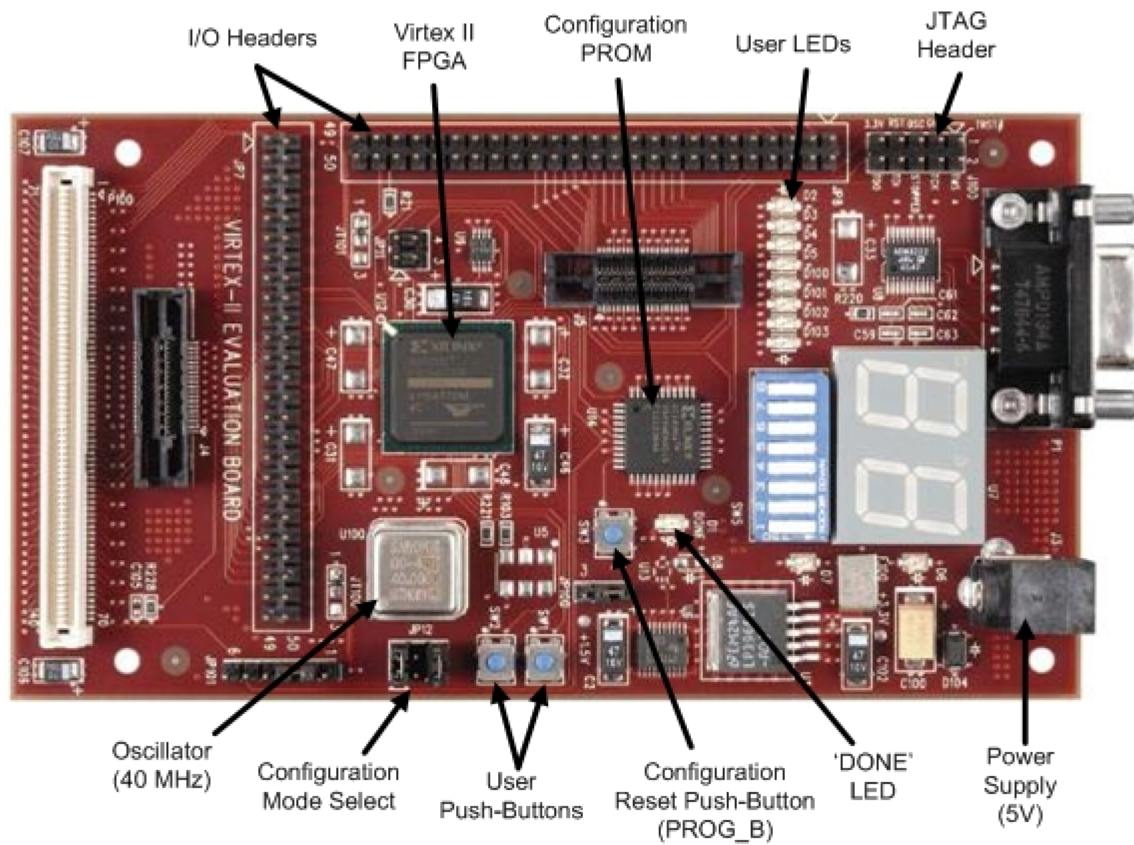


Figure 31: The Virtex-II Evaluation Board

- A Xilinx Virtex II XC2V1000-4FG256 FPGA.
- A Xilinx XC18V04VQ44C configuration EEPROM having a capacity of 4 Mbits.
- Two 50-pin I/O headers (92 FPGA I/O lines, 4 GND, 2 clock and 2 VCC).
- Header for providing JTAG connectivity.
- Eight discrete LEDs and two push-buttons for testing.
- Status LEDs (power and 'DONE').
- A push-button for initiating a configuration reset.
- A jumper header for configuration mode selection.
- A 40 MHz oscillator that supplies the global clock to the FPGA.

The board features several other options that were not used such as an RS-232 serial port, DIP switches, a digital thermometer, MICTOR connectors for logic analyzer support, etc. Three of the five configuration modes supported by the FPGA can be selected using the jumpers on the mode select header as shown in Table 3 (M0 at leftmost position). SelectMAP is an 8-bit wide parallel configuration mode, which although supported by the PROM, has not been enabled on the board. The master/slave serial modes allow the FPGA to be configured from the EEPROM, while boundary-scan can be used to program the FPGA from an external off-board source. The boundary-scan mode was selected to access the FPGA and will be discussed in the next section.

Table 3: Board Configuration Mode Settings

Configuration Mode	**M0**	**M1**	**M2**
Master Serial	Open	Open	Open
Slave Serial	Jumpered	Jumpered	Jumpered
Boundary-Scan (JTAG)	Jumpered	Open	Jumpered
Master SelectMAP	Not Supported		
Slave SelectMAP	Not Supported		

The ‘DONE’ LED is used to indicate a successful device configuration. It is driven by the *DONE* pin of the Virtex II, a configuration status pin that is actively pulled low while programming and released after the device has been properly configured. The configuration reset push-button shown in the figure is connected to the FPGA *PROG_B* pin. This is an active-low pin that is used to clear the configuration memory of the device. The board also has an additional oscillator socket in case a higher system clock speed is required.

5.4 The Joint Test Action Group (JTAG) Interface

JTAG or boundary-scan is the term typically used for the IEEE standard 1149.1. This standard was formulated as a means for testing and debugging devices and is supported by a large number of ICs today. In addition to using it for testing, many of the FPGAs can also be programmed through the JTAG interface.

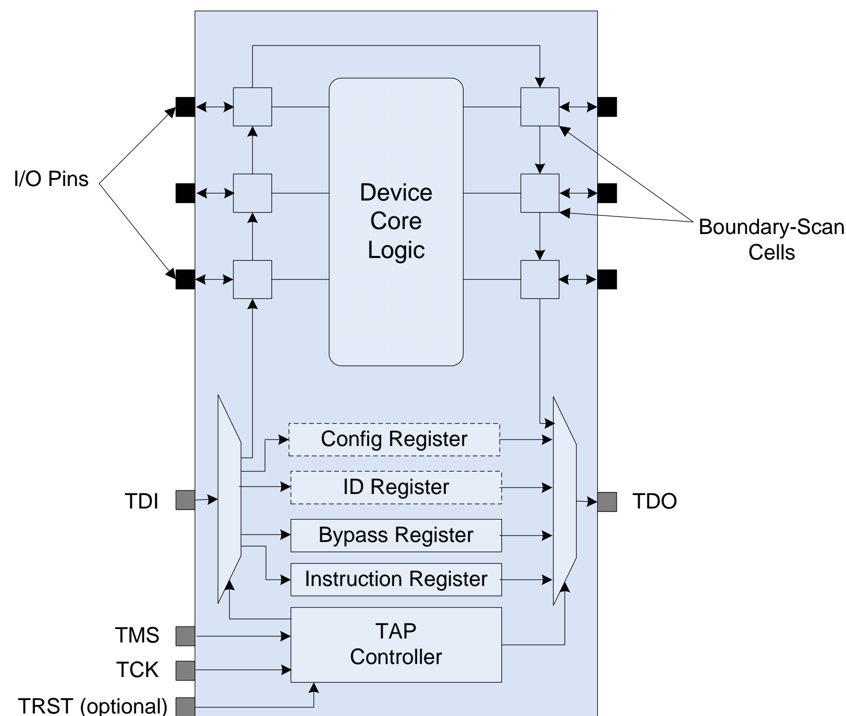


Figure 32: Boundary-Scan Architecture

The standard defines a set of mandatory elements that need to be present for a device to be considered JTAG compliant. These consist of the Test Access Port (TAP), the TAP controller, the instruction register, the instruction decoder, the boundary-scan register, and the bypass register [38]. Other than these, devices can also support optional components to facilitate testing or provide additional services. The Virtex II for example has an optional 32-bit identification register and a 64-bit configuration register. The boundary-scan architecture is shown in Figure 32. The boundary-scan cells are used during an active test operation and collectively form the boundary-scan register

The TAP connector is composed of 5 pins:

1. **TCK (Test Clock) –** This is the JTAG clock that sequences the TAP controller and the registers. It is controlled by the device administering the test or configuration. The pulse rates need not be fixed.
2. **TMS (Test Mode Select) –** This pin decides the state of the TAP controller and is registered at the rising edge of TCK. It has an internal pull-up resistor incase the pin is not driven.
3. **TDI (Test Data In) –** This is the serial input line for the JTAG instruction and data registers. The register that receives the data is selected based on the state of the TAP controller and the instruction stored in the instruction register. The pin is sampled on the rising edge of TCK
4. **TDO (Test Data Out) –** This is the serial output line of the JTAG interface. It supplies the data from the current selected register to the JTAG chain. Its value is updated on the falling edge of TCK.
5. **TRST (Test Reset) –** This is an optional pin that is used to asynchronously initialize the TAP controller. When driven low, the JTAG register is disabled and the device is placed in a normal mode.

The Virtex II TAP incorporates the first four mandatory pins in its TAP. In addition, it has internal pull-up resistors for TMS and TDI incase they are not actively driven.

The TAP controller is essentially a 16-state finite state machine as shown in Figure 33. The state of the controller is decided by the value on the TMS pin at the rising edge of TCK (shown adjacent to each state). The state machine is divided into two sequences, one for the Data Register (DR) and the other for the Instruction Register (IR). Once the register is selected, TDI or TDO is used to shift-in/shift-out data from it. A list of all the JTAG registers as well as the available instructions in the Virtex II can be found in Appendix A. A short description of the TAP states follows:

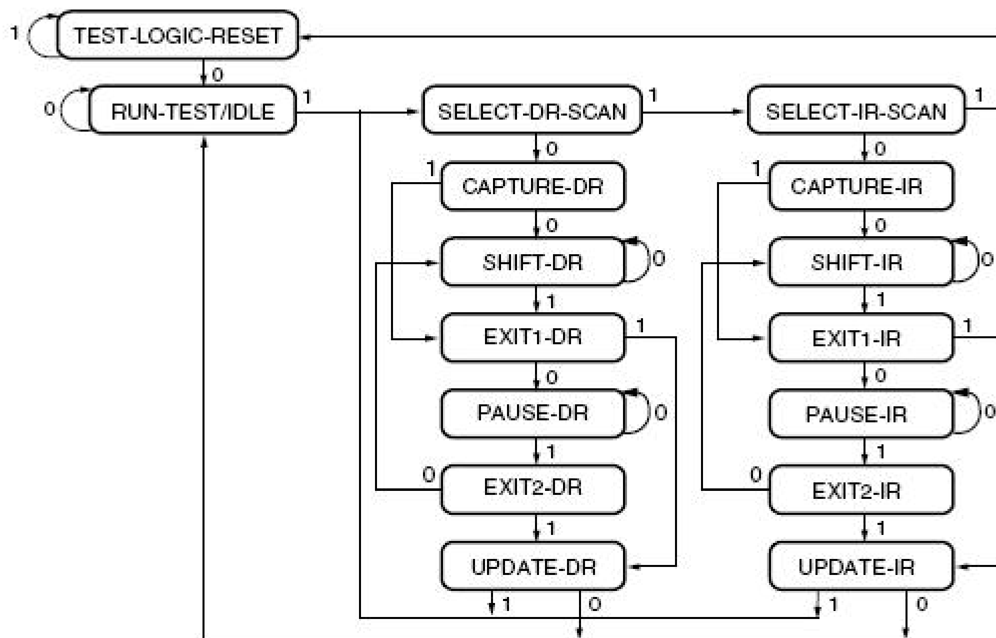


Figure 33: TAP Controller State Diagram [38]

- **Test-Logic-Reset:** This state brings the device into the normal mode by disabling the test logic. Depending on the device, the IR is either loaded with an IDCODE instruction or a BYPASS instruction.
- **Run-Test-Idle:** As the name suggests, this state can be used to run a test, if any, associated with the current instruction. Otherwise the test logic is in an idle state.
- **Select-DR-Scan:** This functions as an intermediate state to enter the DR scan path or proceed to the IR sequence.
- **Select-IR-Scan:** This state is used to enter the IR scan path or else perform a Test-Logic-Reset.

- **Capture-DR:** In this state, data is loaded into the selected DR from parallel inputs on the rising edge of TCK.
- **Capture-IR:** The current device status is captured into the IR in a parallel manner. The two least significant bits should be '01'.
- **Shift-DR/ Shift-IR:** In this state, the DR/IR is linked between TDI and TDO. The captured value can be viewed through TDO while new data is being shifted in from TDI. When shifting in an instruction or a data stream, the last bit needs to be loaded while transitioning to Exit1-DR/Exit1-IR i.e. with TMS = 1.
- **Exit1-DR/ Exit1-IR:** This state is encountered when exiting from a Shift-DR/Shift-IR operation.
- **Pause-DR/ Pause-IR:** This state can be used to temporarily freeze a data/instruction shift operation.
- **Exit2-DR/ Exit2-IR:** This state givesthe TAP controller the option to return to the Shift-DR/Shift-IR state.
- **Update-DR:** The data in the current DR is latched on the falling edge of TCK. This effectively updates the DR.
- **Update-IR:** In this state the instruction in the IR gets latched on the falling edge of TCK, making it the current active instruction.

5.5 XHWIF Implementation for the Virtex II Board

5.5.1 The JTAG Scan Chain

Developing an XHWIF interface for a hardware platform is based upon two key factors. The first factor is the number and types of devices present on the board that are a part of the JTAG chain, while the second is the communication standard used to access these devices. When multiple JTAG-compliant devices are present on a board, they can either each have individual JTAG ports, or they can be connected serially on the same scan chain. The latter arrangement known as a 'daisy-chain' requires just one JTAG port to access all the devices and is employed by most of the boards including the one in question. The daisy-chain sequence herein consists of the PROM followed by the Virtex

II as shown in Figure 34. The normal single-device configuration approach needs to be modified in order to be applicable in this case.

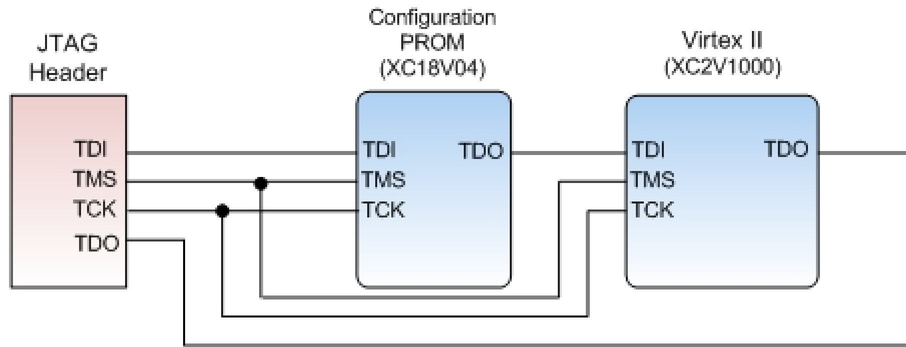


Figure 34: Boundary-Scan Chain on the Virtex II Board

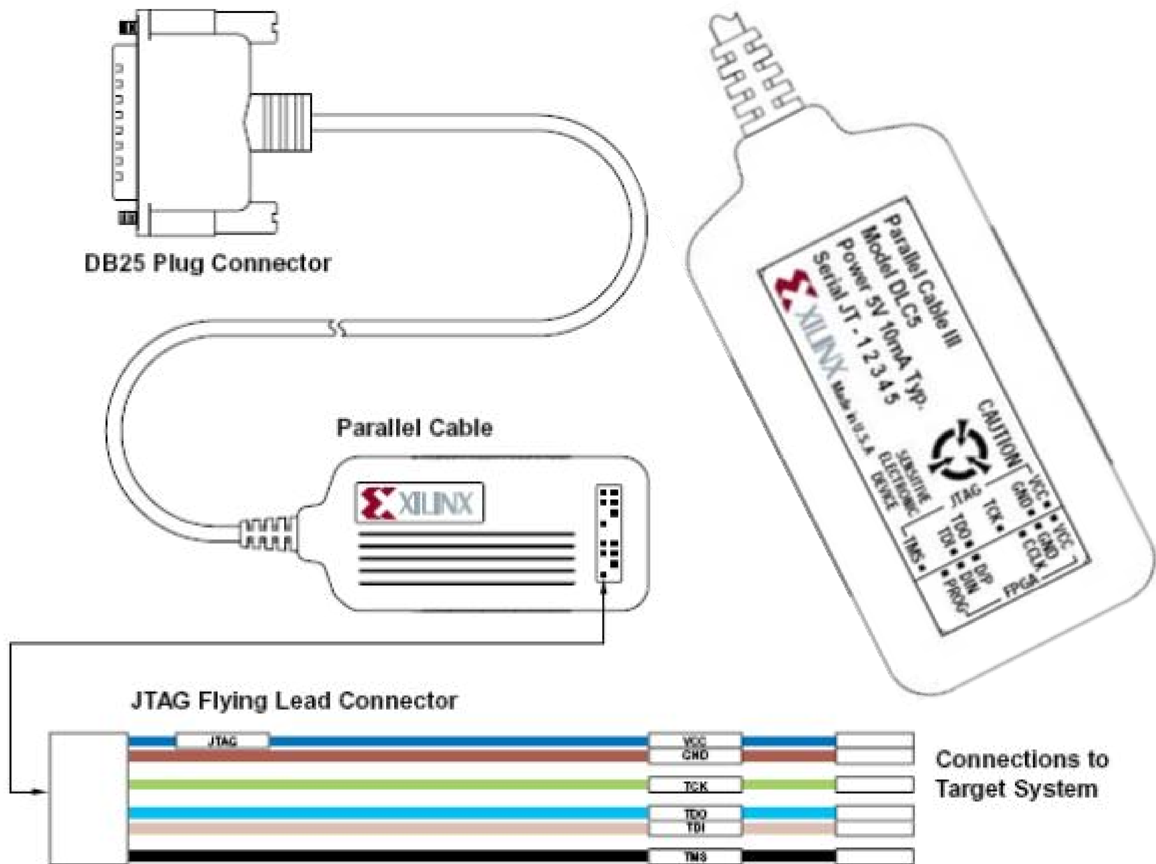


Figure 35: Parallel Cable III [43]

5.5.2 Host PC-Board Communication Interface

The link between the host computer and the board is provided by means of a Xilinx Parallel Download Cable III. The cable provides a parallel port (DB-25) connection on the PC side and a flying-lead JTAG pinout (JTAG header) on the FPGA side. An alternate header (FPGA header) is also provided for configuring in slave-serial mode. The TAP pins map onto the Data and Status ports of the parallel-port interface as indicated by the schematic of Figure 36. All the pins are mapped to true logic.

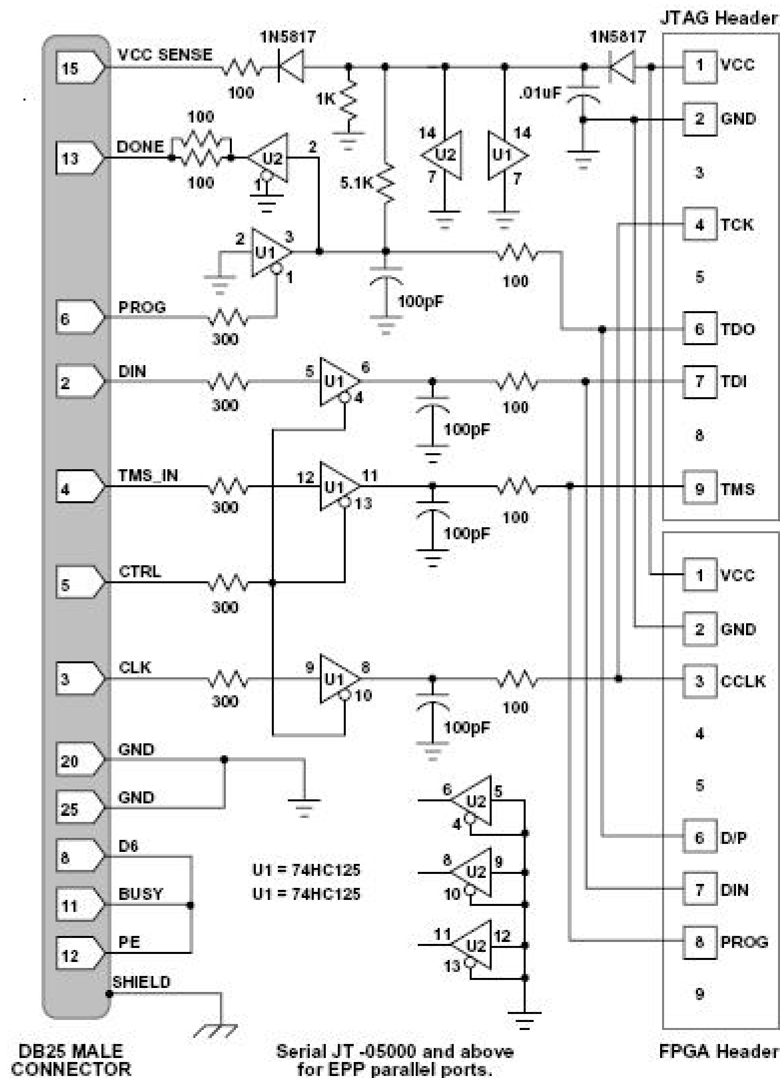


Figure 36: Parallel Cable III Schematic [43]

5.5.3 The Java Native Methods

The XHWIF interface documents several methods that need to be supplied. These include methods to connect/disconnect, perform configuration, readback, read/write to on-board RAM, control the system clock etc. While the methods need to be defined for the interface to be created, a complete implementation for all of them is not required. The functions that are never invoked need only be implemented as dummy functions that at most return a value. The native methods for the interface were created in C++ using the Microsoft™ Visual Studio™ environment. The two key functions that were developed for this port were for FPGA configuration and readback. Other than these some functions were added to enhance functionality.

5.5.3.1 Connecting to the Board

The *Connect()* method functions as an initialization module before other methods are invoked. The function creates a JTAG Port object in order to control the JTAG interface. There is no specific instruction as such required to establish communication with the board. This function merely verifies that the cable connecting the PC to the board is working correctly. This is done via a 'sanity check', i.e. the IDs of the PROM and the FPGA are acquired and checked against their actual values. This ensures that the devices are working properly and there are no faults in the setup. A problem was encountered initially while getting the device IDs and a constant value of 0x00000000 was being read from the devices. The cause was traced to the tri-state buffer controlled by pin 6 of the parallel port (Figure 36) that was getting enabled by default and continually driving TDO low.

5.5.3.2 Getting the Device IDs

The device IDs reside in the 32-bit identification register and can be obtained by shifting in an IDCODE instruction (Appendix A) into the JTAG instruction register. The *CheckIdCode()* method used to execute this follows a multiple-device approach wherein it accesses the devices one at a time simultaneously bypassing the other. Since the board does not have provisions to bypass the devices in hardware, this needs to be done via JTAG instructions. The target device is loaded with the required instruction while the

other device is programmed with a BYPASS instruction. Once a device is bypassed, it acts as a 1-bit buffer by establishing a link between its TDI and TDO pins through the bypass register. It should be noted that the TAP state-machines for all the devices are functionally equivalent. A flowchart depicting the sequence followed to get Device IDs is shown in Figure 37.

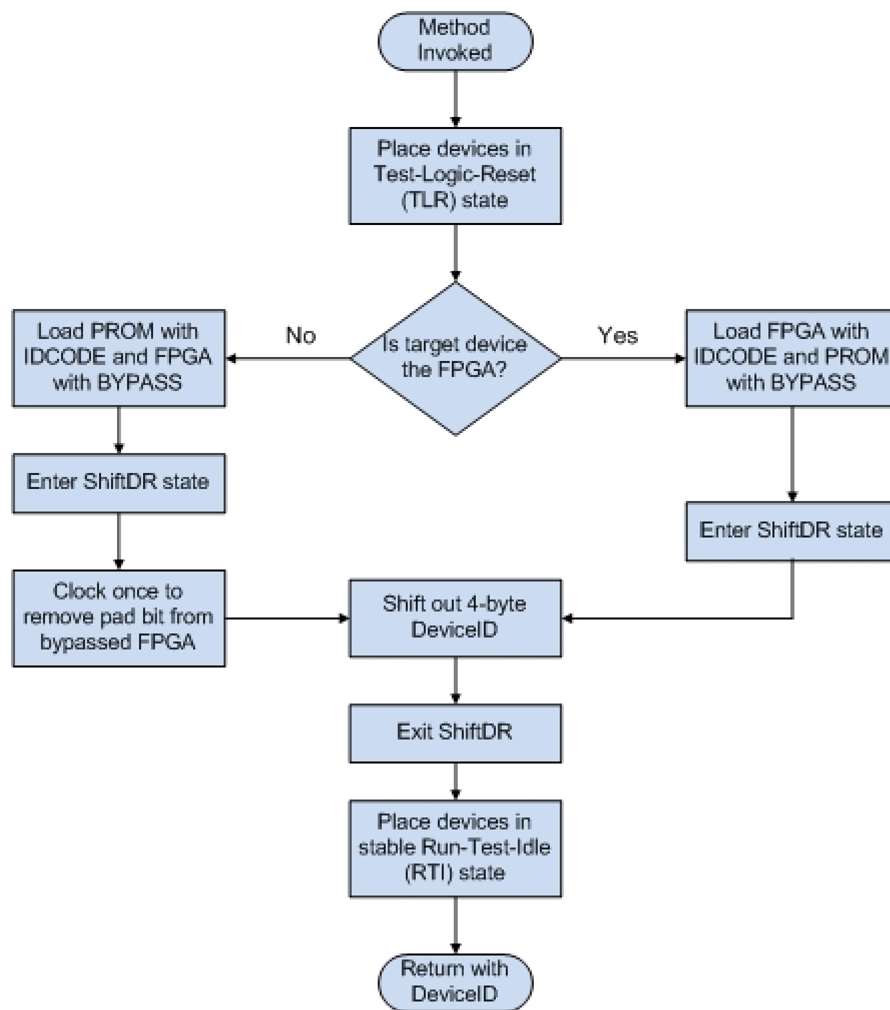


Figure 37: Flowchart for Fetching Device IDs

5.5.3.3 FPGA Configuration

The *SetConfiguration()* method at present only supports configuration of the FPGA. The PROM is only used initially while accessing its Device ID and is otherwise bypassed. Configuration as well as readback in the Virtex II is managed by configuration registers. The registers are selected by individual packets in the device bitstream and are then used to specify configuration settings or handle internal signals before programming the configuration memory. Appendix B documents these registers and gives a detailed explanation for a few of them. Some of the commonly used data packets and packet headers are also provided.

In case of the Virtex II, a configuration can either imply a normal configuration or a reconfiguration. From our perspective, a normal configuration involves configuring a device from scratch, while a reconfiguration is performed on a previously configured device without clearing its configuration memory. Reconfiguration involves some additional steps and can be performed while the device is still operating (active reconfiguration) or by first placing the device in a safe shutdown state (shutdown reconfiguration). A shutdown reconfiguration was used here since the first method can result in data contention and can potentially damage the device.

Figure 38 shows the main steps involved in configuring the Virtex II. Note that instructions are loaded in LSB first while a bitstream shift starts with the MSB. The startup and shutdown sequences are identical and entail placing the TAP in the RTI state and providing a minimum of 12 clock cycles. Certain additional steps (not shown) are also involved while performing a reconfiguration. For instance, the device will ignore packet data until it encounters a synchronization word (0xAA995566). Also, in order to prevent a possible contention with previous configuration data, the AGHIGH command needs to be written to the command register (CMD) before programming the device. This places all interconnects in a high-impedance state by asserting the *GHIGH_B* signal. The current error checking routine makes use of an Instruction Capture as specified in the Virtex II BSDL file to monitor the *DONE* pin. This is a simple approach that only requires a few TDO read cycles. The pin can also be monitored by reading the device

Status Register (see Appendix B). This method was used initially for extended debugging.

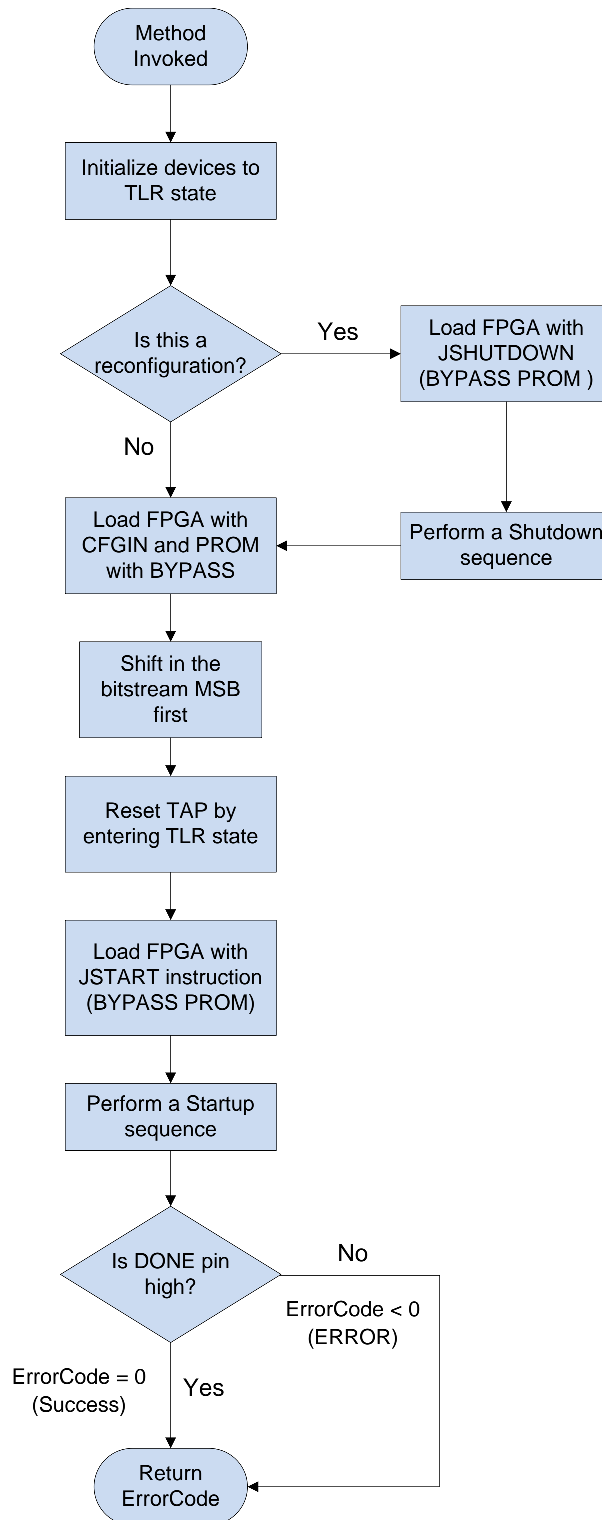


Figure 38: FPGA Configuration Flowchart

5.5.3.4 Readback

In addition to allowing the configuration memory to be readback, the Virtex II also allows the contents of the user memory elements such as the BlockRAM, LUT RAM, etc. to be examined. While useful for debugging a design, it is fairly difficult to identify faults in these bits since the contents constantly change based on the current state of the registers. Hence it was decided to ignore these bits while verifying a device readback.

Similar to the reconfiguration process, the FPGA supports both active and shutdown readback. Again the former approach can cause problems by altering the user memory contents and was therefore not chosen for the current *GetConfiguration()* implementation. Figure 39 highlights the major segments of the function. The shutdown sequence operates in an opposite manner to startup and is signaled by the *DONE* pin being pulled low. In order to readback from the Virtex II, the Frame Address Register (FAR) must be configured with the first frame that is to be read, 0x00000000 in our case since we are performing a full readback. This needs to be followed by the amount of data to be scanned in the form of 32-bit words. Now the XC2V1000 consists of a total of 1104 frames, with each frame containing 106 words [38]. This amounts to 117024 configuration words. However the frame buffer pipelines the read data in a manner that the current frame is shifted out while a frame is fetched from the device memory, hence the first frame always consists of unneeded pad data and can be discarded. For this reason the Frame Data Output Register (FDRO) needs to be configured with a total of (1104 + 1) frames.

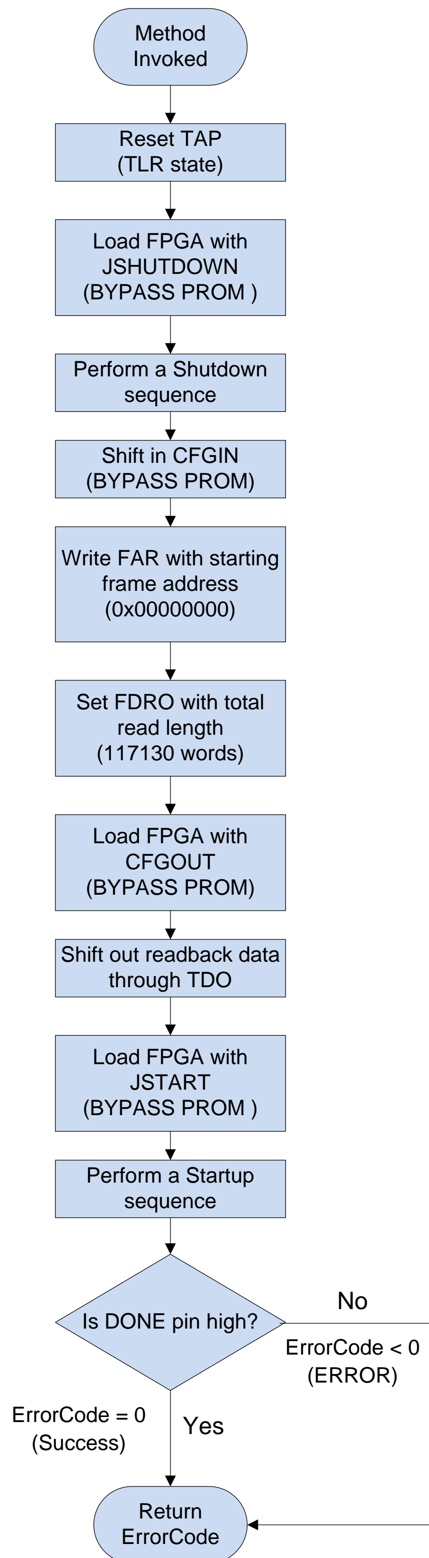


Figure 39: Flowchart showing a Readback Sequence

While performing the readback it was initially found that a status LED, used in the design to indicate that the logic operation had been initiated, would remain OFF although the *DONE* pin indicated a successful readback. Closer inspection showed that all the logic signals appeared to be in a high-impedance state. The problem was found to be that the *GHIGH_B* signal was disabling all interconnects. The LFRM command had to be used to de-activate the signal thereby enabling interconnects. Both partial reconfiguration and readback can be implemented with some modifications. For example, by specifying the address of the targeted frame to the FAR and changing the number of words to be read, only the contents of selected frames can be read back instead of the entire bitstream.

5.5.4 The XHWIF Interface

Once the native methods have been realized, they are supplied to the XHWIF JNI in the form of a library (in our case a Dynamic Linked Library (DLL)). The test application can then use an XHWIF object to communicate with the board. The Java test environment including the XHWIF interface was developed using the Eclipse™ IDE. Since the source for XHWIF was not provided in the JBits3 SDK package, a subclass was utilized to implement additional functions.

Chapter 6

Testing Methodology

6.1 Introduction

The test setup utilizes JBits for performing both the fault simulation as well as correction. A simple form of fault detection would involve periodically performing a readback on the device, and scanning the bits against the golden bitstream for possible errors. However in cases when no faults have occurred this would waste processor computing time. A dedicated logic block was therefore created to perform the function of uncovering errors and provide a means of alerting the processor on detected occurrences.

6.2 SEU Detection Logic

SEU characterization in case of FPGAs can be done through two modes, static or dynamic. Both modes are used to determine the device SEU cross-section using known parametric models such as the Weibull fit [45]. The SEU cross-section is defined as the number of upset events occurring per unit particle fluence and is expressed in units of cm^2/bit.

In the static mode the FPGA is not clocked and the detection scheme merely consists of periodically reading the SRAM configuration memory for affected bits. This method cannot detect upsets in combinatorial elements or SETs. Also, the FPGA is typically expected to be performing some sort of function when integrated into a spacecraft system. Thus the detection strategy developed here was based on dynamic device operation. The approach described in [45] was used wherein identical shift register (SFR) slices were created and their outputs were constantly compared for mismatches due to potential SEUs. The SFRs were fed with a ‘checkerboard’ pattern (alternating ‘1’s and ‘0’s) as depicted in Figure 40. Four distinct DCMs were used to provide the clock for the

SFR slices and their corresponding pattern generators. Upset detection can be carried out by using a minimum of two blocks. However a higher number of logic slices virtually factors out the possibility of upsets affecting the same stage of a redundant SFR and thereby going undetected. Once a fault is registered by the comparator, it activates the *SEU_STATUS* signal that can be used by a microcontroller to initiate a readback. After the affected bits have been recorded, the controller can then clear the register for detecting future upsets.

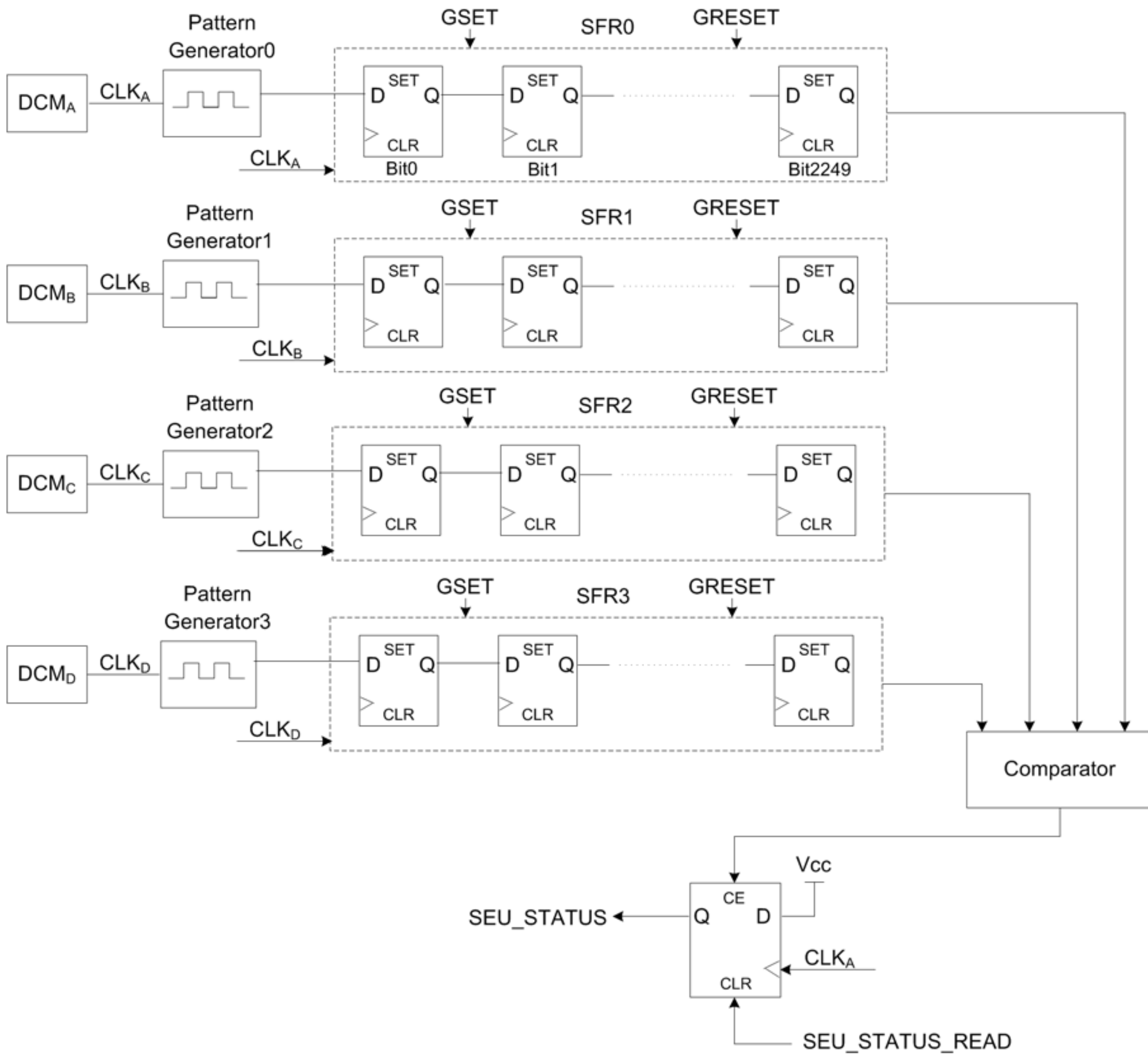


Figure 40: SEU Detection Logic

The design was created in Verilog using Xilinx ISE. The lengths of the SFRs were kept large in order to use a greater number of resources (CLB flip-flops in this case). This implementation utilized 99% of the available Virtex II slices and 89% of the CLB flip-flops. Alternately, resource utilization could also be increased by increasing the number of blocks instead of the SFR length. Since the emphasis of this logic was only on SEU detection, TMR techniques were not implemented.

```
XhwifTest.java    virtex2xhwif.java

 4** @version 1.1 Oct 2007
 5** @author  Shadab Ambat
 6**
 7** This class implements the XHWIF interface Virtex2Xhwif created for
 8** the Avnet Virtex II Evaluation Board containing a XC2V1000 Virtex II device
 9** <p>
10** Usage:  XhwifTest -board [options <file.bit>]
11** <p>
12** NOTE: Although the interface is able to detect 2 devices on the board by
13**       reading their device IDs only the Virtex II device is supported by JBits3
14**/
15import java.nio.channels.*;
16import java.io.*;
17import java.nio.ByteBuffer;
18import com.xilinx.JBits.Virtex2.JBits;
19import com.xilinx.JBits.Virtex2.Device;
```

```
Problems  Javadoc  Declaration  Search  Console
<terminated> XHWIFTest [Java Application] F:\Program Files\Java\jre1.6.0_02\bin\javaw.exe (Jan 6, 2008 5:04:11 PM)
Usage:  java XhwifTest -board [options <file.bit>]

where board is:

        -virtex2xhwif           Avnet Virtex II Eval Board

where [options] include:

        -help                   Print this help message and exit
        -c                      Configure Virtex II device using <file.bit>
        -r                      Enable Readback from device
        -v                      Verify Readback data against original bitstream file
                                Requires <file.bit> to be specified and a <file.msk>
                                to be present in the same path.
                                Also requires atleast one Readback to have been
                                performed earlier (<file.rb> must exist in current dir)
        -f                      Fault Injection. Since this basically relies on RTR a full
                                configuration is recommended before using this option.
        <file.bit>              Bitstream file. Required with the -c, -v or -f option
                                Needs a relative or absolute path if not in current dir
```

Figure 41: Screenshot of the Java Test Program

6.3 Test Procedure

A test program in Java was created specifically for testing the newly developed XHWIF interface and for managing the faults in the bitstream. The program accepts the board name (implemented XHWIF) and the input bitstream as arguments. An *options* switch is used to instruct the program on the operation to be performed (configuration (c), readback (r), fault injection (f) and/or verification (v)). If the design bitstream is not in the program directory, a relative or absolute path needs to be specified.

6.3.1 Phase I: Fault-Free Operation

The aim of this section is mainly proof-of-operation. The bitstream generated for the detection logic is used to program the FPGA, and the same file is then read back from the device and verified. This flow is used to identify any bugs in the implemented native methods and to make sure that they are removed before carrying out any further testing. The content of this section will also serve as an introduction to the various aspects of the test program and the steps involved when verifying a Virtex II bitstream.

The test application is invoked with three options (-cvr) and a relative path to the original design file. The program starts off by scanning the JTAG compliant devices on the board and displaying their device IDs. Although the board has 2 such devices, configuration and readback procedures have only been supplied for the Virtex II as displayed on the console:

```
Connected to:  Avnet Virtex II Eval Board


Found 2 device(s) with following DeviceID(s) in the JTAG chain

Legacy PROM XC18V04: 0x5036093 (ID)
Virtex II XCV1000: 0x31028093 (ID)

Configuration/Readback supported for FPGA Device Type - XC2V1000,
Package Type - FG256
```

After communication with the board has been verified, the configuration phase is started. The application creates local copies of the necessary files from the specified design path.

These are namely a bitstream, and a mask file ($(DesignName).msk) that is utilized during verification. The XHWIF interface then initiates the configuration process by making a call to the *SetConfiguration()* native method. The total time displayed below is measured from the instant when the virtex2xhwif class makes the call till when it returns from the method.

```
Creating a local copy of 'seu_tester.bit' and 'seu_tester.msk' in
current directory


Loading file 'seu_tester.bit'

(Re)Configuring device...
Done!!!
(Time taken 30.875 seconds)
```

The ReadbackCommand class of JBits can be used to automatically generate a readback command stream for the Virtex II, noticeably simplifying the readback process. However, using the class methods did not initiate proper readback sequences on the device. It was therefore decided to move the entire implementation to the corresponding native method. The bits that are read back are stored in a file in the work directory. As mentioned earlier, frame one is removed since it contains pad data.

```
Initializing Readback...
Readback successful!!!
(Total time 27.479 seconds)

Read a total of 117024 word(s) (Frame 1 scrapped)
Readback data was written to file 'seu_tester.rb'
```

Once a readback bitstream is available, verifying it against the original .bit file involves some additional steps. Not all bits in the readback data stream correspond to the actual configuration memory. The locations of interest are identified by the mask file. This file is structurally similar to the original bitstream except that the programming bits are replaced with mask data. If the mask bit is a ‘0’, the corresponding bit in the .bit file needs to be compared, otherwise if it is a ‘1’ the bit can be ignored since it relates to user memory or a null memory location. Now the configuration bitstream consists of a file header that stores the design name, device type etc. followed by a list of configuration

commands. The actual frame-programming bits are placed after these two blocks as shown in Figure 42 [38]. These blocks are absent in the readback stream and it only contains an extra pad frame at the beginning. Thus a prerequisite for the comparison process would be to align the actual frame data of all the three files shown in the diagram. Since the pad frame has already been removed from the readback bitstream, and since the mask file has a similar format as the .bit file, we only need to find the frame data offset in the original bitstream. This is done by scanning the commands block for the FDRI write command (see Appendix B) that marks the beginning of a frame data write process. The 32-bit word that follows the command is the total write length in words i.e. 117,130 (0x1C98A). The comparison process begins right after this word and ends when the last bit in the readback stream is reached. Notice that faults in user memory elements have been ignored. It is quite possible though that the detection logic on the FPGA might capture some of these faults. This can be useful in tagging such events when the verification stage fails in detecting any faults and further testing will be needed in this direction.

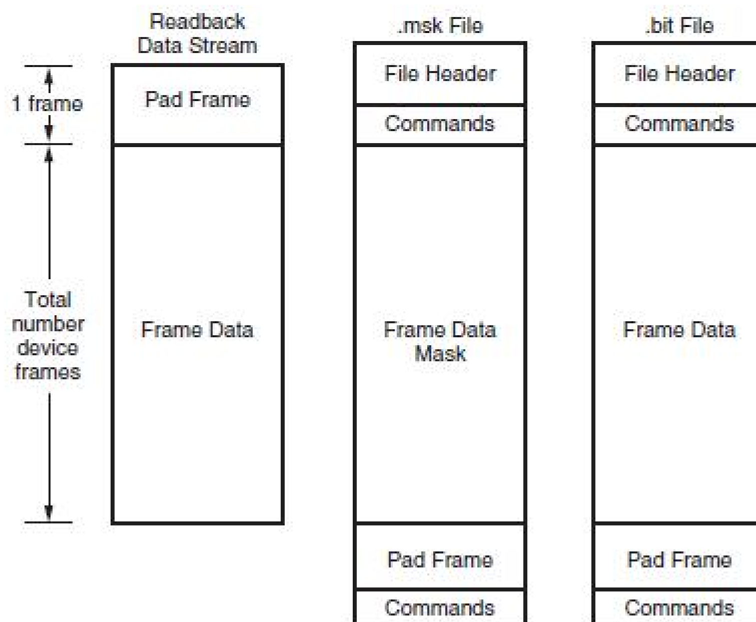


Figure 42: Files used for Bitstream Verification [38]

If a fault is detected the bit is recorded in an error file. More particulars on the file will be given in the next section. The verification process is much faster since the speed is not limited by the parallel port.

```
Verifying Readback...
Verified!!!
(Time taken 0.301 seconds)

Disconnected from:  Avnet Virtex II Eval Board.
```

6.3.2 Phase II: Fault Injection

The fault injection process is inherently an RTR with a faulty bitstream. The FPGA is first loaded with the golden bitstream and the faults are then introduced by modifying individual design related configuration bits. The focus of the current test logic was in detecting SEUs in the CLB flip-flops and therefore the approach would be to target those bits that are specifically associated with these elements. JBits is particularly useful here since it can be used to handle each of the configurable attributes of the CLBs.

To better understand this section, a more detailed diagram of a Virtex II CLB slice is shown in Figure 43 [31]. Either one of the several resources/attributes influencing the operation of slice flip-flops can be altered. For demonstrating the fault injection process, a resource that has been used in the implemented design is first selected. Consider for example the DYMUX element that controls the input to the Y flip-flop. For slice 3 of the CLB element lying at row 23, column 6, this resource is found to be configured to route DY to the D input of the flip-flop. The following method is used to modify this so that BY gets passed instead:

```
jbits.setCLBBits(23,6,DYMUX.CONFIG[3],DYMUX.BYINV);
```

Once the alteration has been done, JBits is then used to generate a faulty bitstream for reconfiguring the device. The active-high *SEU_STATUS* signal then indicates whether the

fault has been captured by the detection logic. For testing, this signal was monitored by routing it to a status LED.

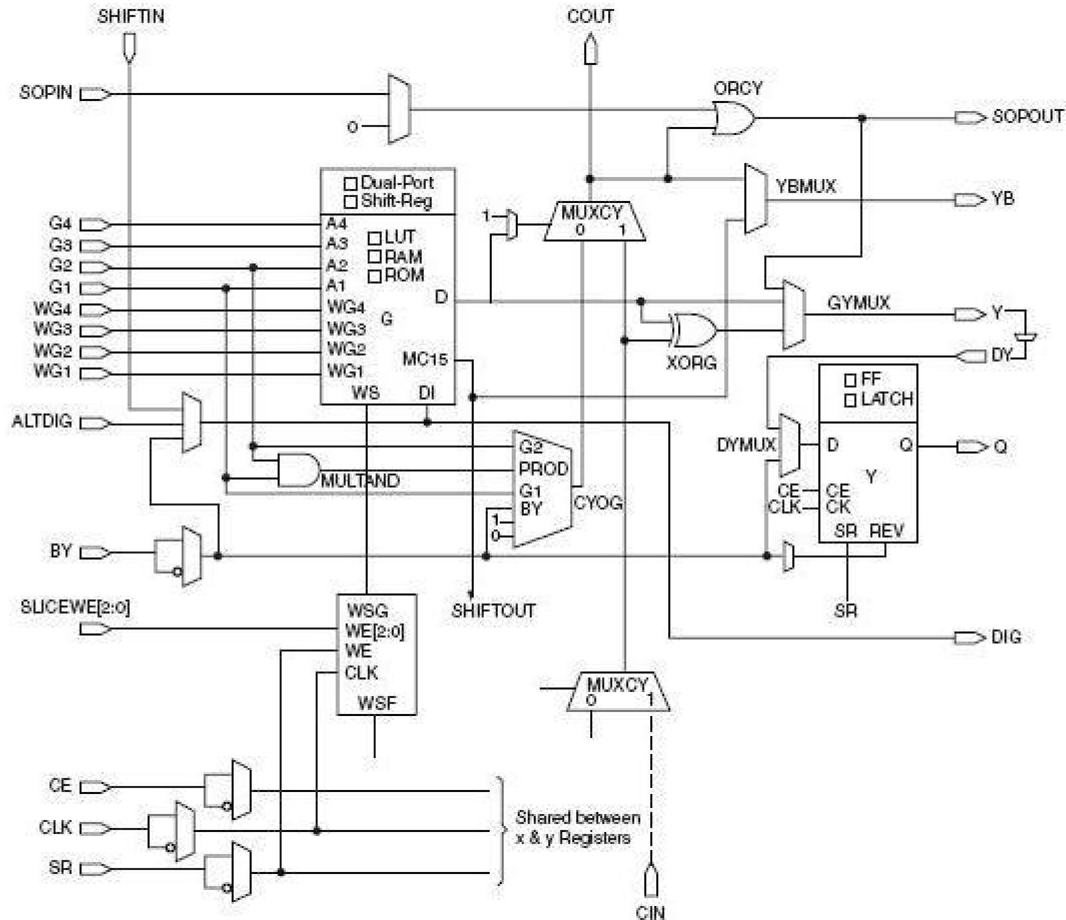


Figure 43: A Detailed Look at the CLB Slice (Top Half) [31]

In order to determine if the incorrect bit is properly identified a readback/verification process is started next. Errors that are detected are logged by the program in an error file ($(DesignName).err).

```
Verifying Readback...
Found a total of 1 error(s)
(Total time 0.32 seconds)

Error bit offsets were recorded in file 'seu_tester.err'
```

The file mainly logs the total errors and their bit offsets with respect to the start of the readback file. A few additional details such as the design name, device type, etc. are also provided in a header for future reference. The offsets are stored as 32-bit words as shown in Figure 44, where the final word block shows the offset of the detected error bit in hexadecimal. Assuming bit 1 at MSB, this can be interpreted as bit 5 of the byte at offset 0x00010CFD in the readback stream. Now the frame data offset calculated for the current design bitstream was 0x9D (in bytes). Hence this translates to byte-offset 0x00010D9A in the configuration bitstream. This is found to be consistent when matched with the altered configuration bitstream. Bit offsets were mainly logged to aid in testing and can be omitted in the final implementation to reduce memory consumption.

```
seu_tester.err

Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  44 65 73 69 67 6E 3A 09 09 09 09 09 73 65 75 5F  Design:.....seu_
00000010  74 65 73 74 65 72 0A 41 72 63 68 69 74 65 63 74  tester.Architect
00000020  75 72 65 3A 09 09 09 09 56 69 72 74 65 78 20 49  ure:....Virtex I
00000030  49 0A 50 61 72 74 3A 09 09 09 09 09 09 58 43 32  I.Part:......XC2
00000040  56 31 30 30 30 46 47 32 35 36 0A 46 69 6C 65 20  V1000FG256.File 
00000050  54 79 70 65 3A 09 09 09 09 09 52 65 61 64 62 61  Type:.....Readba
00000060  63 6B 20 56 65 72 69 66 79 0A 46 69 6C 65 20 43  ck Verify.File C
00000070  72 65 61 74 65 64 3A 09 09 09 09 53 75 6E 2C 20  reated:....Sun, 
00000080  4A 61 6E 20 30 36 20 32 30 30 38 20 31 36 3A 32  Jan 06 2008 16:2
00000090  38 3A 32 35 0A 53 69 7A 65 6F 66 20 65 61 63 68  8:25.Sizeof each
000000A0  20 65 72 72 6F 72 20 6F 66 66 73 65 74 20 28 62   error offset (b
000000B0  79 74 65 73 29 3A 09 34 0A 54 6F 74 61 6C 20 45  ytes):.4.Total E
000000C0  72 72 6F 72 73 3A 09 09 09 09 31 0A 0A 00 08 67  rrors:....1....g
000000D0  EC                                               i
```

Figure 44: Error File Structure

Table 4: Fault-Injection Test Results

Attribute	Original Value	Modified Value	Detected by Logic	Detected by Readback Verification	Effect on Design	Error Persists
DXMUX	BXINV	DX	Yes	Yes	(i)	Yes
DYMUX	DY	BYINV	Yes	Yes	(i)	Yes
BXINV	BX	BX_B	Yes	Yes	(iii)	No
BYINV	BY	BY_B	Yes	Yes	(ii)	Yes
CLKINV	CLK	CLK_B	Yes	Yes	(iii)	No
CEINV	CE	CE _B	Yes	Yes	(i)	Yes
SRINV	SR	SR_B	Yes	Yes	(i)	Yes
SRFFMUX*	ON	OFF	No	No	-	-
REVUSED	ON	OFF	Yes**	Yes	(iv)	No
SYNC_ATTR	ASYNC	SYNC	Yes**	Yes	(iv)	No
FFX	FF	LATCH	Yes	Yes	(iii)	No
FFY	FF	LATCH	Yes	Yes	(iv)	No
FFX_SR_ATTR	SRLOW	SRHIGH	Yes	Yes	(iv)	No
FFY_SR_ATTR	SRLOW	SRHIGH	Yes	Yes	(iv)	No
FFX_INIT_ATTR***	INIT0	INIT1	Yes	No	(iv)	No
FFY_INIT_ATTR***	INIT0	INIT1	Yes	No	(iv)	No

* JBits did not modify any part of the bitstream when this attribute was changed.

** Depending on current state of flip-flop and position of the set (reset) pulse

*** This attribute resides in user memory of the FPGA bitstream

Notes:

(i) Containing SFR output stuck-at-0

(ii) Containing SFR output stuck-at-1

(iii) Corresponding signal out of phase

(iv) None (error effect most likely transient)

6.4 Results

The fault-injection process was followed for several other attributes that were expected to directly influence the flip-flop behavior. A more exhaustive approach might be conformed to in future where other elements are also considered, such as an LUT indirectly driving an input. Each of the test elements documented in Table 4 was carried out for five randomly selected CLB flip-flops for consistency. The attribute names were kept the same as their corresponding names in the Xilinx FPGA Editor for easier identification, while the values are from the corresponding JBits classes. Only resource based faults were tested in the present setup. However a similar approach can be followed for observing the effects of bit modifications in the resource routing. All the faults in the table were rectifiable through a reconfiguration with the 'golden' bitstream.

Each of the attributes corresponds to a specific CLB resource/configuration setting. For instance, BYINV is identified as the MUX that immediately follows the BY input seen in Figure 43. When set as BY, the true logic of the signal is passed while BY_B passes its inverse. This signal is used to force the storage element into a state opposite to that of the SR (set-reset) signal. In the current design, SR has been configured as the global reset signal, hence BY functions as the global set for the CLB flip-flops. This explains the observed effect shown in Table 4 when this resource is modified. This signal is inactive (logic '0') for all flip-flops except the one where the fault is injected. Hence, the faulty element is always in a 'set' state resulting in a constant logic '1' at the output of the corresponding SFR. 'Error Persists' indicates whether the *SEU_STATUS* signal continues to remain active after the error register has been cleared. Faults in several attributes will remain dormant unless activated by certain signals. For example FFY_SR_ATTR decides whether a logic '1' on the SR input will reset (SRLOW) or set (SRHIGH) the flip-flop. Detecting faults through the logic therefore requires the global-reset (set) signal to be pulsed. More information on the CLB resources (attributes) mentioned in the result table can be found in [31].

The ideal case of error detection by logic and by readback verification was found to be true for most of the test elements. However one exception in particular, the SRFFMUX

attribute, could not be tested. Using JBits methods to modify this element did not appear to manifest itself in the generated bitstream. While this could be a possible bug in the current JBits package, it is also possible that this element is not present or configurable for the Virtex II device used here since, although it appears in the FPGA Editor tool, it is not mentioned in the datasheet.

The REVUSED attribute is associated with the MUX element that routes the BY (BY_B) signal to the REV (reverse SR) input of the slice flip-flops. In other words, it decides whether the set input is enabled or disabled for a particular flip-flop. Therefore, to test the consequence once this attribute has been modified for the targeted element, one needs to pulse the global-set signal. This results in all flip-flops, except the one in question, to be 'set'. Now, if the flip-flop is already at logic '1' this will have no effect, and hence will not be detected by the logic. Another constraint placed on the detection is the position at which the asynchronous 'set' pulse occurs. Since the flip-flop is updated on a positive clock edge, the fault might get overwritten by the logic of the previous stage before it gets a chance to propagate to subsequent stages. Detection of faults related to the SYNC_ATTR attribute also depends on a more or less similar set of conditions. This attribute decides whether the reset and set inputs to the flip-flop are synchronous or asynchronous.

The FFX_INIT_ATTR and FFY_INIT_ATTR attributes are special cases since they correspond to user memory locations in the configuration bitstream. Thus it was not possible to detect them through readback verification as was explained earlier. However the fact that faults associated with them are detected by the logic proves the assumption made previously, namely that it is possible that the detection logic might be capable of capturing faults in user memory locations.

Chapter 7

Conclusion

7.1 Summary

This thesis described the SEU problem mainly from the FPGA perspective. SEU effects on COTS FPGAs need to be studied in detail if the capability of using such devices in space is to be explored.

The JBits-based testing approach that was followed required the development of an XHWIF interface for the FPGA board. This interface was created and tested for any possible functional errors, and at present appears to be working correctly. However an issue is sometimes known to arise when a configuration is immediately followed by readback verification i.e. when the Java test program is invoked with the –cvr option. The readback bitstream seems to miss one bit during the readback that causes it to go out of sync when compared with the configuration bitstream. Although only a single bit is missed, this causes several errors during the verification phase since the alignment between the files to be compared is disturbed. Clearing the configuration memory and/or a power-cycle seems to fix this. It is possible that a brief interval needs to pass after a configuration before a readback command is issued to the FPGA and this needs to be further looked into.

JBits was successfully used to simulate SEU faults in CLB flip-flops by modifying the resource parameters and reconfiguring the device with the faulty bitstream. Since each of the parameters mapped to single bits in the bitstream this could be assumed as actual bit-flips occurring when the device is in a high-radiation setting. Most of the faults were detected by both the redundant detection logic as well as during readback verification. Once faults were detected, JBits was used to correct them through a full configuration scrubbing.

7.2 System Implementation

The proposed implementation of the final system is shown in Figure 45. Among the three commonly used radiation detectors that were studied in this paper, the end-window GM tube would be a suitable option. In particular, the sensor used in the MEROPE payload would be a good choice since, being a CubeSat their design constraints would be similar to our own. If possible it would be ideal to have an additional RADFET, as it would also give a measure of the cumulative dose absorbed along with the particle flux. The detector module will require additional components such as a high-voltage source, amplifier etc. as were described previously. Also, the pulses from the sensor can either be counted by the microcontroller by means of an interrupt, or through an additional counting device.

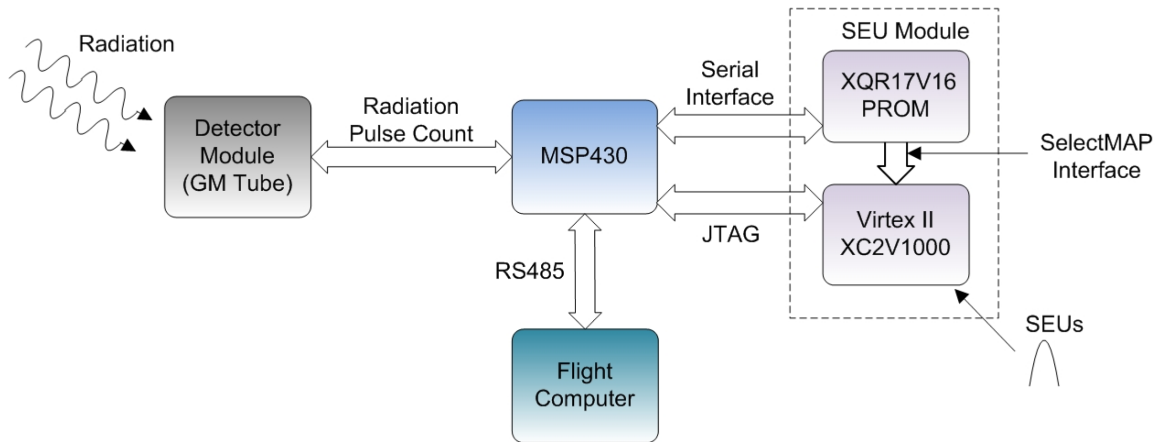


Figure 45: Proposed System Implementation

The purpose of using JBits in the current work was mainly as a source of simulating and correcting faults. Although it has been ported to embedded platforms, the high memory requirement of Java might be a limitation. Since the final implementation will only require correcting faults in the FPGA, the program can be created in a C/C++ setting. The code forming the XHWIF native methods can be reused for this purpose to create a standalone C/C++ application. Apart from managing the configuration and readback on the FPGA, the program would also be responsible for carrying out the bitstream verification. This approach will reduce the memory consumption and increase the range of possible microcontroller options. The MSP430 microcontroller from Texas

Instruments™ could be used as the CPU, since prior experience with this processor has already been acquired through the KySat1 project. The link between the Detection Unit CPU and the Flight Computer can be provided through a serial RS485 interface. Selection of the PROM will be simplified if the program size is limited so that it can completely reside in the microcontroller memory. In this case the PROM will only be required to hold data pertaining to the FPGA and the same PROM that was provided in the Avnet board (XC18V04) can be used. A rad-hardened version, the XQR18V04 is also available, and it would be advisable to use this in order to prevent possible SEU occurrences. Depending on the process employed for readback verification at least two such devices would be required, since the mask file also needs to be stored. Another option would be to use the XQR17V16 that has a higher storage capacity. The processor can communicate with the PROM via the Master Serial mode.

7.3 Future Directions

The testing in this thesis was carried out using software simulated faults. However, the device that is selected for the final payload implementation may need to be tested in an actual particle accelerator facility before it can be deemed suitable for orbit. Apart from testing its susceptibility to SEUs, analysis also needs to be carried out for latch-ups and SEFIs. Another parameter that needs to be considered is the device TID tolerance, since their effects can severely compromise the device lifespan. Several experiments have already been carried out on the Virtex II. For instance, an analysis of the SEU susceptibility of the FPGA was carried out at the Texas A&M Cyclotron Facility [56]. The setup tested static configuration upsets through heavy-ion irradiations on a XC2V1000 device, the same one used in this work. The study found that for the elements probed, the saturation cross-section was approx. $5x10^{-8}$ cm^2/bit, with upset rates of $4.4x10^{-7}$/bit-day for the configuration memory and $1.1x10^{-6}$/bit-day for the block SelectRAM. Tests such as these can be used as references when devising the test setup in our case.

The elements in the current design including the comparator logic were all operated at the oscillator frequency (40 MHz). An alternate option might be to run the SFRs and pattern generators at lower frequencies while keeping the speed of the detection logic high. This configuration was not used since no changes were perceived to the detection through logic or readback verification results in Table 4. Also, several tests had already been carried out and it was desired to keep the results consistent. This might be employed in a future configuration and a possible change could be on its observed effect on the design and/or in making some of the errors in the table persistent.

At present, the parallel port limits TCK to 260 KHz. However, JTAG permits a maximum TCK frequency of 33 MHz [31], and hence the configuration and readback process times can be considerably reduced. Another option is to use partial reconfiguration and/or readback. This was not used in the current implementation since it required additional design flows to be followed. However, using it can further reduce the process completion times to the fraction taken to access only selected frames instead of the entire bitstream. The test environment created here demonstrated fault injection and correction in CLB flip-flops. A similar approach can be used to probe other resources such as the LUTs or block SelectRAM structures, in which case a design that maximizes the utilization of that particular resource will need to be developed.

Appendix A

Virtex II JTAG Registers and Supported Instructions

Virtex II JTAG Instructions

The instructions that are provided in the Virtex II are given in Table 5. The size of the instruction register (IR) also allows support for the relatively new IEEE 1532 standard for In-System Configuration (ISC) that is based on JTAG.

Table 5: Virtex II JTAG Instructions

Instruction	Instruction Binary Code (5:0)	Description
JSTART	001100	Used to initialize the device startup sequence.
JSHUTDOWN	001101	Initiates the shutdown sequence in the Virtex II
CFG_IN	000101	Provides access to the configuration register for writing (configuration).
CFG_OUT	000100	Provides access to the configuration register for reading (readback).
BYPASS	111111	Used to bypass the device by enabling the bypass register
IDCODE	001001	Allows the device ID to be shifted out through TDO
USERCODE	001000	Allows the user specified code to be shifted out through TDO
JPROG_B	001011	Used to clear the device configuration memory (same function as *PROG_B*)
INTEST	000111	Enables the boundary-scan INTEST operation.
EXTEST	000000	Enables the boundary-scan EXTEST operation.
SAMPLE	000001	Enables boundary-scan SAMPLE operation
HIGHZ	001010	3-states the output pins while enabling the bypass register
USER1	000010	Provides access to the USER1 register

USER2	000011	Provides access to the USER2 register
RESERVED	All other codes	Reserved instructions for Xilinx

Virtex II JTAG Register Set

The Virtex II features all the mandatory registers required by the IEEE 1149.1 Boundary-Scan standard as well as some additional ones as given in [38]. These will be listed here for reference along with a brief explanation.

Table 6: JTAG Registers for the Virtex II

Register	Register Length (bit(s))
Instruction Register	6
Bypass Register	1
Identification Register	32
Configuration Register	64
Boundary-Scan Register	3 per I/O
USERCODE Register	32

- **Instruction Register:** This register contains the current instruction opcode. It also captures the current device status during the Capture-IR state as given in the device Boundary Scan Description Language (BSDL) file. The significance of the bits is given below:

XXXX01

Bit 5 : '1' when *DONE* pin is released (part of startup sequence)

Bit 4 : '1' if house-cleaning is complete

Bits 3, 2 : Used for ISC

- **Bypass Register:** It is enabled after a BYPASS instruction. It acts as a buffer while shifting out TDI signals to TDO. The register is reset to zero during the Capture-IR state.

- **Identification Register:** This register is used to store the device ID as defined by the JTAG standard. The format for the Virtex II is given as (31:0):

  ```
  vvvv:fffffff:aaaaaaaaa:ccccccccccc1
  ```

  Where,

  v is the revision code

  f is the 7-bit family code = 0001000 (0x08) for the Virtex II

  a is the number of array rows in the part expressed in 9 bits

  for the XC2V1000 this is equal to 40 rows (0x028)

  c is the company code = 00001001001 = 0x049

  The IDCODE for the XC2V1000 can therefore be formed as follows:

  ```
  vvvv: ffff fff:a aaaa aaaa: cccc cccc ccc1
  vvvv: 0001 000:0 0010 1000: 0000 1001 0011   (0xb)
   <v>:    1     0    2    8     0    9    3   (0xh)
  ```

- **Configuration Register:** This JTAG register enables access to the configuration bus after a CFG_IN or CFG_OUT instruction.

- **Boundary-Scan Register:** It is essentially utilized for testing in combination with instructions such as INTEST and EXTEST.

- **USERCODE Register:** It is used to hold a user-defined code that can be something meaningful such as the current version of the design, date when configured etc. The code is written to the bitstream when it is generated and becomes valid after configuration.

Appendix B

Virtex II Selected Configuration Specifics

XC2V1000 Device Information

- Frame Count: 1104
- Frame Length in 32-Bit Words: 106
- Total Configuration Bits: 3,744,768
- Default Bitstream Size: 4,082,592 (Configuration Bits + Overhead)
- CLB Arrangement:
    - Total Slices: 5,120
    - Array (row x col): 40 x 32

Packet Formats

The Virtex II has two packet types for managing the configuration process through configuration registers. If the register access necessitates a large word count, a Type 1 header can be followed by a Type 2.

Type			WR	RD	Register Address														RSVD		Word Count										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0	0	X	X	X	X	X	X	X	X	X	X	X

(a) Type 1

Type			WR	RD	Word Count																										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0	0	X	X	X	X	X	X	X	X	X	X	X

(b) Type 2

Figure 46: Packet Header Types

Configuration Registers

The configuration registers included in the Virtex II are given below. Descriptions are expanded for selected registers (refer to [38] for a comprehensive explanation)

Table 7: Virtex II Configuration Register Set

Register	Read	Write	Description
FAR	Y	Y	Frame Address Register
FDRI	N	Y	Frame Data Input Register (for configuration)
FDRO	Y	N	Frame Data Output Register (for readback)
FLR	Y	Y	Frame Length Register
CRC	Y	Y	Cyclic Redundancy Check Register
CMD	Y	Y	Command Register
STAT	Y	N	Status Register
COR	Y	Y	Configuration Option Register
CTL	Y	Y	Control Register
MASK	Y	Y	Masking Register for CTL
IDCODE	Y	Y	Device ID Code Register
LOUT	N	Y	Legacy Output Register (daisy-chain DOUT)
MFWR	N	Y	Multiple Frame Write Register
KEY	N	Y	Initial Key Address Register
CBC	N	Y	Cipher Block Chaining Register

- **FAR:** The configuration frame is selected based on the address stored in this register. FDRI and FDRO processes require the starting frame address to be set in order to access the specific frames. The value in FAR auto-increments when total words as specified by FLR have been written/read out. The current command in the CMD register is executed each time FAR is written with a new value.

- **FDRI:** Frame data written to this register is used to configure the FPGA frame as specified by FAR. The frame data is pipelined through the frame buffer in a manner such that the first frame is shifted into the configuration memory while the second frame is being shifted into the frame buffer. Hence, in order to load the last frame, an extra pad frame needs to be written to FDRI at the end. The word following an FDRI write is interpreted by the FPGA as an AutoCRC.

- **FDRO:** This register is used to read the contents of the configuration memory. The number of frames to be read needs to be specified in the packet header. This process is also pipelined, wherein the first frame is shifted out of the buffer while the second one is being read from the configuration memory. As a result the actual data begins from the second frame, while the first frame consists of unneeded pad data. This process does not produce a CRC value or utilize AutoCRC.

- **FLR:** The frame length needs to be specified through a write to this register before an FDRI or FDRO process can start. The value written to the FLR should be in the form of (Actual Frame Length - 1) words e.g. for the XC2V1000, the value is 0x00000069 words.

- **CRC:** 16-bit CRCs are used to check for possible errors in data writes. A data write to any register (except LOUT) initiates a CRC calculation based on the data and address bits of the register. A write operation to the CRC register causes a comparison between the register value and the calculated value. If discrepancies are detected, an ERROR state is entered by enabling the CRC_ERROR bit in the STAT register and by holding the *INIT_B* pin low. CRCs can be carried out by writing a pre-calculated value to the CRC register using a bitstream command or by using ‘AutoCRC’, wherein the last word packet write to the FDRI is automatically read as a CRC value.

- **CMD:** This register is used to manage the configuration process through specific command codes. The command is activated as soon it is written to this register.

Table 8: CMD Register Codes

Command	Code	Description
WCFG	0001	Write Configuration Data. Required before an FDRI write operation
RCFG	0100	Read Configuration Data. Required before an FDRO read operation
RCRC	0111	Reset CRC (register).
AGHIGH	1000	Assert GHIGH_B. Used during shutdown reconfiguration and readback to place all interconnect in a high-impedance state to prevent data contention.
LFRM	0011	Last Frame. Used to enable interconnects by disabling the GHIGH_B signal
START	0101	Begin Startup Sequence. Startup sequence begins after a successful CRC check and a DESYNCH command
SHUTDOWN	1011	Begin Shutdown Sequence. Shutdown sequence begins after a successful CRC check or an RCRC command
GRESTORE	1010	Pulse GRESTORE signal. Used to set/reset (based on user configuration) the CLB and IOB flip-flops
GCAPTURE	1100	Pulse GCAPTURE signal. Used to load the capture cells with current register states
RCAP	0110	Reset Capture. Resets CAPTURE signal after a readback-capture in single-shot mode
MFWR	0010	Multiple Frame Write Register. Used to write a single frame to multiple frame addresses.
SWITCH	1001	Switch CCLK Frequency. Updates the frequency of the Master CCLK to the value specified by the OSCFSEL bits in the COR
DESYNCH	1101	Reset DALIGN Signal. Used at end of configuration to de-synchronize the device following which, values on configuration data pins are ignored.

- **STAT:** This register can be read using JTAG or SelectMAP and is used to check the status of several global signals.

																BAD_KEY_SEQ	DEC_ERROR	ID_ERROR	DONE	INIT	MODE			GHIGH_B	GWE	GTS_CFG_B	IN_ERROR	DCI_MATCH	DCM_LOCK	PART_SECURED	CRC_ERROR
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Figure 47: The STAT Register

Table 9: Description of STAT Register Bits

Bit	Name	Description
0	CRC_ERROR	CRC status 0: No CRC error 1: CRC error
1	PART_SECURED	Triple-DES decryptor status 0: Decryptor security not set 1: Decryptor security set
2	DCM_LOCK	Logical AND function of all DCM locked signals. Unused signals are ‘1’ 0: DCMs not locked 1: DCMs locked
3	DCI_MATCH	Logical AND function of all MATCH signals. Each I/O bank has one MATCH signal. If all DCI I/Os un-instantiated in a give bank its MATCH signal is ‘1’ 0: DCI not matched 1: DCI matched
4	IN_ERROR	Indicates input data being clocked in too fast, e.g. when CCLK frequency exceeds $F_{CC_SELECTMAP}$ in SelectMAP mode 0: No legacy input error 1: Legacy input error

5	GTS_CFG_B	GTS_CFG_B signal status 0: All I/Os in high-Z state 1: I/Os not placed in high-Z state by this signal
6	GWE	Global Write Enable signal status 0: All FFs and BRAMs are write disabled 1: All FFs and BRAMs are write enabled
7	GHIGH_B	GHIGH_B signal status 0: GHIGH_B is enabled 1: GHIGH_B is disabled
10:8	MODE	MODE pins (M2:M0) status
11	INIT	Reflects actual level on *INIT* pin 0: *INIT* pin at logic '0' 1: *INIT* pin at logic '1'
12	DONE	Reflects actual level on *DONE* pin. If pin is released by the device but held low externally this is displayed as low 0: *DONE* pin at logic '0' 1: *DONE* pin at logic '1'
13	ID_ERROR	Attempted to write to FDRI without successful IDCODE check 0: No ID_ERROR 1: ID_ERROR. (Assert PROG to rectify)
14	DEC_ERROR	Write to FDRI issued before or after decrypt instruction 0: No DEC_ERROR 1: DEC_ERROR. (Assert PROG to rectify)
15	BAD_KEY_SEQ	Indicates if encryption keys sent in wrong order 0: No decryptor key sequence error 1: Decryptor keys not used in proper sequence

Glossary

Glossary of significant acronyms used in this thesis.

API	Application Programming Interface
BSDL	Boundary Scan Description Language
CLB	Configurable Logic Block
CMD	Command Register
CME	Coronal Mass Ejection
CMOS	Complementary Metal Oxide Semiconductor
COTS	Commercial-Off-The-Shelf
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DCM	Digital Clock Manager
DIP	Dual In-Line Package
DLL	Dynamic Linked Library
DR	Data Register
EEPROM	Electrically Erasable PROM
FAR	Frame Address Register
FDRI	Frame Data Input Register
FDRO	Frame Data Output Register
FET	Field Effect Transistor
FF	Flip-Flop
FPGA	Field Programmable Gate Array
GCR	Galactic Cosmic Ray
GM	Geiger-Müller
HDL	Hardware Description Language
ID	Identification
IDE	Integrated Development Environment
IOB	Input Output Block
IR	Instruction Register
JNI	Java Native Interface
JTAG	Joint Test Action Group
JVM	Java Virtual Machine
KySat	Kentucky Satellite
LEO	Low Earth Orbit
LET	Linear Energy Transfer
LFRM	Last Frame
LSB	Least Significant Bit
LUT	Look-Up-Table
MOSFET	Metal Oxide Semiconductor FET
MSB	Most Significant Bit

MUX	Multiplexer
NASA	National Aeronautics and Space Administration
PIP	Programmable Interconnect Point
PROM	Programmable ROM
RADFET	Radiation Sensitive FET
RAM	Random Access Memory
ROM	Read Only Memory
RTI	Run-Test-Idle
RTR	Run-Time Reconfiguration
SEB	Single Event Burnout
SEE	Single Event Effect
SEFI	Single Event Functional Interrupt
SEGR	Single Event Gate Rupture
SEL	Single Event Latch-Up
SET	Single Event Transient
SEU	Single Event Upset
SFR	Shift Register
SHE	Single Event Hard Error
SOI	Silicon-On-Insulator
SOS	Silicon-On-Sapphire
SR	Set-Reset
SRAM	Static RAM
TAP	Test Access Port
TCK	Test Clock
TDI	Test Data In
TDO	Test Data Out
TID	Total Ionizing Dose
TLR	Test-Logic-Reset
TMR	Triple Modular Redundancy
TMS	Test Mode Select
UHF	Ultra High Frequency
VHF	Very High Frequency
XHWIF	Xilinx Hardware Interface

References

[1] K. Bedingfield, R. Leach and M. Alexander, "*Spacecraft System Failures and Anomalies Attributed to the Natural Space Environment*", NASA Reference Publication 1390, Marshall Space Flight Center, Aug 1996.

[2] M. Napier, J. Moore, S. Rezgui, C. Carmichael, J. George and G. Swift, "*Single Event Effect (SEE) Analysis, Test & Mitigation of the Xilinx Virtex-II Input Output Block (IOB)*", http://klabs.org/mapld04/abstracts/napier_a.doc, Nov 2007.

[3] Xilinx Inc., "*Virtex-5 Family Overview LX, LXT, and SXT Platforms*", DS100 (v3.2) http://www.xilinx.com/support/documentation/data_sheets/ds100.pdf, Sep 4, 2007.

[4] Xilinx Inc., "*2001 Annual Report*", http://media.corporate-ir.net/media_files/irol/75/75919/reports/xlnx_ar2001.pdf, Dec 31, 2001.

[5] P. Graham, M. Caffrey, M. Wirthlin, E. Johnson and N. Rollins, "*SEU Mitigation for Half-Latches in Xilinx Virtex FPGAs*", IEEE Trans. Nucl. Sci., vol. 50, no. 6, pp. 2139-2146, Dec 2003.

[6] MEROPE Payload, http://www.ssel.montana.edu/merope/subsystems/subsys_payload.html, Nov 2007.

[7] S. Leon, "*A Self-Reconfiguring Platform for Embedded Systems*", Master's Thesis, Virginia Tech., Aug 2001.

[8] R. Sheldon, "*The Outer Van Allen Radiation Belt and the Radiation Belt Storms Probe Mission*", http://gammaray.nsstc.nasa.gov/colloquia/abstracts_spring06/rsheldon.html, Feb 10, 2006.

[9] N. Short, "*Geophysical Remote Sensing*", Remote Sensing Tutorial, http://rst.gsfc.nasa.gov/Intro/Part2_1a.html, Nov 2007.

[10] J. Barth, "*Space & Atmospheric Environments*", NASA/Goddard Space Flight Center, Flight Electronics Branch/Code 561, http://eos.gsfc.nasa.gov/eos-ll/docs/barth-2-02.pdf, Feb 14, 2002.

[11] J. Vette, "*AE/AP Trapped Particle Flux Maps 1966-1980*", http://modelweb.gsfc.nasa.gov/magnetos/aeap.html, Nov 2007.

[12] H. Hoerlin, "*United States High-Altitude Test Experiences: A Review Emphasizing the Impact on the Environment*", Los Alamos Scientific Laboratory, NM, Oct 1976.

[13] Space History Division, "*NASM Space Artifacts Ariel 1*", Smithsonian, National Air and Space Museum, http://airandspace.si.edu/spacecraft/SS-ariel1.htm, Aug 18, 1999.

[14] W. Hess, "*The Effects of High Altitude Explosions*", NASA TN D-2402, NASA Goddard, MD, Sep 1964.

[15] NASA, "*NASA – NSSDC – Spacecraft – Details: Telstar 1*", http://nssdc.gsfc.nasa.gov/nmc/masterCatalog.do?sc=1962-029A, Nov 2007.

[16] J. Vette, "*The NASA/National Space Science Data Center Trapped Radiation Environment Model Program (1964-1991)*", NSSDC/WDC-A-R&S 91-29, Nov 1991.

[17] K. LaBel, M. Gates, J. Barth, A. Johnston and P. Marshall, "*Single Event Effect Criticality Analysis (SEECA)*", Sponsored by NASA Headquarters/Code QW, http://radhome.gsfc.nasa.gov/radhome/papers/seecai.htm, Feb 15, 1996.

[18] D. Bird, S. Corbato, H. Dai, B. Dawson, J. Elbert, T. Gaiser et al., "*Evidence for Correlated Changes in the Spectrum and Composition of Cosmic Rays at Extremely High Energies*", Phys. Rev. Letters, vol.71, no.21, pp.3401-3404, Nov 22, 1993.

[19] "*2B65 - Space Science, Instrumentation and Techniques: 1.4 Photo-conductive detectors*", Mullard Space Science Laboratory, UCL, U.K., http://www.mssl.ucl.ac.uk/~gbr/index.html, Nov 2007.

[20] S. Ghoshal, "*Nuclear Physics*", S. Chand, 1997.

[21] Y. Hosono, Sjafruddin, T. Iguchi and M. Nakazawa "*Fast Neutron Detector using PIN-Type Silicon Photodiode*", Nucl. Instrum. & Meth. in Phys. Res., vol. 361, issue 3, pp.554-557, Jul 1995.

[22] "*Detectors – ESRF*", European Synchrotron Radiation Facility (ESRF), http://www.esrf.eu/UsersAndScience/Experiments/HRRS/ID28/BeamlineLayout/EH1/Spectrometer/Detectors, Sep 13, 2006.

[23] "*RADFET Technical Information*", Tyndall National Institute, http://www.tyndall.ie/projects/radfets/tech.html, Nov 2007.

[24] G. Ensell, A. Holmes-Siedle and L. Adams, "*Thick Oxide pMOSFET Dosimeters for High Energy Radiation*", Nucl. Instrum. & Meth. in Phys. Res., vol. 269, issue 3, pp. 655-658, Jun 1998.

[25] S. Stanic, Y. Asano, H. Ishino, A. Igarashi, S. Iwaida, Y. Nakano et al., "*Radiation Monitoring in Mrad Range using Radiation-Sensing Field-Effect Transistors*", Nucl. Instrum. & Meth. in Phys. Res., vol. 545, pp. 252-260, 2005.

[26] H. Wong and J. Li, "*A Pilot Experiment with Reactor Neutrinos in Taiwan*", Nucl. Phys. B Proc. Suppl., vol. 77, issue 1-3, pp 177-181, May 1999.

[27] JEDEC Standard, "*JEDEC Dictionary of Terms for Solid State Technology*", JEDEC Solid State Technology Association, JESD88B – 3rd Ed., May 2006.

[28] K. Holbert, "*EEE 460 - Nuclear Concepts*", http://www.fulton.asu.edu/~holbert/eee460/eee460.html, May 1, 2007.

[29] "*Miscellaneous ROSAT Images: The South Atlantic Anomaly*", http://heasarc.gsfc.nasa.gov/docs/rosat/gallery/misc_saad.html, Aug 16, 2002.

[30] T. Karnik and P. Hazucha, "*Characterization of soft errors caused by single event upsets in CMOS processes*" IEEE Trans. on Dependable and Secure Computing, vol. 1, issue 2, pp. 128- 143, Apr-Jun 2004.

[31] Xilinx Inc., "*Virtex-II Platform FPGAs: Complete Data Sheet*", DS031 (v3.4) http://www.xilinx.com/support/documentation/data_sheets/ds031.pdf, March 1, 2005.

[32] A. Noore, "*An improved SRAM cell design for tolerating radiation-induced single-event effects*", IEICE Electronics Express, vol. 4, no. 3, pp. 100-105, Feb 2007.

[33] M. Violante, L. Sterpone, M. Ceschia, D. Bortolato, P. Bernardi, M. Sonza Reorda et al., "*Simulation-Based Analysis of SEU Effects in SRAM-Based FPGAs*", IEEE Trans. on Nuclear Science, vol. 51, no. 6, pp. 3354-3359, Dec 2004.

[34] M. Bellato, P. Bernardi, D. Bortolato, A. Candelori, M. Cerchia, A. Paccagnella et al., "*Evaluating the effects of SEUs affecting the configuration memory of an SRAM-based FPGA*", IEEE Design Automation and Test in Europe, pp. 188-193, 2004.

[35] N. Rollins, M. Wirthlin, M. Caffrey and P. Graham, "*Reliability of Programmable Input/Output Pins in the Presence of Configuration Upsets*", MAPLD '02 Laurel MD, 10-12 Sep 2002.

[36] C. Carmichael, "*Triple Module Redundancy Design Techniques for Virtex FPGAs*", Xilinx Tech. Report, XAPP197 (v1.0.1), Jul 6, 2006.

[37] D. Mavis and P. Eaton, "*Temporally Redundant Latch for Preventing Single Event Disruptions in Sequential Integrated Circuits*", Mission Research Corporation Tech. Report P8111.29, Oct 8, 1998.

[38] Xilinx Inc., "*Virtex II Platform FPGA User Guide*", UG002 (v2.1), http://www.xilinx.com/support/documentation/user_guides/ug002.pdf, Mar 28, 2007.

[39] C. Carmichael, "*Correcting Single-Event Upsets through Virtex Partial Configuration*", Xilinx App. Note (v1.0) XAPP216, Jun 1, 2000.

[40] G. Chandler, D. McClure, S. Hishmeh, J. Lumpp, Jr., J. Carter, B. Malphrus et al., "*Development of an Off-the-Shelf Bus for Small Satellites*", IEEE Aerospace Conference, Mar 2007.

[41] S. Guccione, D. Levi and P. Sundararajan, "*JBits: A Java-based Interface for Reconfigurable Computing*", 2nd Annual Military and Aerospace Applications of Programmable Devices and Technologies Conference (MAPLD), 1999.

[42] Avnet Inc., "*Xilinx Virtex-II Evaluation Kit - User's Guide*", Rev 2.0, Jun 18, 2003.

[43] Xilinx Inc., "*JTAG Programmer Guide*", http://toolbox.xilinx.com/docsan/3_1i/pdf/docs/jtg/jtg.pdf, Nov 2007.

[44] NASA Goddard Space Flight Center, "*A New Radiation Belt Model: AE-9/AP-9*", http://lws-set.gsfc.nasa.gov/Documents/NewRadiationBeltModelAEP9.pdf, Nov 2007.

[45] E. Fuller, M. Caffrey, A. Salazar, C. Carmichael and J. Fabula, "*Radiation Characterization and SEU Mitigation of the Virtex FPGA for Space-Based Reconfigurable Computing*", Proc. of the IEEE Nucl. and Space Radiation Effects Conference (NSREC), Nevada, Jul 2000.

[46] "*Ancillary Description Writer's Guide: Orbit*", NASA Global Change Master Directory, http://gcmd.nasa.gov/User/suppguide/, 2007.

[47] D. Stern and M. Peredo, "*Chronology of Magnetospheric Exploration*" http://www-spof.gsfc.nasa.gov/Education/whchron2.html, Dec 25, 2005.

[48] "*NASA's Cosmicopia– Galactic Cosmic Rays*", Astrophysics Science Division, NASA GSFC, http://helios.gsfc.nasa.gov/gcr.html, May 30, 2007.

[49] L. Biermann, "*Solar Corpuscular Radiation and the Interplanetary Gas*", The Observatory, vol. 77, pp. 109-110, 1957.

[50] E. Parker, "*Interaction of the Solar Wind with the Geomagnetic Field*", Physics of Fluids, vol. 1, no. 3, pp. 171-187, 1958.

[51] LND Inc., "*Geiger-Mueller Tubes*", http://www.lndinc.com/, Nov 2007.

[52] M. L'Annunziata, "*Handbook of Radioactivity Analysis*", Academic Press, 2003.

[53] M. Pöschl and L. Nollet, "*Radionuclide Concentrations in Food and the Environment*", CRC Press, 2006.

[54] M. Rudin, W. Richardson, P. Dumont and W. Johnson, "*In Situ Measurement of Transuranics using a Calcium Fluoride Scintillation Detector System*", Journal of Radioanalytical and Nuclear Chemistry, vol. 248, no. 2, pp. 445-448, 2001.

[55] C. Bloomquist and W. Graham, "*Analysis of Spacecraft On-Orbit Anomalies and Lifetimes*", NASA Goddard Space Flight Center, PRC R-3579, Feb 10, 1983.

[56] C. Yui, G. Swift and C. Carmichael, "*Single Event Upset Susceptibility Testing of the Xilinx Virtex II FPGA*", Proceedings of the 5th Annual MAPLD International Conference, pp. P29.1-6, Sep 2002.

Vita

Shadab Gopinath Ambat was born on May 22, 1983 in Ahmedabad, India. After completing his Bachelor's degree in Electronics and Communication Engineering from L.D. College of Engineering, India in 2004, he decided on pursuing a graduate degree. He enrolled for the MS program in Electrical Engineering at the University of Kentucky in Fall 2005. There he joined the IDEA lab where he worked during the duration of his program.