Escuela Superior de Ingenieros
Departamento de Ingeniería Electrónica
Universidad de Sevilla

Tesis Doctoral

Contributions to the Detection and Diagnosis of Soft Errors In Radiation Environments

Autor: Juan Manuel Mogollón García
Director: Miguel Ángel Aguirre Echánove

ESCUELA SUPERIOR DE INGENIEROS
DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA
UNIVERSIDAD DE SEVILLA

DOCTORAL THESIS

Submitted to the
University of Seville
for the degree of
Doctor on Electronics Engineering

Contributions to the Detection and Diagnosis of Soft Errors in Radiation Environments

Author:
Juan Manuel Mogollón García

Director:
Miguel Ángel Aguirre Echánove

Sevilla, 15 de Agosto de 2012

Dedico este trabajo a mi Madre, que me dio las alas, y a Aurora, que me ayuda a volar...

Acknowledgements

"Knowledge is in the end based on acknowledgement."
– Ludwig Wittgenstein

Esta tesis es el fruto de una larga y apasionante odisea profesional, pero también personal. Me gustaría mostrar mi agradecimiento a todos aquellos que hicieron posible que este barco llegara a buen puerto:

A mi director, Miguel Ángel Aguirre, por su entrega y dedicación, por concebir el germen de esta tesis y, sobre todo, por demostrar una confianza férrea en mi trabajo.

A mis compañeros de trabajo Hipólito Guzmán y Javier Nápoles, a Poli en particular por donarme la plantilla LaTeX de su tesis y a los dos por insuflarme ánimos en los momentos de flaqueza. Ha sido un placer enorme formar parte de este equipo, y espero que nuestro futuro profesional continúe mucho tiempo de la mano.

A Rogelio, que me otorgó la oportunidad de dedicarme a la investigación y me pertrechó para ello. También a Alfredo, por enseñarme valiosísimos rudimentos de la ingeniería electrónica en mis primeros proyectos industriales.

A todos mis compañeros del departamento, que me han enseñado de forma ostensiva lo que es trabajar en equipo: Migue, Clara, Enrique, Perdi, Nando, Jose Ramón, Trini, Carmen, Antonio…

A David Merodio, Boris Glass, Jelle Poupaert y Roland Weigand, de la Sección de Microelectrónica de ESTEC (ESA), por hacer que mi estancia allí fuera deliciosa, y por contribuir con sus comentarios a mejorar este trabajo.

A mi familia, por su apoyo incondicional y paciencia ilimitada.

Y por supuesto al Maestro, cada día con él fue un repertorio de *lessons learned* para la vida, thank you Jon.

Abstract

The effects of ionizing radiation on semiconductors are under study since the invention of the bipolar transistor back in 1947. Outer space is a harsh radiation environment as showed by the first orbiting artificial satellites. It was during the space race in the 50's, when the study of induced errors in critical electronic componentes underwent an important impulse. The need for robust electronics against radiation has been always present in the aerospace sector. Nowadays, the trend in industry towards scale down semiconductor technologies is bringing these concerns down to the electronic equipments operating at sea level. Modern nanoscale technologies are challenging designers to develop new and more efficient hardening techniques to guarantee reliability in electronic components for critical systems in civil aviation, automotive or nuclear energy industries, for example.

- Research activities within the group the author belongs in the field of Single Event Effects (SEE) on electronic devices, unveiled the necessity to establish a methodology to diagnose errors observed on electronic components when put under ionizing radiation. Generally, it becomes very difficult to correlate with certainty the observed errors to the internal faults. Furthermore, complexity inherent to the instrumentation in a particle accelerator radiation test, added to the own design complexity of the device under test (DUT), require some kind of criterion for the validation of observed errors that can be of diverse nature (dose damage, physical damage or even signal integrity issues). Obtaining *cross section* curves for SEE (Single Event Effects) is the common way to represent experimental results from radiation tests. This curves represent a collection of experimental data that have to be previously carefully classified. The same is valid in tests addressed to evaluate Real Time Soft Error Rates (RTSER). Regarding error classification, the standard JEDEC JESD89-1A recommends to follow a "failure criteria" for proper identification of detected errors at the outputs of a circuit in radiation tests. A validation method is required. This thesis introduces a methodology which contributes to the identification of observed errors for proper validation of the experimental results.

- The radiation effects research group based at the Department of Electronics Engineering of the University of Seville have a contrasted experience in the use of hardware emulators for early evaluation of robustness against soft errors in digital designs. Hardware emulation platforms are used to inject faults in the netlist of a digital design in order to study the evolution of the logic state of the circuit under operation. The main advantage in comparison to simulation techniques is in hardware acceleration, allowing fast execution of massive fault injection campaigns. Massive or systematic fault injection campaigns allow to check exhaustively the response of a digital circuit in a harsh radiation environment. These campaigns produce a lot of information on the vulnerabilities of the design. This data must be processed statistically. The availability to correlate the clock cycle for fault injection and the corresponding target register with the associated response would be very valuable to establish the cause of observed errors during a beam test, where only outputs are monitored. The main contribution of this thesis is in the detection and diagnosis of soft errors appearing in radiation experiments.
- Results from an injection campaign depend on the DUT, but also on the set of applied stimuli or *workload*. The results from a massive injection campaign can be used to perform a statistical analysis about the *quality* of the test vectors for diagnosis. It is expected that different faults share the same failure pattern (signature) at the outputs so it'd become impossible to determine the cause of the observed error in a radiation experiment. In the setup of a radiation experiment it is recommendable to consider a set of test vectors that minimize the aliasing in diagnosis. The methodology developed in this thesis definitely contributes to this point, introducing a *figure of merit* for the quality of the set of test vectors.

The work described in this thesis is aimed at establishing a methodology to obtain a *dictionary of faults* where a connection exists between a well known internal fault and the circuit's response coded in a signature of few bytes. In a radiation test, these signatures can be recorded in real time to identify the origin of the damage by means of the fault dictionary, generated by exhaustive fault injection in a hardware emulator. In case the signature is not present in the fault dictionary, it can be stated that the observed error was not generated by a fault like the modeled for the dictionary generation. The origin of the error can be a different type of SEE or an error of different nature, for example physical damage.

The thesis culminates with a radiation test in a particle accelerator. The University of Seville hosts the facilities of the National Accelerators Center, in particular the Tandem Van der Graaf Linear accelerator, which has proven to be an excellent testbench for the validation of the error detection and diagnosis methodology proposed in this thesis.

Table of Contents

Acknowledgements **ix**

Abstract **xi**

1 Introduction **23**

1.1 Electronics and Ionizing Radiation 23
1.2 Radiation Effects on Semiconductors 26
1.2.1 Activation by Nuclear Reactions 28
1.2.2 Displacement Damage Effects 30
1.2.3 Total Ionizing Dose 30
1.2.4 Single-Event Effects 31
1.3 Hardness Assurance Against Radiation 35
1.4 Fault Injection Techniques for Digital Circuits 36
1.4.1 Hardware-Based Fault Injection: HWFI 37
1.4.2 Software-Based Fault Injection: SWFI 38
1.4.3 Simulation-Based Fault Injection: SBFI 38
1.4.4 Hybrid Techniques: Hardware Emulation 39
1.5 Hardware Emulators for Fault Injection 39
1.5.1 Instrumented Fault Injection 41
1.5.1.1 Emulation Systems Depending on a Host 41
1.5.1.2 Autonomous Emulation System 43
1.5.2 Fault Injection by Reconfiguration 44
1.6 Revision of FT-UNSHADES 46
1.6.1 Basic Operation 47
1.6.2 Injection Run 48
1.7 Scope of the Thesis 49

2 Soft Error Testing **53**

2.1 Introduction 53
2.2 SEE Computer Simulation 54
2.2.1 SPICE 55

2.2.2 SRIM 56
2.2.3 TCAD 58
2.3 Radiation Testing 60
2.3.1 Cross-Sections 60
2.3.2 Proton Beams 61
2.3.3 Neutron Tests 62
2.3.4 Pulsed Laser Method 63
2.3.5 Ion Beams 64
2.3.5.1 Broad Beam 65
2.3.5.2 Microprobe 65
2.4 Proposed Methodology for Soft Error Testing 65
2.5 Remarks on This Chapter 68

3 Signature Analysis **69**
3.1 Introduction 69
3.2 Use of Signature Analysis for Soft Error Testing 71
3.3 Signature Module 73
3.4 Hash Functions 75
3.5 Fowler-Noll-Vo Versus RIPEMD160 Hash Functions 76
3.5.1 Implementation on an FPGA Device 78
3.5.2 Pros and Cons 79
3.6 Remarks on This Chapter 81

4 Fault Injection Campaigns **83**
4.1 Introduction 83
4.2 Beam Test Mode of Operation 84
4.3 Fault Dictionary Generation 86
4.3.1 SEU Dictionaries 88
4.3.2 SET Dictionaries 89
4.3.3 MBU Dictionaries 89
4.4 Workload Suitability for Fault Diagnosis 89
4.5 Remarks on This Chapter 92

5 Experiment at the Particle Accelerator **97**
5.1 Introduction 97
5.2 Automatic Test Equipment 98
5.3 Microbeam Chamber 100
5.4 Vehicle Under Test 101
5.5 Signal Integrity Issues 105
5.5.1 Emulator-Side Voltage Translation 108
5.5.2 DUT-Side Voltage Translation 108

5.6 Projectile Selection 109
5.7 Final Test Considerations 111
5.7.1 Multi-Bit Upset 111
5.7.2 SET Errors 112
5.7.3 Timing 112
5.8 Remarks on This Chapter 113

6 Discussion of the Results **115**
6.1 Introduction 115
6.2 Experimental Results 115
6.3 Side Effects 118

7 Conclusions and Future Work **121**
7.1 Conclusions of the Present Work 121
7.2 Future Work 122

Appendices **125**

A VHDL Considerations for Turning FT-UNSHADES into an ATE **127**

B PCB Designs for the Microprobe End-Line **129**

C Log Files from Radiation Experiments **133**

D VHDL Entity for FNV **135**

E VHDL Package for RIPEMD-160 **137**

Bibliography **143**

Glossary **160**

List of Figures

1.1 Explorer I technical details. Source: U.S. Army SMDC 24
1.2 Twins RBSP satellites and Van Allen belts. Image courtesy of NASA/Johns Hopkins University Applied Physics Laboratory . . 25
1.3 Activation by capture of slow neutron 29
1.4 Technology Computer Aided Design (TCAD) simulation of ionization track in a MOS transistor after ion strike at normal incidence [33]. 32
1.5 Parasitic resistors and transistors in CMOS structure giving place to unwanted thyristor eventually triggered by ion strike. 33
1.6 Propagation Induced Pulse Broadening effect after a transient voltage excursion in a node of a chain of inverters. Schematic for mixed-mode simulation [68]. 34
1.7 Schematic of a 6-transistor SRAM cell. 34
1.8 3D model of a 6-transistor SRAM cell for TCAD simulation [33] . 35
1.9 Schema with main blocks in an emulator system managed by a host computer. Source: [21] 41
1.10 Fault Injector architecture for each FF in the design. Source: [22] . 42
1.11 Fault Injector architecture DFF_{inj} for every FF to be injected. Source: [109] 43
1.12 Big picture of an autonomous hardware emulator. Source: [56] . . 43
1.13 Autonomous hardware emulator block diagram. Source: [56] . . . 44
1.14 FT-UNSHADES board. 46
1.15 FT-UNSHADES block diagram. 47

2.1 Pulsed current shaping to simulate the currents appearing after ion or laser irradiation close to the nodes of a MOS transistor. *Note: Time is in logarithmic scale, so rise time is orders of magnitude shorter than fall time. 56
2.2 SPICE parametric simulation to find the threshold value of Q_{tot} in equation 2.1 giving place to bit-flip 57

2.3 Linear Energy Transfer profile for a 12 MeV Mg ion passing through a simplified layer model of the AMIS C5 technology. . . . 58
2.4 SRAM cell for mixed simulation Spice-TCAD to estimate the LET threshold for bit-flip. 60
2.5 Mixed-mode transient simulation for the strike of 12 MeV Mg ion on the cell of Figure 2.4. 61
2.6 Cross section vs. LET curve. Y-axis represents the probability of having SEE per unit area and incident ion. The X-axis is the ion LET. Source: The Aerospace Corporation. 62
2.7 Cyclotron at the CNA. Protons were accelerated to 18 MeV for a first dynamic radiation experiment on a Xilinx CPLD [107]. . . . 63
2.8 Flow chart of the proposed approach. 67

3.1 Figure on the top represents the normal data flow from the DUT output port and the associated hexadecimal signature or *golden* signature. Figure on the bottom represents the circuit operating under radiation, an erroneous data flow (the error pattern is shadowed) and the associated *anomalous* signature. 72
3.2 Modelsim simulation of a bit-flip induced by pulsed laser irradiation of a digital circuit [85]. The method for error detection in this case was a coincidence detector designed to be a reduced version of FT-UNSHADES [3] for radiation experiments. 73
3.3 Signature time delay Td compared to the system clock period Tc. . 74
3.4 Graph illustrating some features of hash functions. 75
3.5 A stage of the cryptographic hash function RIPEMD160. 77
3.6 Schematic view of the FNV module by the Xilinx Synthesis Tool. 79

4.1 Flow chart of the proposed approach. The left branch corresponds to the fault dictionary generation. 85
4.2 Block Diagram showing integration of the Hash Function in the classic architecture of FTUNSHADES 86
4.3 Multi Bit Upset simulation by proton irradiation, triggering a nuclear reaction resulting in direct ionization of several sensitive volumes by an oxygen ion [106]. 90
4.4 (1/2) Failure percentage distribution attending to the number of possible faults involved for different programs running on the Intel 8051 processor. First sector to the right corresponds to failures diagnosed without uncertainty (1 to 1). The number of possible faults involved for a given failure is represented increasing clockwise up to 1 to >3. 93

4.5 (2/2) Failure percentage distribution attending to the number of possible faults involved for different programs running on the Intel 8051 processor. First sector to the right corresponds to failures diagnosed without uncertainty (1 to 1). The number of possible faults involved for a given failure is represented increasing clockwise up to 1 to >3. 94
4.6 Failure percentage distribution for different designs attending to the number of possible faults involved. First sector to the right corresponds to failures diagnosed without uncertainty (1 to 1). The number of possible faults involved for a given failure is represented increasing clockwise up to 1 to >3. 95

5.1 Flow chart of the proposed approach. The right branch corresponds to the radiation experiment. 99
5.2 Automated Test Equipment (ATE) / Test Fixture. 100
5.3 Floor plan of the experimental setup at the CNA facilities. 101
5.4 Vacuum chamber at the nuclear microprobe line of the 3MV Tandem Van der Graaf Accelerator. 102
5.5 Top plate of the vacuum chamber at the nuclear microprobe line and target integrated circuit. The golden lid covering the silicon die is removed before irradiation. 103
5.6 Layout of one of the six replicas of the circuit under test. 103
5.7 JONIC. View of the target die featuring six replicas of the circuit showed in Figure 5.6. 104
5.8 Block diagram of the design under test. See text for a detailed description. 104
5.9 Waveforms of input stimuli. A glitch is showed in the input data of the SR. 106
5.10 Waveforms of input stimuli. A glitch appears in the signal “ENA” that controls the parallel load of the PS registers. 107
5.11 Ringing in the clock signal in the DUT side. 107
5.12 Simulated ionizing LET profile for 13 MeV Carbon ions striking a layered model of JONIC device. 110
5.13 Detail of the ionizing LET profile in Fig. 5.12 to show direct ionization in the active region of the silicon device. 111
5.14 Floor plan of the 32-bit Shift Register under radiation. Colored tiles represent possible patterns for MBU fault injection in order to get MBU fault dictionaries. 113

6.1 Chart representing the FFs affected during irradiation of the 32b Shift Register 117

6.2 Chart representing the interval of clock cycles in the workload affected during irradiation of the 32-bit Shift Register. 118
6.3 3D graph showing the map of faults in the 32-bit SR. The picture in the base plane is the actual layout of the shift register. 119
6.4 Block diagram of the SET detector in JONIC. 119

B.1 Top layer of the testbed holding the DUT inside the microprobe's vacuum chamber. Not to scale. 130
B.2 Bottom layer of the testbed holding the DUT inside the microprobe's vacuum chamber. Not to scale. 130
B.3 Top layer of the pcb adapter for FT-UNSHADES connection to the microprobe's hermetically-sealed feedthrough. Not to scale. . . 131
B.4 Bottom layer of the pcb adapter for FT-UNSHADES connection to the microprobe's hermetically-sealed feedthrough. Not to scale. 131

List of Tables

1.1 Total Ionizing Dose effects. Relevant primary and secondary radiations in different mission scenarios. Source: Handbook of Mitigation techniques against Radiation Effects for ASICs and FPGAs [32] . . . 27
1.2 Displacement Effects. Relevant primary and secondary radiations in different mission scenarios. Source: Handbook of Mitigation techniques against Radiation Effects for ASICs and FPGAs [32] . 28
1.3 Single Event Effects. Relevant primary and secondary radiations in different mission scenarios. Source: Handbook of Mitigation techniques against Radiation Effects for ASICs and FPGAs [32] . 29

2.1 Layer model of an AMIS C5 device . . . 58
2.2 Charge deposition in the layer model of an AMIS C5 device. . . . 59

3.1 Implementation comparison of two different hash algorithms using Xilinx tools and FPGAs. . . . 80

4.1 HDL Synthesis summary for the digital designs used for fault dictionary generation . . . 91

CHAPTER 1

Introduction

“I think there’s a little bit of sizzling here. Honestly, I can feel it.
The ions are flying back and forth."
– Regis Philbin

“Don’t fear failure. Not failure, but low aim, is the crime. In great attempts it is glorious even to fail"
– Bruce Lee

Contents

1.1 Electronics and Ionizing Radiation 23
1.2 Radiation Effects on Semiconductors 26
1.3 Hardness Assurance Against Radiation 35
1.4 Fault Injection Techniques for Digital Circuits 36
1.5 Hardware Emulators for Fault Injection 39
1.6 Revision of FT-UNSHADES 46
1.7 Scope of the Thesis 49

1.1 Electronics and Ionizing Radiation

Ionizing radiation effects on semiconductors showed up for the first time on several in-flight anomalies detected in some missions of the early space race. These anomalies have been reported since the very beginning of the space era [110].

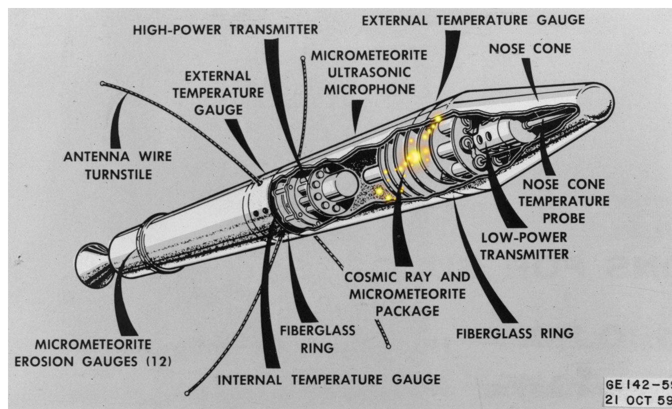


Figure 1.1: Explorer I technical details. Source: U.S. Army SMDC

In 1953, only six years after the invention of the bipolar transistor, Walter Kohn (awarded with the Nobel Prize on Chemistry in 1998) started researching the effects of energetic electrons on semiconductors at Bell Laboratories, under the supervision of William Bradford Shockley [37]. The aforementioned W. Kohn wrote: *"My project was radiation damage of Si and Ge by energetic electrons, critical for the use of the recently developed semiconductor devices for applications in outer space"*. It is noticeable that the effects of radiation on semiconductor devices were a field of study three years before Bardeen, Brattain and Shockley were awarded with the Nobel Prize on Physics for the invention of the bipolar transistor. This anecdote is an example on how the race for the leadership on Space technologies boosted the research of radiation effects on semiconductors.

USA's first artificial satellite Explorer-I (see Figure 1.1) was launched on January 31, 1958. It carried aboard several Geiger detectors as part of an experiment proposed by J.A. Van Allen to measure cosmic rays in orbit. When the satellite reached an altitude of 900 km approximately, detectors mysteriously stopped counting particles. It was found out that Geiger counters had run into saturation detecting much higher flux of particles than expected in the outer space, so Van Allen thought it suggested regions of dense radiation surrounding the Earth. That day were discovered what we know today as Van Allen Belts.

Almost two months later, in March 26, satellite Explorer-III was put into a very eccentric orbit to measure particle flux densities at a wide range of altitudes confirming the existence of the Van Allen Belts. Figure 1.2 represents the electromagnetic confinement of charged particles in the Earth's magnetic field which gives place to the so called Van Allen Belts. The picture is an artistic representation of the twins satellites of the Radiation Belt Storm Probes mission to be launched no earlier than Aug 23, 2012. There exist two belts consisting mainly of high energetic electrons (outer belt, 13000-60000 km) and a combination of protons and electrons (inner belt, 100-10000 km). There are other species of light

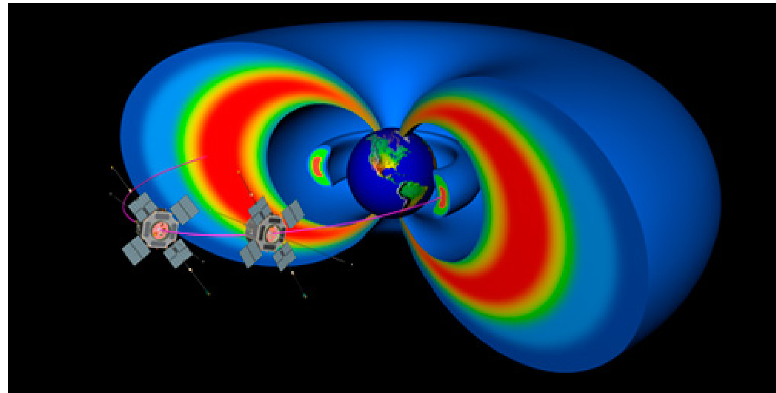


Figure 1.2: Twins RBSP satellites and Van Allen belts. Image courtesy of NASA/Johns Hopkins University Applied Physics Laboratory

ions like alpha particles but in a minor proportion. Earth's magnetic field confines charged particles from the outer space preventing them from reaching Earth's surface and acting as a shield against solar wind and cosmic rays. Recently it has been discovered [1] a belt of anti-protons generated by nuclear reactions of galactic cosmic rays interacting with atmosphere. At typical altitudes of civil aviation and below, the main radiation background consist of neutrons produced in nuclear reactions after cosmic rays or high energy protons penetrate the atmosphere and collide with molecules in it.

Few years after launching the Explorer-I, on July 10, 1962, NASA launched the AT&T's telecommunication satellite Telstar-1, designed and constructed at Bell Labs. Telstar-1 started the era of the telecommunications via satellite. A day before the launching of Telstar-1, on July 9, 1962, the U.S. government carried out a nuclear experiment in high altitude named Starfish Prime. The experiment consisted on the detonation of a thermonuclear bomb equivalent to 1.4 megatons of TNT at 400 km of altitude above the Pacific Ocean. Among other consequences, the detonation injected a big amount of electrons in the magnetic shield of the Earth, increasing considerably the effects of the Van Allen belts on the electronic devices aboard Telstar-1. After several failures during almost one year of operation, the satellite finally lost communications and go out of control on February 21, 1962. It was the first satellite to get lost due to the effects of ionizing radiation on the electronics aboard. Even though problems on Telstar-1 electronics were indirectly provoked by the action of humans, other anomalies on electronics caused by natural space radiation coming from solar wind, cosmic rays and Van Allen belts appeared since the beginning of space exploration. A data base of satellite failures on flight can be found on the web [97].

Density of particles is higher at the poles of the terrestrial magnetic axis, which

is 11° apart from the rotation axis of the Earth. Furthermore, both axes intersect near 500 km above the center of the Earth. This tilt and translation make the south pole of the inner belt closer to the Earth than the north pole originating a region known as the South Atlantic Anomaly (SAA) centered close to the coast of Brazil. The SAA is the area where the trapped particles are closest to Earth's surface, affecting considerably satellites operating at Low Earth Orbit (LEO) which usually switch off their electronics devices when passing through it.

But the problem of radiation on electronics is also present at sea level. First failures reported occurred in 1978 when Intel engineers detected several errors in the stored data of Dynamic Random Access Memory (DRAM) modules. Those effects were named *soft fails*. The origin of the radiation in this case were traces of radioactive elements in the chip encapsulation producing a yield of high energetic alpha particles.

Traditionally the major concerns on the effects of radiation on components and systems were for the space agencies, aerospace industries and for military industry involved in aerospace technologies. However, the increasing scale of integration of new technology nodes is bringing all the issues of radiation closer to other electronic industries like computer makers and automotive industry mainly, among others.

1.2 Radiation Effects on Semiconductors

As far as this thesis is not directly related to the physics of semiconductors, the present chapter is intended to provide a brief introduction to different effects observed in semiconductor devices when irradiated with high energetic particles or photons, without setting out the solid state physics and equations behind them. Interaction of radiation with semiconductors can be classified in three main types attending to the physical effect involved. These types are:

- Activation by Nuclear Reactions
- Displacement Damage Effects
- Ionizing Effects

All these effects must be taken into consideration when designing for reliability in harsh radiation environments and it is not trivial to test hardening against one of this physical effects without being affected by another. For instance, experiments with protons to test soft error rates in a bank of memories must control carefully the fluence of the irradiation session, to prevent dose effects from appearing [78]. It is of great importance to know about side effects during radiation experiments

Mission type	Important primary radiations	Important secondary radiation
LEO	·Trapped p and e^- ·Solar p	X-rays from e^-
High MEO	·Trapped e^- ·Solar protons	X-rays from e^-
Low MEO	·Trapped e^- ·Solar p	X-rays from e^-
GEO	·Low Energy trapped p ·Trapped e^- ·Solar p	X-rays from e^-
Interplanetary space	·Cosmic rays ·Solar energetic particles ·Other planetary trapped-belts	X-rays from e^-
Planetary lander	·Solar energetic particles	Secondary p&n

Table 1.1: Total Ionizing Dose effects. Relevant primary and secondary radiations in different mission scenarios. Source: Handbook of Mitigation techniques against Radiation Effects for ASICs and FPGAs [32]

to account for unwanted phenomena like degradation in the performance of the device or errors coming from causes other than the targeted in the experiment. Discrimination of the errors observed at the outputs of irradiated circuits is a major issue and is one of the motivations of this thesis, as discussed in section 1.7.

The consequences of radiation on devices depend on several factors, as the absorbed dose or the energy of the incoming radiation. High radiation doses can lead to the complete destruction of the device or a permanent failure state only recoverable through annealing or hard resetting. Destructive or physical damage is out of the scope of this thesis and neither non-destructive effects like aging nor performance degradation due to total ionizing dose are considered after this preliminary chapter. Only Soft Errors due to single events will be considered and are suitable for the test technique developed in this thesis. The main radiation effects threatening the reliability of electronics during space missions are showed in Tables 1.1, 1.2 y 1.3.

Among ionizing effects, it is necessary to differentiate between accumulated dose effects and single-event effects in semiconductor technologies. Each type of them will be described separately later in this section.

Single-event Effects on Metal-Oxide-Semiconductor devices are of special interest and typical errors affecting Complementary-MOS transistors will be discussed in more detail due to the presence of this technology in today electronics.

Mission type	Important primary radiations	Important secondary radiation
LEO	·Trapped p ·Trapped e^- ·Solar protons	Secondary n
MEO	·Trapped p (low MEO) ·Trapped e^- ·Solar p	Secondary n
GEO	·Low Energy trapped p ·Trapped e^- ·Solar p	Secondary n
Interplanetary space	·Cosmic rays ·Solar energetic particles ·Other planetary trapped-belts	Secondary n
Planetary lander	·Cosmic rays ·Solar energetic particles	Secondary p&n

Table 1.2: Displacement Effects. Relevant primary and secondary radiations in different mission scenarios. Source: Handbook of Mitigation techniques against Radiation Effects for ASICs and FPGAs [32]

1.2.1 Activation by Nuclear Reactions

Stable nuclei exposed to a flux of energetic particles are candidate to go unstable and radioactive due to nuclear reactions. Particle like protons, neutrons or ions can strike a stable nucleus and remove some nucleons by inelastic scattering. Also low energy neutrons can be captured by a nucleus making it unstable. These nuclei decay by emission of radiation usually in the form of positrons, that can ionize the surroundings in what is a sort of indirect ionization (see Figure 1.3).

There are a variety of possible nuclear reactions with a corresponding cross section[1] depending on the incident particle and the targeted nucleus but the most common in space environments are those produced by high energy protons. Ionization from secondary particles like neutrons are specially important in thick materials.

The most important source of activation is an intensive proton flux located at altitudes of inner Van Allen Belt's equator, with energies ranging from 30 to 400 MeV. For other altitudes, cosmic rays are the leading source of activation and for interplanetary distances solar protons dominate [44]. Except for the case of

[1] The concept of cross section is widely used in nuclear physics to measure the probability of a given nuclear interaction and is measured in barns ($10^{-28}m^2$). The cross section is the equivalent area of the whole target nucleus area producing a given reaction.

Mission type	Important primary radiations	Important secondary radiation
LEO	·Trapped p ·Solar energetic particles ·Cosmic rays	Secondary n
MEO	·Trapped p ·Solar energetic particles ·Cosmic rays	Secondary n
GEO	·Solar energetic particles ·Cosmic rays	Secondary n
Interplanetary space	·Cosmic rays ·Solar energetic particles ·Other planetary trapped-belts	Secondary n
Planetary lander	·Cosmic rays ·Solar energetic particles	Secondary n,p and heavier ions

Table 1.3: Single Event Effects. Relevant primary and secondary radiations in different mission scenarios. Source: Handbook of Mitigation techniques against Radiation Effects for ASICs and FPGAs [32]

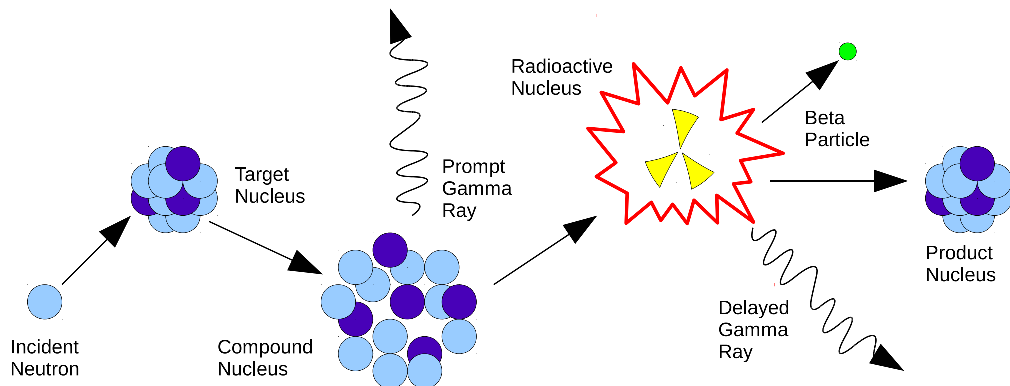


Figure 1.3: Activation by capture of slow neutron

neutrons with no direct ionization, dose due to activation by nuclear reaction is smaller than dose due to primary particle ionization.

1.2.2 Displacement Damage Effects

When particles pass through semiconductors, elastic collisions with atoms in the crystal lattice can result in displacement effects. Eventually the hit of an ion results in the displacement of one lattice atom leaving a vacancy. These defects in the crystal lattice affect the periodicity of the crystal structure, allowing new energy states or energy traps in the band gap for the electrons in the semiconductor, affecting charge carriers mobility. This kind of structural damage is quantified by the Non-Ionizing Energy Loss (NIEL) related to the material for a given incident ion. Macroscopically, the effect associated to displacement damage is a degradation in the electrical response of the device like increase in leakage currents or gain deterioration in bipolar devices. A complete survey on displacement damage effects on silicon can be found in [102].

1.2.3 Total Ionizing Dose

When ionizing radiation pass through the bulk of a semiconductor device, a number of electron-hole pairs are created. The so generated charged particles quickly recombine as they move away by drift-diffusion mechanisms. Charge built up in dielectric layers and interfaces however is not removed easily, mainly the positive charge that remain trapped for long time. Long term exposition to radiation led to the well-known ionizing dose effects. In CMOS devices, charge generated by ionizing radiation in the gate oxide is the responsible for aging or degradation of the electrical characteristics of the device. A common effect in CMOS transistors is a progressive shift in the threshold voltage that increase with dose and directly impact noise margins in digital circuits.

It is important to note that from 130 nm CMOS processes and on, the gate oxide is thin enough to prevent trapped charge from appearing due to quantum tunnel effect. Hence, total dose in modern devices is not so critical and fluence must not be a serious problem in Single Event Effect (SEE) radiation experiments [10].

There are other effects related to rapid increase in the **dose rate** resulting in the injection of big amounts of electron-hole pairs in the semiconductor. The subsequent effects are mainly transient excursion in voltage signals.

1.2.4 Single-Event Effects

These effects are particular cases of ionization where the strike of a single ion or neutron on the active areas of a semiconductor can lead to what is also known as Single-event phenomena (SEP). Depending on the sensitivity of the device, the resulting effect can destruct it or corrupt in some sense its functionality. Figure 1.4 illustrates the ionization track along the first microns of the bulk region in a MOS transistor.

The charge deposition by an ionizing particle passing through a MOS device provokes fast current fluxes in neighbor contact nodes, which lead to transient voltages in capacitive loads. The current pulse is usually modeled like a double exponential with parameters obtained heuristically from experiments, as seen later in this chapter.

The integration of this current pulse is the charge collected and is directly related to the charge deposited by the incident particle. The amount of charge collected is the key parameter determining the behavior of the circuit, but also the shape of this current is influencing as can be read in [41]. When considering Single Event Upsets (SEU) in SRAM cells, there is a threshold value for the minimum charge deposited by the ionizing particle giving place to the flip of the stored bit, this values is known as **critical charge** (Q_{crit}) [28]. High-scaled technologies feature low critical charge, this way lowering more and more the ion linear energy transfer (LET) threshold required to achieve a bit-flip. On the other side, the sensitive volume is also lower in modern technologies balancing the effect of decreased (Q_{crit}) and giving place to a repertory of single event effects [12].

The Dictionary of Terms for Solid State Technology [48] includes the following definitions adopted in this thesis and introduced here for the sake of language accuracy in the rest of the document:

- SEB - Single-event Burnout. *An event in which a single energetic-particle strike induces a localized high-current state in a device that results in catastrophic failure. This effect is associated normally to power MOSFET devices.*

- SEFI - Single-event Functional Interruption. *A soft error that causes the component to reset, lock-up, or otherwise malfunction in a detectable way, but does not require power cycling of the device (off and back on) to restore operability, unlike single-event latch-up (SEL), or result in permanent damage as in single event burnout (SEB).*

- SEGR - Single-event Gate Rupture. *An event in which a single energetic-particle strike results in a breakdown and subsequent conducting path through the gate oxide of a MOSFET.*

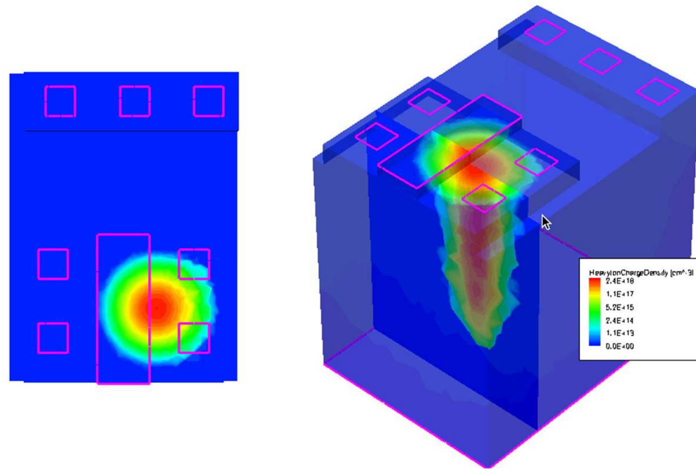


Figure 1.4: Technology Computer Aided Design (TCAD) simulation of ionization track in a MOS transistor after ion strike at normal incidence [33].

- SEH:SHE - Single-event Hard Error. *An irreversible change in operation resulting from a single radiation event and typically associated with permanent damage to one or more elements of a device (e.g., gate oxide rupture).*
- SEL - Single-event Latch-up. *An abnormal high-current state in a device caused by the passage of a single energetic particle through sensitive regions of the device structure and resulting in the loss of device functionality.*
- SET - Single-event transient. *A momentary voltage excursion (voltage spike) at a node in an integrated circuit caused by a single energetic-particle strike.*
- SEU - Single-event upset. *A soft error caused by the signal induced by a single energetic-particle strike.*[2]
- MCU - Multiple-cell upset. *A single event that induces several bits in an IC to fail at the same time.*
- MBU - Multiple-bit Upset. *A multiple-cell upset (MCU) in which two or more error bits occur in the same word.*

In the context of this thesis, the only events of concern are those provoking soft errors or non-destructive effects.

[2] In some publications the term SEU also covers latch-up events

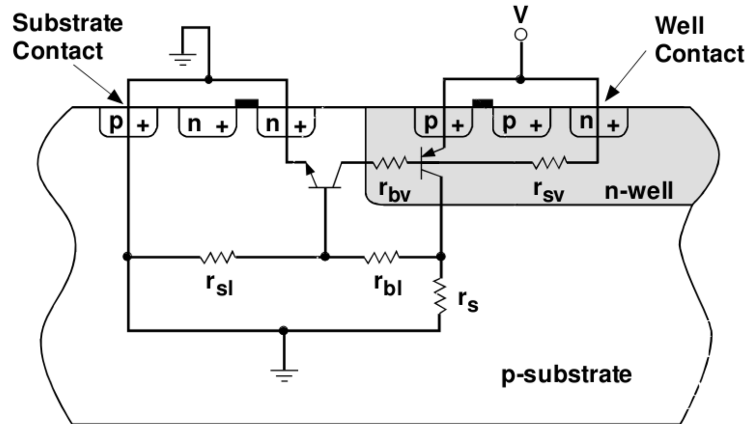


Figure 1.5: Parasitic resistors and transistors in CMOS structure giving place to unwanted thyristor eventually triggered by ion strike.

- Single-event Latch-up.

  Single-event latch-up usually leads to destruction of the affected device if it is not unbiased immediately by an anti-latch-up circuit or similar. In CMOS technology, the charge generated along the ion track can trigger a parasitic thyristor between the power rails (see Figure 1.5) that only can be switched off by opening the supply current path. In some circumstances, it is possible that, due to serial resistance from other parts of the circuit, the consumption is limited and the device is not burned out. These cases are commonly detected as stuck-at bits tied to a fixed voltage.

  It is worth noting that single-event latch-up can't show "latchup windows" [49] as in the case of dose rate irradiation [9] from gamma ray for example.

- Single-event Transient.

  The immediate effect after an ion strikes a semiconductor device is a fast current appearing in the affected node. In CMOS devices this current either charges or discharges the capacity associated with the node giving place to a rapid momentary change in the voltage [24]. These voltage glitches behave as pulses propagating through combinational logic to reach sequential elements like flip-flops. Along the path, these pulses eventually get broader by a phenomenon known as Propagation Induced Pulse Broadening or PIPB Effect [68]. This effect makes transient pulses more likely to be captured by a clock edge when reaching flip-flops, corrupting one or more elements. Figure 1.6 represent the simulated effect in a chain of inverters.

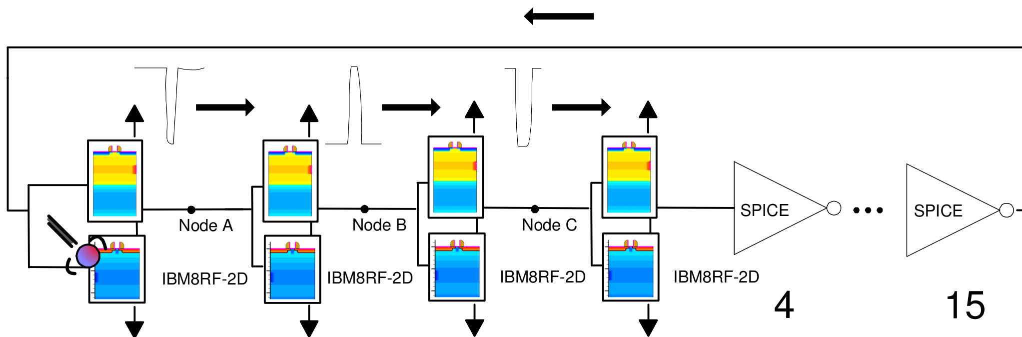


Figure 1.6: Propagation Induced Pulse Broadening effect after a transient voltage excursion in a node of a chain of inverters. Schematic for mixed-mode simulation [68].

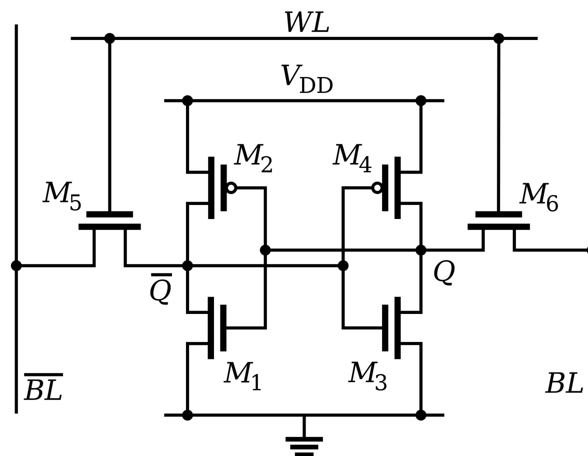


Figure 1.7: Schematic of a 6-transistor SRAM cell.

- Single-event Upset

  The existence of SEU errors was discovered in 1975 [16] although were predicted back in 1969 [112]. The most well known device upset is probably the event of a ion striking a SRAM cell. The upset consists of flipping the logical state stored in the cell. The flipping of a bit is not necessarily an error, since the Flip-Flop (FF) can be rewritten before the corrupt data propagates to other parts of the circuits. Figure 1.7 is the schematic of a typical SRAM cell with two inverters connected back-to-back in a metastable configuration. The transient charge unbalance created by an ion strike can change the state of Q. A Technology Computer Aided Design (TCAD) 3D model of the SRAM cell is showed in Figure 1.8

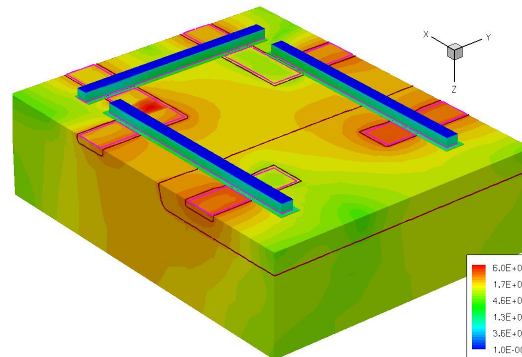


Figure 1.8: 3D model of a 6-transistor SRAM cell for TCAD simulation [33]

As discussed later in Section 1.7, the present thesis focuses on test methods for soft errors. Hence for the rest of the document all the references to errors, test methods, techniques or whatever other topics, must be understood regarding soft errors in semiconductor devices. Dose effects on semiconductors are out of the scope of this Thesis.

1.3 Hardness Assurance Against Radiation

Hardness assurance of electronic parts is a major issue in aerospace industry where semiconductor devices are supposed to be exposed to harsh radiation environments. The Space Product Assurance standard ECSS-Q-ST-60-02C for ASIC and FPGA development establishes a comprehensive set of guidelines for designers including requirements for hardness assurance against radiation. Also the ESA Handbook on Space engineering product assurance entitled: "Techniques for Radiation Effects Mitigation in ASICs and FPGAs" deal with validation methods for the mitigation of radiation effects. Nowadays in space missions, radiation engineers have to get involved from the early stages of the project. As CMOS technology nodes go deeper and deeper into the nanometer scale, other industries with high reliability constraints at ground level are also concerned about environment radiation and usually maintain task forces in the field of radiation hardness validation. The techniques and procedures developed in this thesis are intended to be a contribution in this field.

In computer industry for example, equipments deploying big amount of memory banks such as racks of servers are statistically more vulnerable and traditionally implemented mitigation techniques against radiation. The higher and higher integration and density of memory cells make these devices the most vulnerable, and the most significant contribution to the Soft Error Rate (SER) in System on

Chip (SoC) devices. This point explains why memories have been the "laboratory rats" for researchers, with tons of papers on TCAD and Spice simulations, radiation tests for different species of ions and neutrons, design best practices, etc. Serve as an example the widely tested 6-transistor SRAM (Static Random Access Memory) cells.

As mentioned in Section 1.1, soft errors in commercial electronic devices were reported more than 30 years ago, however the history of standards for commercial semiconductor industry is much more recent. JEDEC standard JESD89A "Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices" is the main document regarding testing of Soft Error Rates of integrated circuits and reporting of results. The Automotive Electronics Council (AEC) published requirements for common electrical component qualification. Document AEC-Q100-Rev-G "Failure Mechanism Based Stress Test Qualification for Integrated Circuits" specifies the cases when integrated circuits require SER testing and refers to the JESD89 standard for the test method. Also DO-254 and IEC/TS 62396 are examples for standards of the avionic industry considering radiation effects issues.

Space and military agencies started standardization and certification procedures for electronic parts hardness assurance establishing the guidelines for radiation testing as in the ESA/SCC Basic Specification No.25100 from the European Space Agency (ESA) or the MIL-STD-750 from the Department of Defense of the United States of America.

There are several papers on design for soft error mitigation, for example [79], with several approaches to enhance robustness against soft error due to radiation.

Testing integrated circuits against environmental radiation is an issue of concern in modern electronic technologies and researchers all over the world are more and more interested in. As long as new hardness strategies and procedures develop, big effort need to be made on the methods to test these advances

1.4 Fault Injection Techniques for Digital Circuits

Fault injection plays a key role in this thesis as introduced in section 1.3. In the SER test methodology presented in this thesis, a preliminary massive fault injection campaign is required to generate a full-coverage fault dictionary for the Circuit Under Test (CUT). The more complete the fault dictionary, the more likely to diagnose faults in the radiation experiment. Of course, covering the whole available cause of errors in a medium-size digital circuit can be a very time consuming task, in much cases unaffordable. Hence, the fault injection platform must be chosen to optimize the generation of the fault dictionary.

There is a wide spectrum of techniques aimed to inject faults in digital circuits

and of course there is no universal platform suitable for all purposes, generally different needs require different fault injection platforms. The main use of these platforms is in validating mitigation techniques at the design stage as in [53]. A well-established classification from text [13] for fault injection techniques is adopted in this document to introduce the state of the art in this field.

Fault injection (FI) platforms must optimize preferably *Controllability, Observability, Intrusiveness, Velocity* and *Cost*. Controllability in this context is the ability to induce faults in all the elements in the target susceptible to fail. Observability is the capacity to observe in time and space the internal and external state of the target. Intrusiveness is the degree of target instrumentation by the injector, it is desirable to have as less intrusiveness as possible. Velocity and Cost must be also taken into consideration. Most of FI techniques are in a tradeoff between these properties and have a limited field of application. The use of these tools is traditionally in the design stages of integrated circuits and in tests of reliability or fault tolerance to faults induced by radiation.

Induced faults in fault injectors are models of the physical fault suitable for a determined level of abstraction, except in the case of ion beam induced faults. In the Register Transfer Level (RTL), the "bit-flip" model of error is of great interest when dealing with soft errors in digital circuits. Except by SET in analog circuits, all soft-SEE can be modeled using the bit-flip model. So this is an atomistic building block to model more complex effects like MCU or MBU.

The bit-flip model is essential in the development of this thesis and is the model adopted for most of fault injectors used to model soft errors. The concept is quite simple and consist of flipping the logic state of a single bit in a given clock cycle. Other models are the stuck-at fault and the bridging. Stuck-at model consists of forcing either logic high or low value in a given port, bit or flip-flop in the Device Under Test (DUT). Bridging is induced by forcing a single port, bit or flip-flop to the logical value of an adjacent port, bit or flip-flop.

1.4.1 Hardware-Based Fault Injection: HWFI

Hardware Fault Injection requires a hardware architecture (in most cases ad-hoc hardware) suitable for fault injection and storage of the resulting effects. For these techniques a working prototype of the DUT must be available, so that these techniques are not to be used in the early stages of the design. There are several platforms for fault injection based on the stuck-at or bridging fault models. The usual way to inject these faults is by sticking or bridging some pins in the prototype, so that these methods are referred to as "pin-level" fault injection. Examples of this approach are MESSALINE [8] developed by the LAAS Research Center at Toulouse (France) and RIFLE [58] developed by the University of Coimbra (Portugal).

Another approach is to inject faults by mean of ion beams generated in a particle accelerator (low controlability) which is in fact the most appropriate test to reproduce the operating conditions in a harsh radiation environments (except by in-flight radiation experiments, of course). For these purposes, the TIMA Laboratory developed THESIC [111] and THESIC+ [34] for radiation testing of Integrated Circuits. To some extent, all radiation test facilities offer a basic testbed for the user to setup an experiment. Intense pulsed laser is a widely used alternative to particle accelerators to inject faults in integrated circuits using backside irradiation with good results [62].

The main advantage of HWFI techniques is in the execution time of injection campaigns (not so the setup time) since the DUT is clocked at hardware speed. There is also the possibility of inducing faults in some blocks that are not accessible by other methods. Some withdraws are usually rigidity, limited observability and poor controlability.

1.4.2 Software-Based Fault Injection: SWFI

These techniques don't need a final prototype of the DUT and are mainly aimed to test complex systems like microprocessors or microcontrollers. The observability and controllability is normally limited to user or context registers and the targets in this case are code blocks, subroutines or applications running in a given microprocessor of interest. The code under execution is modified to change the normal operation of the system in the same way that it would happen when affected by radiation. It is possible in some cases to inject faults in several levels of abstraction, from RTL or failures at the memory elements, to errors in applications like packet duplication in communication networks. A common approach [51] is to include a Code Emulated Upset (CEU) that deliberately corrupts data in a given memory address and is triggered by an interruption request. One of the disadvantages of these techniques are a speed penalty due software execution. There are several mature platforms available like FERRARI [52] or XCEPTION [59].

1.4.3 Simulation-Based Fault Injection: SBFI

A model of the Device Under Test is simulated in a computer and faults are injected by changing logic levels in the model during simulation. These techniques allow for fault tolerance studies in the design stage of the DUT, when still no prototypes are available. Of course, an HDL model of the DUT must be available, which is not always possible. The main advantage is a great observability and controllability, and the major drawback is the penalty in velocity due to all the system is software-based. These approaches provide high observability and controllability.

The most commonly used technique consists of simulating HDL models of the DUT. Among the SBFI platforms we can mention MEFISTO [17], AMATISTA [35] or the "SEU Simulation Tool" [36] developed by the European Space Agency (ESA) and nowadays exploited by the Antonio de Nebrija University for fault tolerance analysis [94].

1.4.4 Hybrid Techniques: Hardware Emulation

Hardware emulation is a type of hybrid technique combining the advantages of SBFI and HWFI techniques. The properties of FPGA devices make them very attractive to be used for hardware emulation. The use of FPGAs allows to synthesize an (VHDL) model of the DUT to test it at hardware speeds, moreover fault injection can be performed via reconfiguration or instrumentation of the DUT (as usually in HWFI techniques). FPGA-based tools are versatile and fault injection campaigns can be highly controllable with full access to all the registers in the design. Cost is not an issue due to constant decrease of prices and enhancement of FPGA technologies. Execution times are improved several order of magnitude in comparison to SWFI techniques. Observability is usually a tradeoff to time, but if time is not a major constraint, full observability is usually achieved. Primary classification in hardware emulation techniques is based on the injection mechanism, differentiating between fault injection by reconfiguration and instrumented fault injection.

Due to the full controllability and high fault rates achieved, hardware emulation is the proper approach for the purpose of this thesis where massive fault injection campaigns are demanded for generating fault dictionaries. Next section is dedicated to a more in-depth analysis of hardware emulation techniques.

1.5 Hardware Emulators for Fault Injection

This section is a revision of several techniques for hardware emulation based on SRAM FPGAs. Firstly, two different approaches are introduced with the different variants that are commonly encountered in the specialized literature. Afterward a first classification is done of the different fault injection platforms operating in the context of a FARM [8] campaign. The FARM model implies the selection of the set of faults to be injected (F: Faults), establishing the set of test vectors to stimulate the system to be evaluated (A: Activation), compilation of observations (R: Readouts) and measuring the reliability of the system (M: Measurements).

The field of application of hardware emulators is in the RTL level of abstraction which turns to be a powerful scenario as discussed in the following.

Actual FPGAs allow for the emulation of systems made of thousands or even millions of equivalent logic gates and can be used to setup fault injections experiments in systems from the proper HDL description. Fault injection techniques in FPGAs also allow for a more accurate and flexible analysis than that from HWFI techniques, since it is possible to inject faults in a controlled manner in all the registers of the design with similar execution times. There are of course a remarkable advantage in execution times when comparing with SBFI techniques [54]. All the platforms described in this section use the bit-flip model but most of them can be tuned to also inject stuck-at faults and others. Stuck-at faults particularly are of great interest for dependability testing community and early works on hardware emulation were based on that fault model [20, 45].

Fault effects are generally classified with the following criteria:

- Failure. Fault induces a wrong behavior observed at the output ports during the execution of the test.
- Latent Fault. Fault modifies the internal logical state of the system with no effects at the outputs during the execution of the test.
- Silent Fault. Fault is removed by the design itself. The affected register is overwritten before fault propagates.

The duration of a fault injection campaign can be estimated by taking into account the number of flip-flops (FF) in the circuit and the number of clock cycles in the whole test-bench (C). If one wants to inject systematically in each and every single FF in the DUT, then the product FFxC is a low limit for the time in clock cycles required to complete such a campaign. Moreover, for systematic campaigns with faults induced in every FF but also in every clock cycle, the low limit is given by FFxCxC clock cycles[3]. As an example, for a set of test vectors with 10^6 clock cycles and a DUT with 10000 FFs, clocked to 10 Mhz, the resulting systematic campaign would take 31.7 years. So that test-benches and DUT size must be constrained to the capacity of the platform to be used. Anyway, speed optimization is achieved only by mean of hardware emulation. Chapter 4 deals in detail with fault injection works done for this thesis. There exist basically two different approaches to inject faults in systems emulated in FPGAs attending to the injection mechanism: **Instrumented Fault Injection** and **Fault Injection by Reconfiguration**.

[3]There are platforms able to lower this limit by saving the whole state of the system before injection, resuming the emulation from this saved state for subsequent injections.

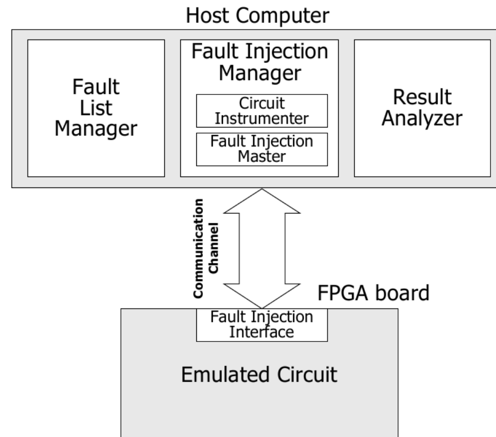


Figure 1.9: Schema with main blocks in an emulator system managed by a host computer. Source: [21]

1.5.1 Instrumented Fault Injection

Instrumentation consists of adding specific injection blocks to the original HDL model of the DUT to have external access to the elements to be attacked by mean of extra ports. There are several tools based on this concept that have proven to be successful. Instrumentation, as it implies modification on the design to be tested, is intrusive, which is probably the major disadvantage of this approach. Moreover there is an area overhead that can't be neglected if injection is required to cover all the flip-flops of the design. On the other hand, execution times in these approaches is minimized and very high fault rates are achieved. The fault injection blocks are attached to FFs in the design and connected in a daisy chain or *scan-path*. The stimuli test vectors are stored in a buffer controlled by the host. Emulator applies test vectors to the instant previous to injection and then resume the execution to the end of test vectors. Failure is detected by comparison of DUT outputs with error-free outputs. Whenever the error is detected and classified before the end of the test vectors, the host stops the run. Fault injection rates reported are really high as in [21] where a score of 100 μs/fault is achieved.

1.5.1.1 Emulation Systems Depending on a Host

Researchers at the Instituto Politecnico di Torino developed a platform for instrumented fault injection [22] with a host computer managing the fault list, operation and result evaluation during the campaign. This architecture is depicted in Figure 1.9. *Fault Injection Manager* or FIM is the core of the system and is in charge of picking up faults from a fault list (optimized by the *Fault List Manager* which is able to perform fault collapsing [14] to some extent, to avoid faults leading

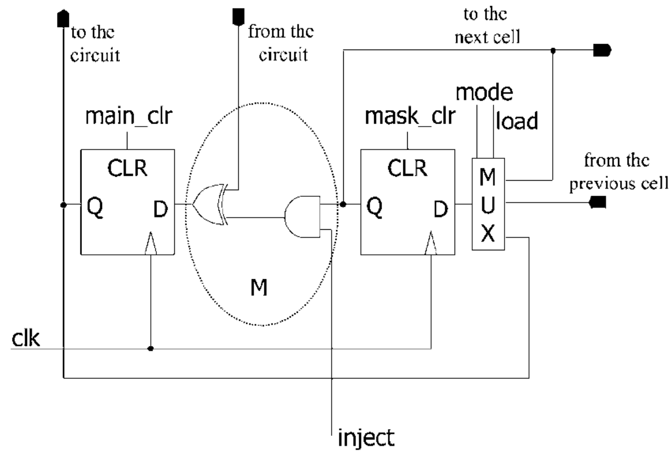


Figure 1.10: Fault Injector architecture for each FF in the design. Source: [22]

to identical results), fault injection in the emulated circuit and evaluation of the emulation results.

The host downloads to the FPGA the instrumented version of the hardware description of the DUT where every FF is modified in the way showed in Figure 1.10. As can be observed, an extra FF and a multiplexer are introduced making up the basic unity of a parallel-to-serial shift register which is the building block of the scan-path. Between this block and the original FF there are glue logic to externally trigger the injection which flips the logic value coming in from the circuit.

Before injection, a fault-free run allows to store the *golden* outputs that will be used to detect failures during the fault campaign by cycle-by-cycle comparison. The scan-path is serially loaded with the fault mask used to set the FF to be flipped (all-zero chain with 1 in the selected FF) and the test vectors are applied to the clock cycle previous to the injection. Then the “inject” input is asserted and the execution resumes. It is noticeable that the logic state of each FF can be saved for subsequent runs and also to detect latent damage.

Another approach is described in [109]. Here, every FF is replaced by a structure named DFF_{inj} that is showed in Figure 1.11. The clock of the DUT is held and then a *CLKSEU* cycle is applied to flip the logic value stored in the FF if *INJ* is asserted. In this case, there is no way to save the state of the FFs.

All the inputs for the injection instrument are shared across the DUT except by the *INJ* signal. That input is dedicated, so it is necessary to implement as much ones of them as FFs are instrumented, which can be a limitation.

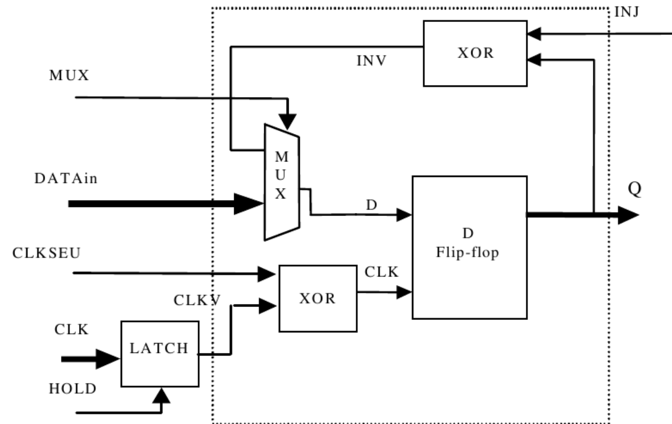


Figure 1.11: Fault Injector architecture DFF_{inj} for every FF to be injected. Source: [109]

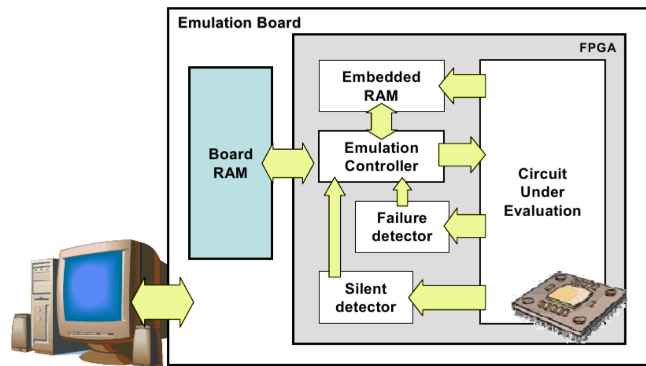


Figure 1.12: Big picture of an autonomous hardware emulator. Source: [56]

1.5.1.2 Autonomous Emulation System

The autonomous emulator is not interacting with a host during the fault injection campaign. This way, the rate of fault injections is maximized with no latencies due to data traffic between host and emulator. Data from each run is stored in RAM memories on board that are dumped to the host computer when filled (see Figure 1.12). The only communication between host and emulator is established at the beginning and at the end of the campaign.

Figure 1.12 shows the block diagram of the architecture. As can be seen, all the blocks are implemented in the FPGA except by the RAM module. The core of the system is the *Emulation Controller* (EC) which manages the whole execution. RAM memories on board provide a test vector each clock cycle and also store the expected output vector (error-free run).

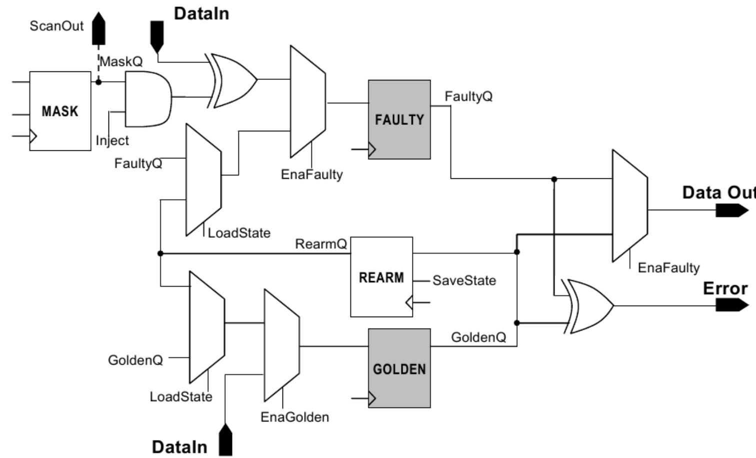


Figure 1.13: Autonomous hardware emulator block diagram. Source: [56]

The EC block commands the application of stimuli to the circuit under evaluation, the injection of the faults and the storage of fault evaluation in the RAM memory. There are modules detecting failures at the circuit's outputs by direct comparison with the expected outputs.

The most complete implementation of this technique is achieved by replacing every single FF in the circuit under test by the block depicted in Figure 1.13. This way, the *golden* and *faulty* instances of the design are multiplexed in time, sharing the same combinational logic. There are FFs capturing the state of both instances and an *Error* signal that is asserted if a discrepancy is found. There exists also the possibility of saving the state of the design at any clock cycle in the run, so runs can be started from the cycle prior to the fault injection.

Autonomous hardware emulation is by far the fastest way for fault injection, of course there is a penalty in area overhead and presumably in the process of instrumentation of the design, if it is relatively complex.

1.5.2 Fault Injection by Reconfiguration

Reconfiguration is the technique consisting of a partial or total change in the configuration bits of an SRAM or Flash FPGA that can be achieved in the field. Reconfiguration is always[4] a time consuming process that contributes to delay the whole campaign execution time, so that it is very important to optimize the injection process timing. This is probably the main penalty for this techniques, specially for those requiring full reconfiguration. On the other hand, these techniques

[4]It is possible to achieve reconfiguration without halting the device but it depends on designs made from parts running independently

are not intrusive since no additional HDL blocks need to be added to the design for the purpose of fault injection.

From long ago, FPGAs allow for partial reconfiguration of the configuration bits reducing drastically the time needed to modify the implemented design or application. For this reason, most of modern techniques used for fault injection rely on partial reconfiguration. Nevertheless, early works like [20] were also based on full reconfiguration, in that case to inject stuck-at faults.

The whole process is commanded by a host application running in a computer and the set of test vectors is stored in memories on board using compression algorithms. There are two ways to cope with reconfiguration regarding the execution of the application implemented in the FPGA. Reconfiguration performed before starting execution is known as Compile-Time Reconfiguration or CTR [46] and reconfiguration in run-time is known as RTR [7]. In the context of fault injection, CTR implies that fault is present in the DUT from the very beginning of the test as in [5]. Run-time reconfiguration is performed by halting the running application in the FPGA for globally or partially reconfiguring it and then resuming execution.

In [7] RTR is performed to inject stuck-at and bit-flip faults in the DUT. However, the proposed methodology is not optimizing the reconfiguration time, because a complete reading of all the FFs in the FPGA is necessary, which makes delays depending on the size of the configuration file. In this case, the Jbits API [38] is used to interface the DUT implementation running in the FPGA. Another approach is reported in [26] where only the portion of the configuration memory where the FF to be injected is allocated, needs to be read and modified. The Xilinx Radiation Test Consortium (XRTC) developed a Fault Injector that is also a test platform for radiation experiments intended to test SRAM FPGAs .

Hardware emulation is the proper technique for massive campaigns as required for the purposes of this thesis. Among the existing techniques for HWFI, in the context of this thesis, a high automation would be desirable to enhance campaign timing, since user only need to define an initial massive campaign and then press the “big button” to record all the possible responses of the CUT. Attending to time consumption, which is probably the major constraint in the technique proposed in this thesis, autonomous emulation is the most suitable approach. However, as far as it is not in the scope of this thesis to achieve the best performance, the well-known FT-UNSHADES platform has been deployed with excellent results. Moreover, the more sophisticated FTU2[5] fault injection platform is inheriting the know-how from FT-UNSHADES and will be ready to perform autonomous emulation for systematic campaigns.

Next section is entirely devoted to a detailed description of FT-UNSHADES, the fault injection platform used in this thesis to generate fault dictionaries per-

[5] System under development. More information on FTU2 can be obtained from [69]

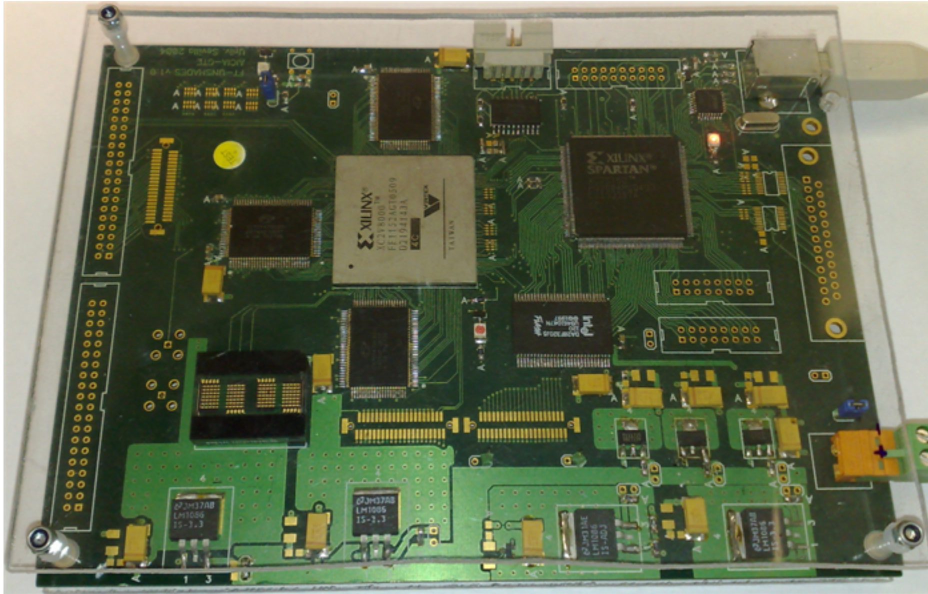


Figure 1.14: FT-UNSHADES board.

forming partial RTR via Select-MAP.

1.6 Revision of FT-UNSHADES

FT-UNSHADES is a hardware emulator for fault injection analysis of digital designs based on Xilinx FPGAs (see the platform board in Figure 1.14). The system is intended to perform early evaluation of design reliability against soft errors. The design under test is implemented in a system-FPGA from its hardware description or a netlist file, and faults are injected dynamically by partial reconfiguration via Select-MAP. Therefore, the emulator works at the Register Transfer Level. A complete description of the system can be found in [76].

The basic operation mode of the emulator consists of the injection of a fault by flipping a single or multiple FF in the whole design. By mean of capture-readback-reconfiguration cycles, it is possible to get the state of a single FF, change it and then reload the new value in run time.

A host application running on a computer processes a file containing the logic allocation (.ll) of the netlist in the FPGA device featured by the emulator. The allocation file is generated by Xilinx standard tools and contains information relating the logical resources in the FPGA with the netlist synthesized in the early steps of the design process. All the configuration bits in the FPGA dedicated to implement user's funcionality are perfectly related to a node in the design netlist. This way

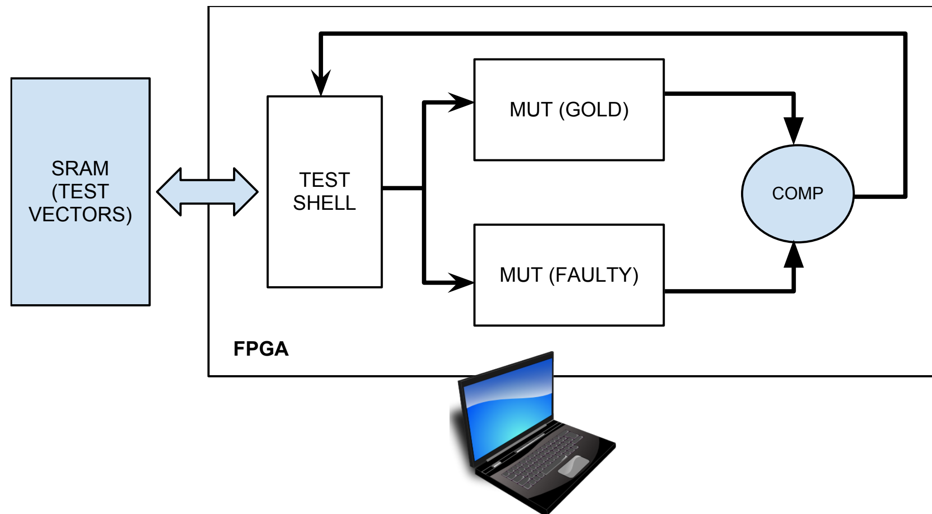


Figure 1.15: FT-UNSHADES block diagram.

is how FT-UNSHADES assures *observability*. The test is also perfectly controllable since DUT clock can be paused in a precise test cycle and faults injected in whichever FF in the design, providing also full *controllability*.

This tool is mainly conceived for fine selection of targeted registers before starting the fault injection campaigns and not so to perform massive attack over the whole design, as done by some other tools introduced in section 1.5. There are more information available at the web sites: [4, 72]

1.6.1 Basic Operation

The main component in the system is a Xilinx Virtex-II FPGA where the Module Under Test (MUT) is instanced two times, one of them is used for fault injection and is known as SEU-MUT, and the other is used as an error-free instance running in parallel, known as GOLD-MUT. The golden replica is there for output comparison. The architecture of this basic operation mode is shown in figure 1.15. Together with the two instances of the MUT there are an extra module to manage test timing and for the application of the input stimuli. The test shell controls the MUT clock to pause the execution, inject the fault and resume execution. If logic for comparison detects any discrepancy between SEU and GOLD instances, the test is stopped and the error logged.

The basic mode of operation is not suitable to be used for microprocessor systems where the running program can implement self-recovery subroutines if errors are detected. In such cases, the two MUT instances in the basic mode of operation go out of synchronism (misaligning) for the number of clock cycles that

takes to the SEU-MUT instance to repair the damage. To avoid this situation, the Smart Table mode of operation was developed for FT-UNSHADES, that replaces the GOLD-MUT by a table containing the expected outputs for the processor for all the workload. The Smart Table also controls the possible misaligning in the outputs with a time-to-recovery parameter that flag a non-corrected error when it is exceeded [39]. This mode of operation is known as FT-UNSHADES-uP.

An extra operation mode was also implemented to inject faults in the configuration bits of the FPGA instead of injecting in the FF of the design. This mode of operation is known as FT-UNSHADES-C [40].

For this thesis, a new operation mode has been developed to use FT-UNSHADES in a real radiation test with little modification of the basic mode operation. This mode is explained in detail later in Chapter 4.

1.6.2 Injection Run

A whole fault injection campaign is made of a number of "Injection Runs" where a time and location are defined to perform injection in the design. These runs imply a complete execution of the workload with at least one bit-flip insertion. The steps followed in a single run are summarized here:

1. The clock cycle for fault injection and the targeted FF or FFs are sent to the control FPGA in the FT-UNSHADES board.
2. The test shell stores the injection cycle and starts the run.
3. When the workload cycle equals the injection cycle, the execution is paused and the targeted FF or FFs are flipped. This step implies a capture-readback-reconfiguration cycle involving the host computer[6].
4. The execution is resumed until an error is detected at the outputs or until the workload is completed.
5. If an error is detected, the cycle of detection and port affected are stored.

There are complete flexibility to inject faults whatever in the design and in the time of execution. It is also possible to inject more than one bit-flip in a single clock cycle to emulate MBU or MCU.

In some cases of interest, it is possible to perform cycle-by-cycle analysis to check fault propagation through the circuit.

[6]This transaction with the host computer will be avoided in the upcoming FTU2 system, speeding-up considerably the time per injection.

1.7 Scope of the Thesis

To set out the reasons and circumstances that motivated the work done for this thesis, it is necessary to introduce the recent activity of the research group that the author belongs. This thesis has been developed in the context of three technical projects: EMULASER, RENASER and RENASER+ in which the author got involved the last four years.

Project EMULASER (Emulation of the effects of cosmic particles on electronics by LASER irradiation) was co-funded by the Spanish Center for Industrial Technological Development (CDTI) and the Spanish Ministry of Industry, Tourism and Commerce to be a study of viability on the industrial use of pulsed laser for fault injection in semiconductor devices for aerospace systems. EMULASER put together several task forces with different capabilities in the field of the effects of radiation on electronics to approach the problem from different angles, from solid state physics TCAD simulation to pulsed laser experiments. The use of hardware emulators was proved to be useful to improve the visibility of test vectors in the real laser experiments. In radiation experiments, when testing hardened designs or fault tolerant technologies, it is necessary to choose a set of stimuli that minimizes silent faults. Faults silenced by the test vectors could hide a vulnerability in the design itself. Furthermore, a reduced version of FT-UNSHADES (also known as FTUSB) hardware emulator was used during laser experiments to apply stimuli to the DUT, and also as a coincidence detector comparing golden outputs with those from the irradiated DUT to find *failures*.

First experiences with pulsed laser were carried out at the Faculty of Physics of Salamanca (Spain) on a digital full-custom integrated circuit[7] in a 0.5 microns CMOS technology from a low cost educational foundry [105]. Results from these experiments were published here [85] and the need for a test methodology to identify and diagnose soft errors showed up for the first time in our research group. Our test vehicle was fully designed in the Dept. of Electronics Engineering of the University of Seville and have been targeted in different experiments. Latest version was fabricated in a 130 nm CMOS process from STMicroelectronics. The design behind this test vehicle is in the following referred to as JONIC (Jonathan Tombs Integrated Circuit[8])

Projects RENASER (Effects of Radiation on Aerospace Systems, An Investigation on Emulation) and its continuation RENASER+ (Integral Analysis of Digital Circuits and Systems for Aerospace Applications) are funded by the Spanish Science and Technology Inter-ministerial Commission (CYCIT) to boost research in the field of radiation effects on semiconductors. The author's expertise in parti-

[7]Description of the DUT design will be done later in the chapters related to the experimental tests.

[8]This circuit was named JONIC in memory of Prof. Jonathan Tombs

cle accelerator tests has been acquired in the frame of these two projects through several experiences at the National Accelerators Center (CNA) based in Sevilla, Spain. Early experiences at the 18-MeV Cyclotron [107] showed that experimental setup at ion radiation facilities is not trivial, and a number of best practices must be followed in order to achieve valid results. Later experiments were carried out at the 3 MV Tandem Van der Graaf accelerator using the micro-probe end line for high focusing of the beam. All the radiation tests where commanded by the FTUSB platform. The success in the use of this platform led to the use of the full FT-UNSHADES emulator for the key experiment on this Thesis.

The JEDEC STANDARD JESD89A states in section 3.3.3 a difference between static and dynamic testing. Through several irradiation experiences within the aforementioned Projects, different static and dynamic tests were done over the JONIC design in both 0.5 μm and 0.13 μm chips. In the context of this Thesis, static tests are those in which device is not clocked but frozen in a given logical state when irradiated, this is the way SRAM memories are tested. Dynamic Testing on the other hand, applies when the device is running while irradiated. For ASICs, dynamic testing is preferable to study the behavior of the chip in normal operation under radiation. Results from Dynamic testing on a rather simple design as JONIC show that a methodology for diagnosis and verification of soft errors is required. The main purpose of the methodology developed in this thesis is to address the next to issues:

- **Soft Error Validation**. Ion beam is the definitive test for part validation against radiation and components must be certified in such complex facilities. Furthermore, it would be desirable to perform dynamic testing in many cases to represent the normal operation of the component to be evaluated. The problem of observability in these experiments is well known, since devices performs as black boxes where only output ports are registered. Even though it is possible to detect failures by comparison with expected outputs, it is not evident to relate them with single-bit SEU-like faults or with any other causes like MCU or transient power cut off. An observed failure can be produced in a real experiment for many other causes even not directly related with radiation but with the usually complex experimental setup. For this reason, the experimenter needs to be sure about what he is observing, as stated in JEDEC STANDARD JESD89A-1A, Section 6 ("Failure Criteria"): *Any result that does not match expectation is a possible soft error and shall be recorded. If a possible error is identified, action should be taken to verify that it is a soft error.*

- **Failure Diagnosis**. Maximum focusing in laser experiments is near 1 square micron in top-side irradiation. It helps to identify the origin of the damage

spatially, not so in the temporal dimension. However, in ion beam radiation experiments, it is much more difficult to determine the origin or cause giving place to the observed failures. In ESA/SCC Basic Specification No. 25100 entitled "Single Event Test Methods and Guidelines" section 4.2.1, it can be read: *The test software shall be capable of logging the number of upsets, the location and the time*. Knowing about origins of failures or diagnosing then become important from the designer's point of view to evaluate reliability of hardening solutions or technologies [93]. As said before, the need of knowledge about the origin of soft errors in dynamic tests of ASIC or other logic circuits arose naturally in radiation experiments, even in laser experiments, where all the experimental variables are much more controlled.

Therefore, the scope of this thesis is in developing a technique for soft error validation and failure diagnosis to enhance observability in radiation experiments of digital circuits, providing a methodology not bound to specific platforms or tools. The tools and resources deployed in this Thesis are not optimal but only used as *proof of concept*. Optimization of the technique is out of the scope of this Thesis.

CHAPTER 2

Soft Error Testing

"It could be happening on everyone's PC, but instead everyone curses Microsoft."
– Paul E. Dodd

Contents

2.1 Introduction 53
2.2 SEE Computer Simulation 54
2.3 Radiation Testing 60
2.4 Proposed Methodology for Soft Error Testing 65
2.5 Remarks on This Chapter 68

2.1 Introduction

The study of soft errors in digital circuits is commonly constrained to SRAM and DRAM devices due to a higher sensitivity per part area. However, deep sub-micron technologies are sensitive enough, and soft errors in circuital core logic must be considered [42, 101], like SETs and SEUs in internal registers.

This chapter is a first approach to radiation testing of digital circuits introducing ground-level experiments to predict SER. Moreover, some examples of dynamic radiation tests are commented where the technique of detection and diagnosis developed in this thesis would be applicable.

In-lab radiation testing is the final procedure for hardness assurance of components designed to work in harsh radiation environments or validation of Commercial Off the Shelf (COTS) parts to be flown in space missions. Radiation tests on

enhanced radiation background to induce soft errors, are in the group of Accelerated Soft Error Rate (ASER) techniques. These techniques make the target device undergo a high radiation exposition under controlled energy, flux and fluence. Prediction for component reliability must be made by extrapolation depending on the in-orbit radiation environment. This is the common procedure for researchers and radiation engineers. On the other hand, System Soft Error Rate (SSER) consists of irradiation of a high number of devices in the radiation environment similar to the environment of operation of the chip. This method is also known as *field testing* and is usually related to industrial validation of components.

Microelectronic parts have to be qualified against radiation attending to the component use. The rate of soft error events in orbit for a given component can be calculated from the in-lab response to radiation and the environmental radiation conditions expected when the component is on duty (mainly the particle fluence versus LET). For heavy ions, ground tests have to cover the full range of LET by trying different Z and energies for ions in a particle accelerator. Robustness against neutrons must be tested with neutron sources, for example for avionics [104], and high energy proton irradiation has to be considered for certain space missions due to the predominant presence of these ions in space.

The methodology presented in this thesis is introduced and proposed as an improvement in the visibility of the results of ASER radiation experiments for dynamic radiation tests. Instead of memories, tested statically, other logic devices like microprocessors or digital ASICs have to be tested under run to get confidence in the SER estimations. It has been demonstrated that static radiation tests of these devices could lead to overestimation in the SER predictions [15, 108].

2.2 SEE Computer Simulation

Computer simulation of SEE on semiconductor devices is recommendable when facing radiation experiments, specially in low energy accelerators. For a given technology, it is necessary to estimate whether the accelerator facility is in the range of LET needed to observe SEE effects or not. Moreover, selecting the ion specie and energy required depending on the objectives of the experiment, is a key first step and must not be improvised. This section is a brief introduction to a set of different simulation techniques.

The author of this thesis has been involved in computer simulation of SEE prior to face the accelerator experiments carried out for this thesis at the CNA 3MV Van der Graaf facility [71, 87]. Preliminary SEE experiments involving old CMOS 0.5 μm technology in this particle accelerator resulted in an upper bound for the maximum achievable bit-flip capability in this low energy accelerator, dedicated mainly to Ion Beam Analysis (IBA) techniques. These experiences allowed

us to address further experiments in the scope of this thesis, over a test vehicle developed on the newer CMOS 0.13 μm technology.

A work done on the simulation of the PIPB effect on a CMOS 0.13 μm [68] is mentioned later in Chapter 5, Section 5.7, to establish an upper bound for the device clock frequency in order to avoid SET errors in the SEU radiation experiment carried out for this thesis.

The following subsections are a brief introduction to the simulation methodology adopted to estimate the LET range necessary to induce soft errors in a given technology.

2.2.1 SPICE

During the design phase, engineers can take advantage of the tools provided by design suites for spice simulation, to get a first estimation of the effects of radiation in a given circuitry topology [29]. The common approach consists of placing a pulsed current source in a sensitive node to trigger a discharge in the same way that it occurs after ion striking or laser irradiation [91]. For a CMOS transistor in off state, charge from ionizing radiation establish electrical paths in the bulk silicon, allowing charge from the drain node to be evacuated and dropping voltage at this node. This rapid voltage excursion in some circumstances can induce a flip in the logic value of a SRAM cell, for example.

The current pulse depends on several factors and there are different analytical expressions to model it, however a double exponential is commonly used [41]:

$$I_{pulse}(t) = \frac{Q_{tot}}{\tau_f - \tau_r} \left[\texttt{exp}\left(-\frac{t}{\tau_f}\right) - \texttt{exp}\left(-\frac{t}{\tau_r}\right) \right] \tag{2.1}$$

Parameter Q_{tot} is the total charge evacuated resulting from integration of the curve between 0 and ∞. Parameters τ_r and τ_f are the characteristic rise and fall characteristic times of the double exponential. Typical values for these parameters are:

$$\tau_f = 100ps \tag{2.2}$$

$$\tau_r = 100fs \tag{2.3}$$

These values are related to the response of the semiconductor to the generated charge. Fast drift currents appear in the depleted region due to the high electrical fields, contributing to the steep rise in the current pulse in Figure 2.1. The long-term decay in the curve is due to the diffusion of carriers through the bulk, also known as “delayed-diffusion” [13].

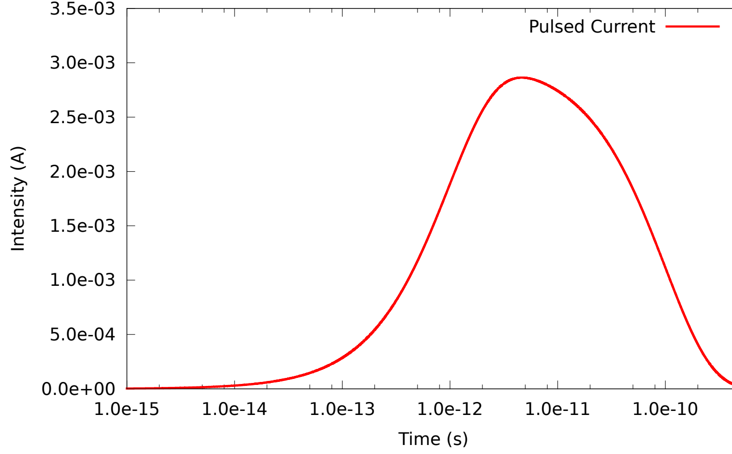


Figure 2.1: Pulsed current shaping to simulate the currents appearing after ion or laser irradiation close to the nodes of a MOS transistor. *Note: Time is in logarithmic scale, so rise time is orders of magnitude shorter than fall time.

Putting this pulsed current source in a critical node and sweeping the parameter Q_{tot} during SPICE simulation it is possible to find the threshold value for Q_{tot} inducing the flip of a SRAM cell for a given technology. This parametric analysis in Figure 2.2 shows a threshold value for Q_{tot} of near 510 fC (green line). This value is a rough first approximation for Q_{crit} an can be considered when no more precision is required. This value is directly related to the charge generated by ionization, but it is not exactly the same; there are loses by recombination and an amplification effect caused by a parasitic BJT transistor in CMOS devices, which can be about 30-40% of Q_{tot} [27].

The value Q_{crit} obtained by this method is a hint or educated guess for the simulation procedures introduced in next section.

2.2.2 SRIM

The Bethe formula describes the ionizing energy loss of a charged particle (proton, alpha particle or ion, not electron) as it passes through matter. The relativistic version of this formula was proposed by Hans Bethe in 1932:

$$-\frac{dE}{dx} = \frac{4\pi}{m_e c^2}\frac{nz^2}{\beta}\left(\frac{e^2}{4\pi\varepsilon_0}\right)\left[ln\left(\frac{2m_e c^2\beta^2}{I\left(1-\beta^2\right)}\right)-\beta^2\right] \tag{2.4}$$

Formula 2.4 considers the charged particle completely stripped of electrons and is not a good approximation for slow or high-Z ions. Linhard-Scharff-Schiott

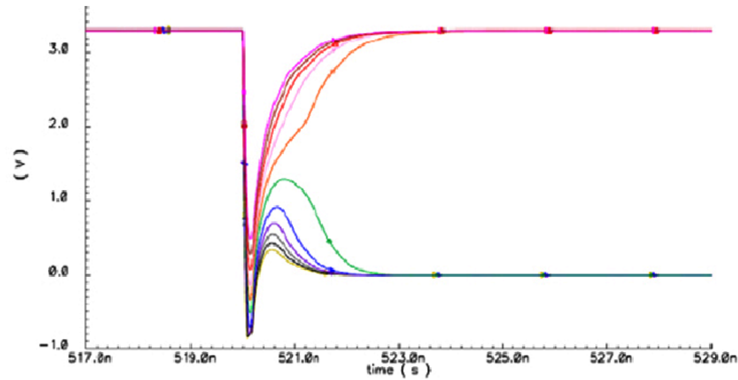


Figure 2.2: SPICE parametric simulation to find the threshold value of Q_{tot} in equation 2.1 giving place to bit-flip

(LSS) Theory improves the theoretical treatment of the ion-matter interaction including a complete model for the dependence of the charge state of the ion on its velocity as it travels through matter.

The software Stopping Range of Ions in Matter (SRIM [114]) is a powerful tool based on Monte Carlo simulation for the study of ions passing through matter including all the theoretical knowledge in this field since the Bethe formula. This tool is rather simple and is widely used by the radiation effects community [63, 64, 113].

The output of the simulator is, among other data, the direct ionizing energy deposition of the ion along the track. This linear energy transfer can be converted approximately to charge by means of the energy necessary to create an electron-hole pair. In Silicon it is:

$$E_{e-h} = 3.6eV \tag{2.5}$$

It is necessary to know the ion range, energy deposition by direct ionization and also the critical volume of charge collection in the semiconductor in order to compute the amount of charge generated. That is basic to make estimations on the total charge drained from a neighbour capacitive node. Range and energy deposition of the ion can be extracted from SRIM calculation if a proper layer model of the target is used. The collection volume or sensitive volume depends on the targeted technology, for a CMOS 130 nm bulk silicon technology the collection depth is around 1 micron [6]. For older CMOS technologies like AMIS C5 [105] with 0.5 μm feature size, the critical depth can be around 3 μm and has to be taken into account to obtain the net charge in the sensitive region.

As an example, Figure 2.3 shows the SRIM LET calculated for a 12 MeV Mg ion on an AMIS C5 [105] target modeled by the layers in table 2.1. Several

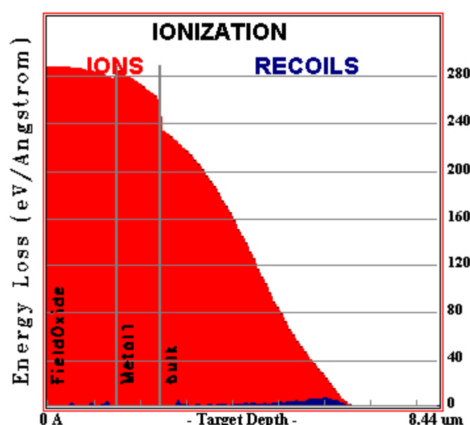


Figure 2.3: Linear Energy Transfer profile for a 12 MeV Mg ion passing through a simplified layer model of the AMIS C5 technology.

simulations for different ions and energies result in table 2.2. Recalling the result from SPICE simulations, these results suggest that for a 0.5 μm CMOS technology, 50 MeV S ions or higher are required to flip the logic. This is not feasible for the cocktail of ions and energies available at the 3 MV electrostatic accelerator at the CNA facilities. However, more accurate TCAD simulations lowered the estimation for the required LET [67].

As discussed later in section 5.6, the use of SRIM simulations led to define the proper ion specie and energy for the radiation experiment carried out in this thesis.

Material	Thickness	Density (g/cm^3)
Oxide (SiO2)	1.5μm	2.200
Metal1 (Al)	0.94μm	2.702
Substrate	6μm	2.321

Table 2.1: Layer model of an AMIS C5 device

2.2.3 TCAD

Technology Computer Aided Design (TCAD) [99] is a suite of software tools for the simulation of technology processes and semiconductor devices. Three-dimensional solid state physics equations involving charge carriers and electric fields in the semiconductor are solved by numerical methods. For semiconductor device simulations, the main equations involved are the non-linear Poisson equation (eq. 2.6) and the continuity equation (eq. 2.7).

Ionic Specie	Energy (MeV)	Charge Deposition (fC)
Fe	200	866.7
Ca	100	666.7
S	50	500.0
Al	25	360.0
Mg	12	143.3
O	6	65.3

Table 2.2: Charge deposition in the layer model of an AMIS C5 device.

The 3D model of the device is complex and incorporates all the structure, doping profiles and characteristic concentrations to a given technology. For the first time a technology is going to be put under the beam to test reliability, it makes sense to simulate the response of the device to a given ion available at the accelerator facility. It becomes mandatory for older technologies and low energy accelerators for SEU experiments, for example.

$$\nabla \cdot \varepsilon \nabla \phi = -q\left(p - n + N_D - N_A\right) - \rho_{trap} \tag{2.6}$$

$$\nabla \cdot \vec{J_n} = qR_{net} + q\frac{\partial n}{\partial t} \qquad \nabla \cdot \vec{J_p} = qR_{net} + q\frac{\partial p}{\partial t} \tag{2.7}$$

New technologies show up effects never detected before in older technology nodes, as it occurs when dealing with SET [61] for 0.35 μm CMOS and newer.

To check the capability of the 3MV Tandem Van der Graaf accelerator at the CNA to accelerate ions with LET over the bit-flip threshold, a thorough simulation study was done on a well-known 0.5 μm CMOS bulk technology[1]. It was followed by an assessment on the available high energetic ions at the CNA. Spice-TCAD mixed-mode simulations resulted in the validation of the facility and different radiation experiments showed that prediction from simulation were right. Results were reported in [89]. Nevertheless, the test vehicle used as proof of concept for this thesis has been fabricated in a 0.13 μm CMOS bulk technology, therefore having lower LET threshold for bit-flip than that for the 0.5 μm. Validation of the 3MV Tandem Van der Graaf to induce bit-flips in the 0.5 μm technology has been shown to be an upper bound for SEU generation when testing technologies with smaller feature sizes.

Figure 2.4 is the basic SRAM cell for Spice-TCAD simulation. TCAD supports a tunable heavy ion model, and transients simulations can be run on voltages at nodes A and B. It is useful to get the LET profile leading to the flip of the cell.

[1]A 3D model for a transistor in this technology was profusely studied and precisely calibrated by the author [67].

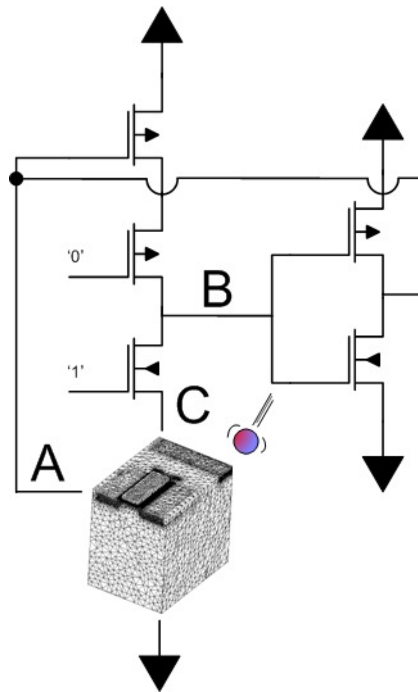


Figure 2.4: SRAM cell for mixed simulation Spice-TCAD to estimate the LET threshold for bit-flip.

A transient simulation is plotted in Figure 2.5 using the estimated LET profile for a 12 MeV Mg ion. The title of the graph means that, for the ion energy, LET actually depends on depth with a lateral profile that can be obtained by SRIM simulation (see 2.2.2). The light green line represents the charge current at the struck node, showing a sharp peak due to fast drift current, as mentioned before. Voltage at node A is represented by the black line, and voltage at node B is represented by the red line. The transient simulation clearly shows the flip of the SRAM cell for the given ion. These simulations were very valuable to establish a "LET of reference" at the CNA.

2.3 Radiation Testing

2.3.1 Cross-Sections

Cross-section studies are the most used for component characterization against soft error and can take advantage of signature analysis as a way to apply failure criteria. A typical cross-section curve is shown in Figure 2.6 for SEE in microelectronics components (usually memory parts). The cross-section represents the

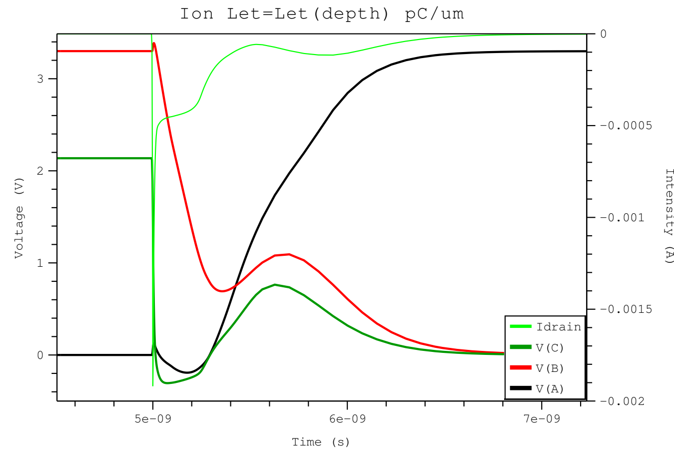


Figure 2.5: Mixed-mode transient simulation for the strike of 12 MeV Mg ion on the cell of Figure 2.4.

probability of finding SEE errors depending on the incident ion LET.

The cross-section data are commonly fitted to a Weibull curve with a low-LET threshold, a knee and a saturation region. In some cases, data plots include error bars accounting for some uncertainty in the number of faults computed for a given LET. This uncertainty is related to the lack of confidence in the cause of the observed error. For this reason, cross-section studies require a method for failure classification or validation.

2.3.2 Proton Beams

High energy protons are the most predominant particles in the cosmic ray background in space. Protons are also predominant in solar wind and in the Van Allen inner belt. Even for relatively high energetic protons the associated LET is rather low, and rarely in space protons induce SEE by direct ionization [78]. Protons may also interact with nuclei triggering nuclear reactions giving place to recoiling nuclei (elastic collisions) or reaction products (inelastic) that can be highly ionizing particles.

SEE Cross-sections for protons are obtained by gradually increasing the proton energy from energies below the nuclear reaction threshold which is close to 3 MeV [50], to energies in the saturation region of the cross section, near 100 MeV.

Depending on the device sensitivity to SEE and proton energy, it must be necessary a high fluence to get a representative amount of errors which can lead to TID damage. It is recommendable to estimate firstly the accumulated dose during the SEE test and, previous to the experiment, put the device in a ^{60}Co gamma source to check the reliability of the device. TID damage can be monitored by

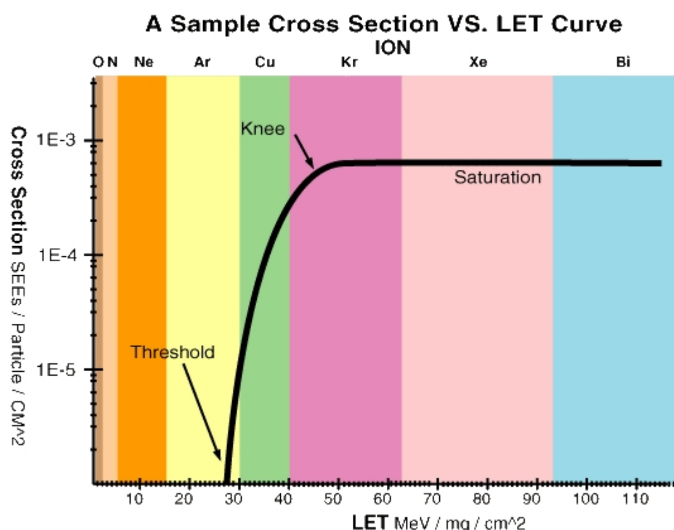


Figure 2.6: Cross section vs. LET curve. Y-axis represents the probability of having SEE per unit area and incident ion. The X-axis is the ion LET. Source: The Aerospace Corporation.

measuring leakage currents, power consumption or other failures. Early experiences at the Cyclotron facility at CNA [107] (in Figure 2.7) irradiating a programmable logic device resulted in dose damage during dynamic tests. In this case, the output errors were not clearly determined as SEE errors as long as the errors were detected by coincidence with a golden output, instead of using signature analysis to compute the whole error patterns.

2.3.3 Neutron Tests

Fast Neutrons are a major source of soft errors at atmospheric altitudes. Civil aviation systems and even ground-level electronics equipments are susceptible to be affected by these particles originated by cosmic ray interaction with atmospheric molecule. Low energy neutrons interact mainly with some specific elements present traditionally in the semiconductor technology. The most known reaction is that with the ^{10}B isotope in the borophosphosilicate glass (BPSG) of SRAM devices. However, this element was removed from the process flow on the CMOS SRAM 180 nm node [11] so the SEE contribution from neutrons below 1 MeV is negligible in modern technologies.

Neutron source facilities are of main interest for the radiation effects community to study the effects of neutrons in modern technology nodes, as long as ground electronics go deeper and deeper into the semiconductor nanoscale. There are works demanding for a way of validation of SEU or other SEE errors in dy-

Figure 2.7: Cyclotron at the CNA. Protons were accelerated to 18 MeV for a first dynamic radiation experiment on a Xilinx CPLD [107].

namic radiation tests of logic devices as for example [47].

2.3.4 Pulsed Laser Method

Laser testing is a mature technique for SEE fault injection on semiconductor devices. Pulsed laser facilities have been longer used to emulate the effects of ionizing particles present in space or other radioactive environments. Probably the main advantage of these facilities is the reduced cost of use when compared with particle accelerators, and also the controlability of the experimental conditions. In the Single Photon Absorption (SPA) approach, the laser can be focused to a small area (approx. 1μm) in the circuit under test to study hot spots in the layout. However, metal layers are highly reflective and in technologies with more than 2 or 3 metal layers the irradiation is performed from the bottom side of the silicon die, exploiting non-linear interactions with Silicon via Two Photon Absorption (TPA). Hence, SPA is only used for very old technology nodes. The sample preparation for the TPA approach is not straightforward and is possibly the main drawback of the technique. Nevertheless, these techniques are of extensive use in different researches as in the study of charge sharing or the PIPB effect through logic gates, for example.

Signature analysis is used in [103] to diagnose faults and evaluate mitigation strategies in a high speed bulk Si/Ge technology under laser irradiation. This technology features high robustness against total dose damage, but it is actually very

sensitive to SEE errors, so hardening by design strategies must be considered and tested. In this context, diagnostic during irradiation is necessary. Other laser experiments to generate dynamic cross sections in digital ASIC were carried out in [85]. Here, software VHDL simulation was used after the experiments to analyze error patterns and identify SEU faults in the circuit under test.

2.3.5 Ion Beams

Ion beam test is the definitive validation procedure for microelectronic components against in-orbit heavy ion radiation. Light ions like alpha particles and higher-Z species are also used in research to reproduce the effects of heavy ions effects on electronics.

The following text is an excerpt from ESAPSS_01_609 section 19.2.1 referring to the simulation of space radiation in ground facilities like particle accelerators:

"The differences between radiation conditions in space and their simulation in the laboratory are frequently quite large[...] one must therefore modify the raw short-term results of "simulation" tests when converting them into predictions of space radiation conditions. One may also wish to monitor the irradiated device at intervals of minutes, days and months and introduce factors to allow for qualitative differences in space and laboratory radiation[...]

Despite the above mentioned problems, a surprisingly wide range of devices can be tested suitably by the use of high-activity gamma-ray-emitting isotopes, e.g. the well-known cobalt-60 hot cell or "irradiator". However, one must always classify carefully the physics of the effect one is aiming to simulate - whether it is classed as a total-dose ionization effect, permanent bulk damage effect single event upset or a transient effect of some other kind.»"

This text summarizes some of the difficulties that the radiation engineer faces when planning a radiation experiment in the context of ASER techniques. First, results from accelerated tests must be extrapolated to achieve confidence with the real error rate expected in orbit. Second, the effects of radiation on semiconductor devices are very different in their nature and consequences, and depending on the effects investigated, it is necessary to establish the proper conditions during the experiment, mainly to avoid physical damage in the chip and dose rate or TID effects.

Eventually, in-beam experiments are carried out in vacuum to avoid ion interaction with surrounding molecule. Devices under test must perform in an environment with no heat dissipation on air, so temperature must be monitored if it is expected to consume relative high currents during the experiment.

2.3.5.1 Broad Beam

Broad beam is generally used for part qualification to have global statistical information on device dependability. Broad beams require low focusing, so there are few restrictions imposed by magnetic lenses on the ion LET, allowing the highest LET values and fluxes of particles. Heavy ion experiments are normally performed in broad beam, and diagnostic on soft errors can be useful to find hotspots in a wide exposed area, and also to guarantee that the observed errors are SEU or other SEE errors.

2.3.5.2 Microprobe

Broad beam experiments with heavy ions are usually used in part qualification testing against SEE, having generally the response of the whole component to the radiation. However, for diagnostic purposes it is useful to control approximately the beam location, and it is achieved by using nuclear microprobes. However, new technology dimensions are shrinking and very often focused area covers several cells in the layout. In this cases, diagnosis by signature analysis can add valuable information about fault location for SEE.

2.4 Proposed Methodology for Soft Error Testing

Accelerated radiation experiments for soft error analysis are usually performed to detect the vulnerabilities of a given component operating in a harsh radiation environment. These experiments are usually costly, time-consuming and complex and it would be desirable to get as much data as possible from beam sessions. Dynamic radiation testing introduces extra complexity to the test regarding the time dimension. Data flow processing for error detection is not straightforward in most cases and a testing protocol with a set of *best-practices* is required to enhance experiment observability. This thesis comes to contribute with a procedure involving fault injection campaigns and data analysis during radiation experiments, to bring closer the roles of the designers and radiation engineers.

Figure 2.8 is a flow chart with the main steps in the proposed methodology. The different blocks are deeply explained and proved in further chapters, but here a global introduction is done.

The idea behind this methodology is to perform a SEE *cause-effect* analysis on the design under test, generally previous to the radiation experiment, to record the expected behavior of the circuit and generate a database or fault dictionary to be used later for diagnosis. The starting point in the whole methodology is the netlist or HDL description of the design to be tested. Attending to the flow chart in Figure 2.8, there are two branches well defined corresponding to the emulated fault

injection campaign and to the radiation experiment. Each branch is independent in time and can be completed separately. Of course, the test vectors are shared for the two branches, to apply exactly the same set of stimuli to the DUT during the emulation and radiation experiment.

Fault injection is performed via hardware emulation covering the whole design or just the blocks to be tested later during the radiation experiment. A fault model have to be considered for a first generation of a soft error database (or fault dictionary) according to the expected experimental conditions. By simplicity, a first fault dictionary for SEU-like errors can be generated using the bit-flip model. Faults are induced systematically and circuit outputs are processed to generate signatures at each round of injection or “run”. In the fault dictionary, a single signature is related to a given fault.

The radiation experiment is performed on a prototype from the design netlist (Integrated Circuit). An Automated Test Equipment (ATE) controls the test by applying continuously the set of stimuli and recording the outputs while the device is running and irradiated on the accelerator beam (or other radiation facility). Outputs are processed in the same way as during the emulation fault injection campaign to get the signatures. For the output failures, the resulting signatures are looked up in the fault dictionary. In the event that signatures are all present in the dictionary, all the errors can be diagnosed. In other case, a new fault dictionary must be generated considering for example SET-like or MBU-like errors. This procedure is the main contribution of this thesis.

There also exists the possibility of using the technique in a *effect-cause* analysis. Instead of generating an exhaustive fault dictionary, the radiation test can be carried out and failure recorded to later investigate the causes by fault injection in the hardware emulator. Knowing the experimental conditions can help to reduce the candidate nodes to be injected, and also the fault model to be used (SEU, MBU, MCU, etc..).

In addition to reliability analysis of ASICs, there are a wide variety of studies involving Soft Errors induced by radiation where fault dictionaries and signature analysis could be applied. Programmable devices are generating rising interest from aerospace industry. Regarding radiation of programmable devices, and concerning this methodology, the limitation is in the configuration state of the component. Configuration must be non-sensitive to radiation, as for example in the One-Time Programmable (OTP) or flash-based FPGAs. Flash-based FPGAs configuration switches are non-sensitive to SEE but not so the logic associated to the implemented design, which is vulnerable to soft errors [95], this way radiation of these devices keeps functionality safe and errors come only from the design implemented. Flash-based FPGAs are an interesting framework for dependable designs to be tested in-beam before fabricating the final prototype. SRAM FPGAs are a particular case considered for future work in Section 7.2, where the configuration

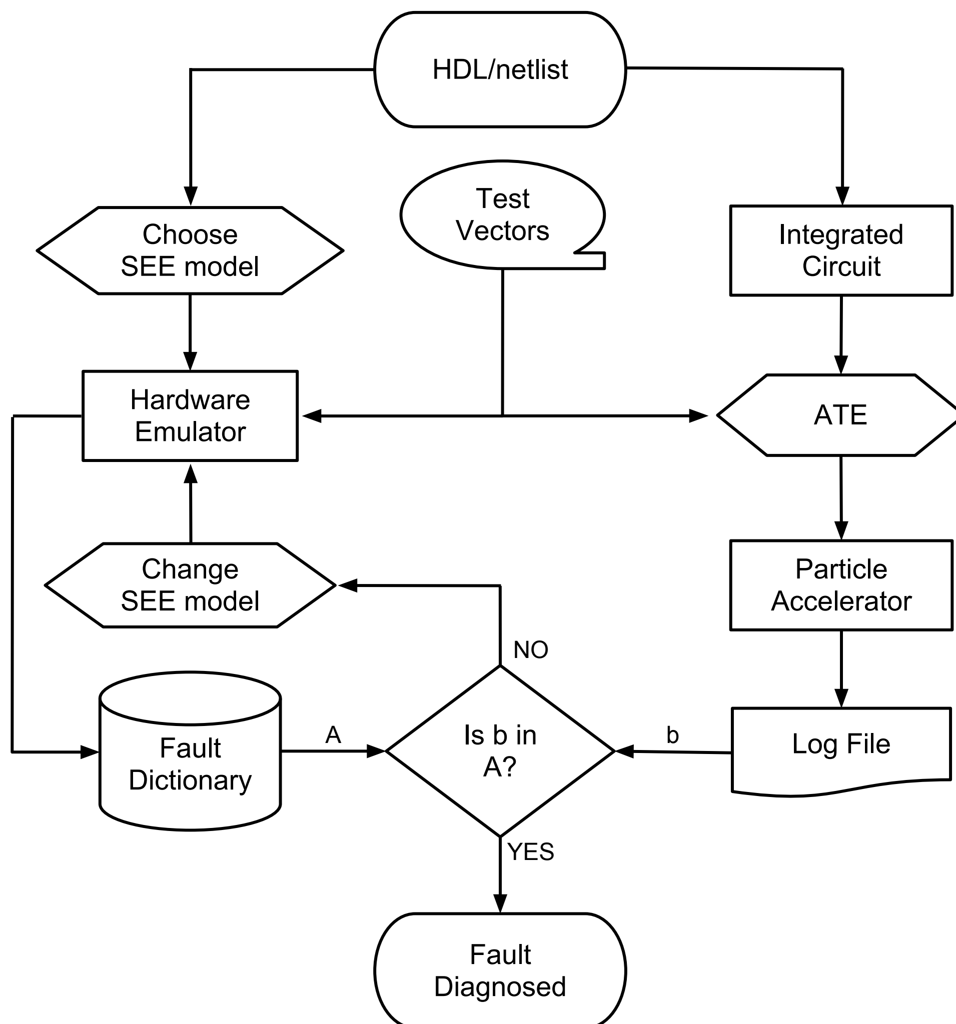


Figure 2.8: Flow chart of the proposed approach.

state is sensitive to radiation.

Another field of study where detection and diagnosis can be of interest is in SET mitigation topologies in antifuse FPGAs like the Actel RTAX-S devices. These OTP FPGA devices present SEU-hardened user registers with Triple Modular Redundancy (TMR) protection. However, ion beam tests have demonstrated they are vulnerable to SET even for frequencies down to 6 MHz [96]. In this case, soft errors are induced in form of SET that are sampled at the user registers and can be cataloged.

If the layout of the DUT is accessible, microprobe analysis can be applied in a given topology to find the error dependence on experimental variables as temperature, bias voltage or flux in a similar way as in [100], signature analysis here would provide a failure criteria.

2.5 Remarks on This Chapter

Along this chapter, a number of simulation and radiation testing techniques have been discussed that are relevant for the rest of this thesis. These techniques also complement the methodology introduced in this chapter, which is the main contribution of this thesis. There is a number of journal and conference papers coauthored by the author of this thesis that turned to be crucial for the development of the ideas and experiments that are the basements of the thesis.

SEE have been simulated using TCAD models in [33] and the SEU threshold is investigated and experimentally verified in [88].

Laser experiments have been helpful to estimate some critical experimental parameters and design vulnerabilities. A thorough study on the capabilities of TCAD to simulate ionization by pulsed laser is done in [86]. A method to estimate the critical charge for a CMOS technology in laser experiments is explained in [82] and also the simulation of a bit-flip induced by pulsed laser in [87].

One step forward, in [84] a complete laser test-bench is set up for the analysis of SEE and a cross section analysis is done in [83].

Transient propagation have been also investigated by TCAD simulation in [68] to measure the broadening effect in a 130nm CMOS technology. Results are valid for the 130nm CMOS technology irradiated in chapter 5.

CHAPTER 3

Signature Analysis

“Regla de Oro:
Todo Flip-Flop debería tener un reset,
y solamente Flip-Flop puede tener reset."
– Jonathan Noel Tombs

Contents

3.1 Introduction 69
3.2 Use of Signature Analysis for Soft Error Testing 71
3.3 Signature Module 73
3.4 Hash Functions 75
3.5 Fowler-Noll-Vo Versus RIPEMD160 Hash Functions 76
3.6 Remarks on This Chapter 81

3.1 Introduction

This chapter is introducing Signature Analysis as a way to detect and diagnose circuit failures and its application in radiation tests. The applicability of this technique for diagnosis of consumer electronic equipment has proven to be successful from long ago, with the advent of test tools addressed to electronic designers instead of quality assurance engineers [43].

Recording failure signatures is recommended in the standard JEDEC JESD89 about soft errors induced in semiconductors, in Section 3.6 entitled “Data Collection”. The state of a digital circuit is fully deterministic for a given set of stimuli

and initial state. This feature makes it possible for logic circuits to establish permanent cause-effect relationships between the observed outputs and the internal state of the circuit.

Considering a classic *Moore* or *Mealy Machine*, fault injection in a logic circuit in general can be considered as an additional stimuli that changes the internal state of the machine putting the circuit on an unwanted state. This can be expressed with the following equations:

$$S(t+1) = \Theta(S(t), I(t), \xi(t)) \tag{3.1}$$

$$O(t+1) = \Pi(S(t), I(t), \xi(t)) \tag{3.2}$$

The internal state of the digital circuit is given by $S(t)$ and the corresponding outputs are represented by $O(t)$. The function modeling internal faults is $\xi(t)$, with normal circuit operation corresponding to $\xi(t)$ being non-present or null. For a given input stimuli $I(t)$, signature analysis is performed by processing the sequence of $O(t)$ corresponding to a set of faults modeled by $\xi(t)$.

A tradeoff exists in signature analysis to produce small enough signatures while keeping low the probability of having repeated signatures for different sequences of $O(t)$. In other words, the smaller the signature is, the greater the probability of having repeated signatures for different sequences of $O(t)$. Another key point is finding the proper model function $\xi(t)$ for the expected faults during radiation experiments.

Minimizing signature size is mandatory for medium-large circuits with large sets of stimuli, in order to reduce memory usage. For example, as can be read in [3], for a campaign of up to 2 million faults injected on the unprotected Leon2 processor, just 17% of these runs were detected at the outputs for a given test bench with 275000 vectors. In this case, the signature database would consist of 340000 signatures indexing the faulty runs, which means roughly 6 MB of data. This amount of memory is negligible when compared with the amount of data digested by the signature function during the whole campaign, close to 256 GB.

The fault model used to emulate the effects of radiation before the radiation experiment must be carefully chosen in relation to experimental conditions. For the study of soft errors in digital circuits, the simplest $\xi(t)$ function at the Register Transfer Level consists of inducing single bit-flips in the design, to emulate the effect of a particle hit affecting only a given FF.

This chapter is devoted to describe the use of a signature function in the normal work-flow of hardware emulators and radiation test equipment. The main ideas were exposed in the RADECS Conference held in Seville in September 2011 [70].

3.2 Use of Signature Analysis for Soft Error Testing

The use of signature analysis for error detection and diagnosis in electronic circuits, components or equipments is usual for the test community. In the field of radiation effects on component and systems, there are works in the use of signature analysis for detecting and diagnosis of errors in electronic affected by SEE as in [80], addressed to find errors in the program control flow of a microprocessor system, or involving error diagnosis in SRAM FPGAs [75]. The concept underlying to signature analysis is the generation of unique and simple codes that identify certain amount of data.

In dynamic radiation testing of digital circuits, the data flow at the outputs of a digital circuit during normal operation can be processed by a signature function to generate a unique code or signature. Under normal operation, in absence of errors, the output data pattern will be always the same, and so the signature associated. However, output soft errors modify the error-free data flow and can be associated to a different unique signature.

The two basic requirements for a given signature function to be used for error detection and diagnosis are the following:

1. Determinism
2. Collisionless

A given data flow or data buffer must always generate exactly the same signature, so it must be fully *deterministic*. Furthermore, for exact diagnosis, the signature function should be *collisionless*, that means that different data flows must produce different signatures. No signature function can guarantee to be completely collisionless, but the probability of finding collisions can be reduced extremely in order to achieve confidence in the uniqueness of signatures. Figure 3.1 depicts signature generation for an error-free data flow (top) and for a data flow containing soft errors (bottom).

First experiences in Spain with an ultra-short pulsed laser[1] used to induce soft errors dynamically in a digital circuit [85], showed that the cause of failures should be unambiguously identified for a proper assessment of the origin of errors. In static radiation tests of SRAM memories, it is straightforward to determine the location of injected faults from the observed errors, but dynamic testing produces eventually complex failure patterns that must be correlated carefully with an internal fault or upset.

Figure 3.2 is a screenshot from a Modelsim simulation of the aforementioned pulsed laser experiment. This was the first approach to identify observed errors af-

[1]Faculty of Physics, University of Salamanca (Spain) [23]

Figure 3.1: Figure on the top represents the normal data flow from the DUT output port and the associated hexadecimal signature or *golden* signature. Figure on the bottom represents the circuit operating under radiation, an erroneous data flow (the error pattern is shadowed) and the associated *anomalous* signature.

ter laser irradiation, consisting on recording data in the laboratory and later comparing the error pattern obtained to a software simulated fault injection. The test platform for these experiments was based in a FPGA-based coincidence detector known as FTUSB, recording discrepancies between golden and faulty outputs during irradiation.

The waveforms in Figure 3.2 show the detection of an error in three outputs of the target, and the time it'd take for the coincidence detector to record the events. As can be seen, an error appears firstly at *SR out* and later in the simulation, two errors appear simultaneously at *PS Out 1* and *PS out 2*. After an error is detected, several FPGA clock cycles are required to process and record the necessary data about the error while the DUT is paused (see how the corresponding waveform for the *Iteration cycle* is frozen in some intervals). This mode of operation is not the best for dynamic testing because during these time lapses, the circuit is irradiated statically so the test is not completely dynamic (of course, the time the circuit is paused is relatively small in comparison to the complete set of stimuli). It would be much more appropriate to use the time between runs to record error data and avoid interruptions in the normal execution of the DUT.

For this purpose, signature generation does not interfere with the normal operation of the target circuit and signatures are produced on the fly in an external module by processing cycle by cycle the outputs coming out the circuit. There-

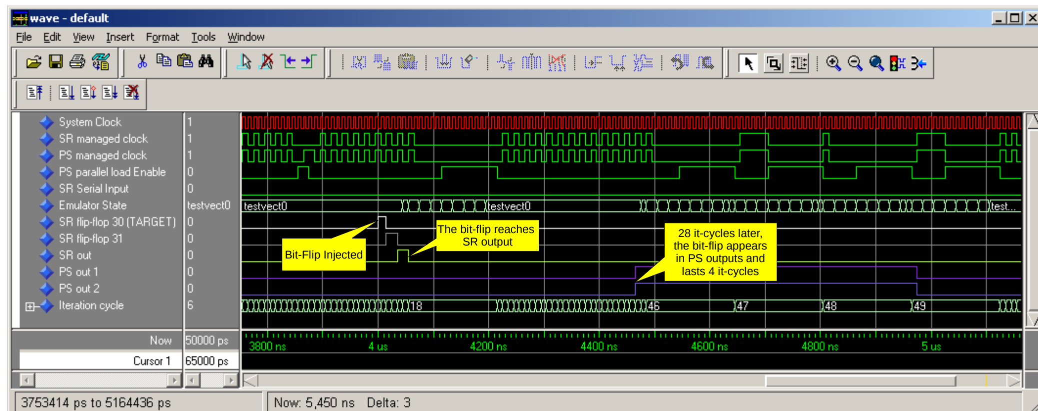


Figure 3.2: Modelsim simulation of a bit-flip induced by pulsed laser irradiation of a digital circuit [85]. The method for error detection in this case was a coincidence detector designed to be a reduced version of FT-UNSHADES [3] for radiation experiments.

fore, the test is never paused and circuit errors are always induced dynamically. Signature generation provides information about the whole error pattern no matter how many cycles it takes, and the number of bits required per signature is fixed.

3.3 Signature Module

There are many different algorithms for signature generation that could be apparently well-suited for a given device and set of test vectors. However, together with the basic requirements noted in the previous section, it is necessary to observe a number of recommendations in order to get the expected results:

1. Small Signatures
2. Reduced Hardware Requirements
3. Low Propagation Delay
4. High Data Rate

Generation of exhaustive fault dictionaries normally requires the storage of a large amount of signatures, therefore it is recommendable for the signature module to produce signatures as simple as possible. Of course, the problem with small signatures is in the collision probability because larger signatures are statistically more robust to collisions. Another point is the hardware requirement for the implementation of the signature module. For the signature analysis methodology to be

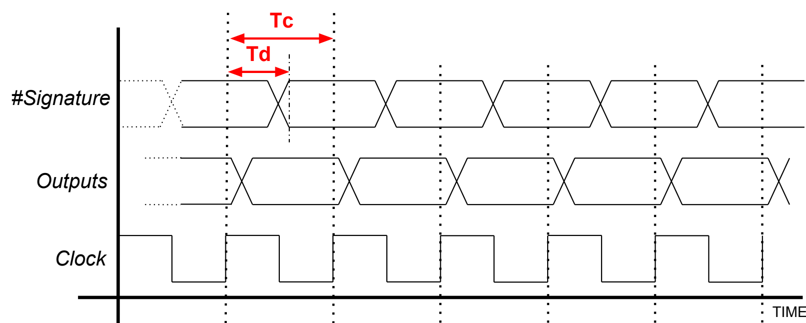


Figure 3.3: Signature time delay Td compared to the system clock period Tc.

easily implanted in existing hardware emulators and ATEs, it is recommendable for the signature module to be tiny and portable. For the use of FT-UNSHADES, the module must be implemented in the system FPGA sharing resources with two instances of the DUT and some extra logic, so the area overhead must be minimal.

The propagation delay is the amount of time it takes for the signature function to refresh the signature value from the current circuit outputs. The outputs from a digital circuit can change every clock cycle, so the propagation delay must be shorter than a clock cycle. Figure 3.3 illustrates this situation by comparing the propagation delay (Td) to the clock period (Tc). The propagation delay for the signature is short enough to have a stable signature after each clock cycle, so the system can operate to the rated clock frequency.

The propagation delay could be a limitation for the system clock, specially in the study of SET errors when the clock frequency has to be relatively high. Hence, for SET studies, propagation delay should be a major concern in the election of a signature module.

Another recommended feature for the signature function is having high data rate. This recommendation is for circuits having a high number of output ports, to preserve the clock frequency. The problem of having multiple output ports can also be solved by placing in parallel several instances of signature functions with low number of inputs.

An additional recommendation, related to a possible future line of study for the signature module, is the suitability to be hardened by design against radiation, because in future test vehicles the module could be implemented in silicon as part of a Built-in Self-Test (BIST) strategy.

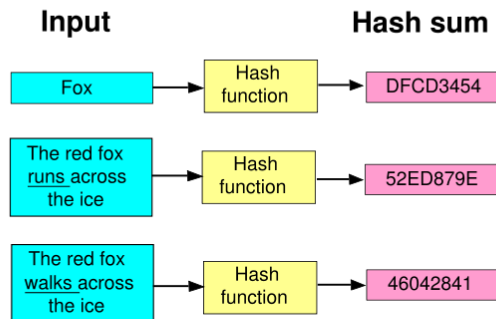


Figure 3.4: Graph illustrating some features of hash functions.

3.4 Hash Functions

Among the possible candidates to implement a signature module in the workflow of a radiation test equipment or hardware fault injector, hash functions are here proposed to be a recommended choice. Nevertheless, the use of other approaches based on Cyclic Redundancy Check (CRC) codes for example, it is not discarded and it is not the purpose of this thesis to propose the best possible solution, since it can be very dependent on the specific experimental conditions (circuit complexity, set of test vectors, number of outputs, SET analysis, etc...). Hash functions are powerful and in many cases introduce no significant area overhead in FPGA designs. Moreover, there are a variety of algorithms for different characteristics of the circuit to be tested. Hash functions are widely used to check data integrity in large files, data encryption, table look-up and others. There are also plenty of work available about hardware implementation and optimization of hash functions for FPGA devices, which is generally the main component in hardware emulators.

Figure 3.4 shows some features of hash functions that are relevant to be used as signature module for soft error detection and diagnosis. As can be seen, no matter the size of the incoming message, the resultant signature (hash sum) has a fixed size. Hence, the hash signature is not dependent on the size of the set of stimuli.

Hash functions are designed in most cases to provide very different signatures even when the output error sequence is similar to other, specially in the case of cryptographic hash functions. This feature may help to distinguish between signatures even for data streams that are very similar. A single bit change in a megabyte of data processed from the circuit outputs can give place to a very different hash code.

There are several hash algorithms in the bibliography suitable to be used for signature generation in the context of radiation testing of electronic components.

Among the cryptographic hash functions, we can find MD, SHA or RIPEMD [98], offering high robustness against collision. Another suitable non-cryptographic hash functions are Pearson hashing, Jenkin or Fowler-Noll-Vo (FNV) hash functions [55].

Collisions in hash functions are found when two or more different hash inputs (messages) give place to the same hash code. If it happens when generating the fault dictionary, the resulting hash table is no longer an 1-to-1 relationship, and the fault dictionary would offer a list of fault candidates causing a given hash code. Of course, two identical output data sequences coming from different fault injection runs will also lead to the same hash code.

To estimate the probability of collision in an ideal collision resistant hash function, one may consider the *birthday paradox* to keep in mind that the probability of finding a collision is around 50% for $2^{Nb/2}$ messages digested, where Nb is the number of bits of the hash code. That means that a hash function producing 32-bit wide hashes allows 64k messages to be digested with a 50% probability of finding one collision. Then a 64-bit wide hash function has the same probability for up to 4G messages which should be enough for whatever design and test. Small designs and small sets of test vectors will require a lower Nb than complex designs.

This thesis introduces the use of hash codes to index pairs of time and location for each injected error, so that the hash function has to guarantee that there is no collision for the fault injection campaign. There are powerful but simple enough algorithms that can generate a hash code using a very reduced amount of circuitry. The minimalist FNV hash function [81] is proposed for the practical examples in this thesis as proof of concept because it complies the requirements of simplicity and speed. The main advantage of this algorithm is that it is fast and can be implemented in a small FPGA using very low hardware resources. We use here the FNV-1 variant to produce 32-bit hashes, the input is one byte wide and it can digest one input per clock cycle. For circuits with a higher number of outputs the basic FNV cell can be cloned to process a higher number of bytes and generate a wider hash value just joining the outputs of all the cells.

Following section is a comparison between two different hash function implementations in a FPGA to see the convenience of using one or another.

3.5 Fowler-Noll-Vo Versus RIPEMD160 Hash Functions

The non-cryptographic FNV hash function is a very simple but powerful algorithm for mapping input messages to 32 or 64 bit codes. The input port is 8-bit wide and it produces a new hash code at every clock cycle. Robustness against collisions

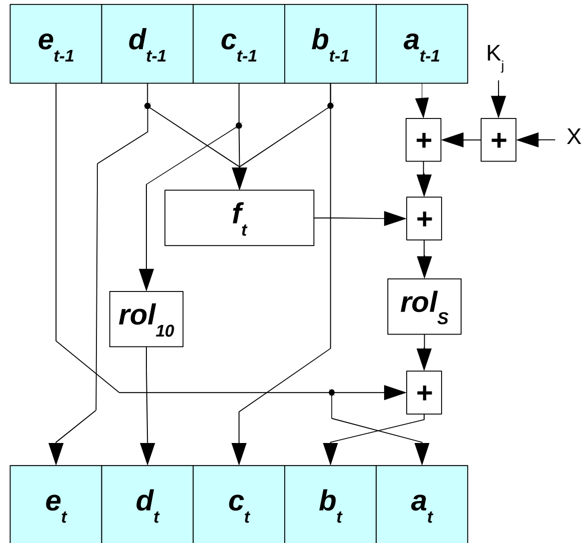


Figure 3.5: A stage of the cryptographic hash function RIPEMD160.

is not high and depends statistically on the number of incoming messages. In the context of fault injection campaigns, that is to say the number of runs required to complete the fault dictionary.

On the other side, the cryptographic hash function RIPEMD-160 is rather more complex, with several stages and robust against collisions. RIPEMD-160 is a strengthened version of RIPEMD. The hash codes generated are 160-bit wide, which is 16 or 8 times the size of the FNV codes. The input port is 512-bit wide, so it can be connected to the outputs of a digital circuit with up to such a number of output ports. In this case, hash codes are not ready in just one clock cycle, and the inputs must be present for a number of clock cycles. This is a limitation on the maximum clock frequency for the circuit under test, but can be saved by buffering if the number of output ports in the circuit is well below 512.

This two examples are probably antagonist, and are discussed here to show two different solutions for different requirements. For a simple circuit with a relatively small set of test vectors, the FNV option can work properly and is easily implemented. Complex designs with a heavy workload could require the use of a better performance hash function to guarantee no collisions. Anyway, selection of the hash function shouldn't be critical.

The algorithm code for FNV is as follows:

```
hash = FNV_offset_basis
```

```
for each octet_of_data to be hashed
    hash = hash x FNV_prime
    hash = hash XOR octet_of_data
return hash
```

where FNV_offset_basis is the seed of the algorithm, FNV_prime spreads any small change in the input among the output hash bits and $octet_of_data$ is the input byte from the message to be hashed. This algorithm has proven to be effective for signature analysis for radiation experiments as seen in the following chapters. The FNV algorithm starts with a fixed offset value that is multiplied by a seed word. The result is then XORed with the incoming 8-bit word to get the hash code. This hash code is used as offset for the next word, so the final hash depends on the complete sequence of data coming from the DUT every clock cycle.

A more complex implementation for a hash function is the cryptographic RIPEMD-160. The algorithm in this case is given by a number of successive stages described by the following equations and the graph in Figure 3.5:

$$
\begin{aligned}
e_t &= d_{t-1} \\
d_t &= ROL_{10}(c_{t-1}) \\
c_t &= b_{t-1} \\
b_t &= e_{t-1} + ROL_S\left[f_t\left(b_{t-1}, c_{t-1}, d_{t-1}\right) + a_{t-1} + X_i + K_j\right] \\
a_t &= e_{t-1}
\end{aligned}
$$

Here, ROL is a Rotation-Left Operation, f_t is a boolean function, K_j are constants and X_i is a 32-bit word from the 512-bit input. See Appendix E for the VHDL implementation of the algorithm developed for this study.

3.5.1 Implementation on an FPGA Device

Table 3.1 shows an implementation comparison of the FNV and RIPEMD-160 algorithms for different FPGA devices. All the data has been obtained using post Place&Route analysis reports. The use of resources is minimal for the FNV hash function as can be observed, with good timing specifications. It must be noted that the number of FF required for the RIPEMD-160 in Virtex-2 platform has been obtained with a previous version of the Xilinx synthesizer, which can explain the area increase respect to other platforms. Although FNV is less robust against collisions, it is more suitable for small designs because the implementation is much smaller and it is faster, so it was the hash function selected for the fault campaigns and radiation experiments. Moreover, although each FNV core can only process one byte (8 outputs), it could be cloned to be used with designs

with a higher number of outputs as a “compromise solution”. The VHDL for the RIPEMD-160 pipeline architecture was developed from the description done in [65] and as can be seen has the highest bit rate, so it is specially suited for a large number of output ports and complex designs. Figure 3.6 shows the schematic after the Xilinx synthesis tool for the FNV module. The main blocks are the multiplier (top center) and the XOR gate (bottom left).

3.5.2 Pros and Cons

As mentioned before, the selection of the function to hash the output data flow from the DUT is not critical. However, there are a number of points that must be observed before making the choice among the comprehensive set of hash functions available.

First, it is necessary to estimate the probability of having collisions due to the hash algorithm. This probability is closely related to the spectrum of possible faults to be injected in a given design and the workload. Of course, the higher the number of FFs and clock cycles, the more complex must be a hash function to be collisionless. The birthday paradox is a rough estimation to evaluate the probability of collisions.

Hash functions with a uniform distribution of hash values and low number of bits (32-bit hash codes) can be valuable for small-medium designs and a rather small number of test vectors. It is recommendable to select the simplest option complying with the collisionless requirement. The main advantages are in the need of few resources from the configurable devices in the emulation platform. Furthermore, it is easier to implement an VHDL description of a simple hash function. Eventually, when a configurable device is shared by the DUT and the hash block designs, reducing the resource requirements for the hash function implies that more resources will be available for the DUT. In most cases, best timing characteristics are achieved by simple hash functions like FNV rather than by more complex algorithms.

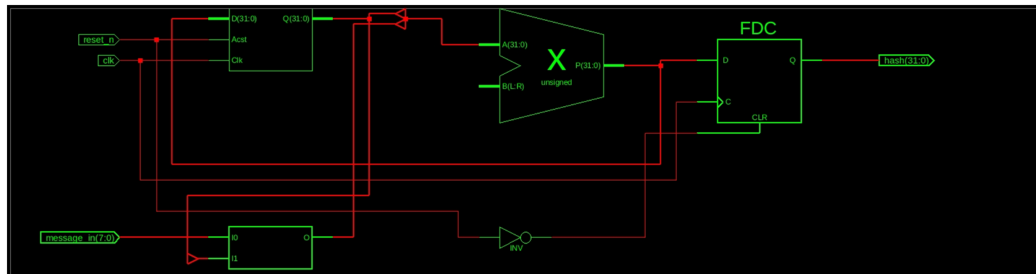


Figure 3.6: Schematic view of the FNV module by the Xilinx Synthesis Tool.

	RIPEMD-160			***FNV***		
FPGA	**No of FF**	**Max. Clock Freq. (Mhz)**	**Bit Rate (Mbps)**	**No of FF**	**Max. Clock Freq. (Mhz)**	**Bit Rate (Mbps)**
Spartan-3 XC3S4000	2119	40.7	1452.8	46	67.8	542.4
Spartan-6 XC6SLX100T	2128	55.3	1768.3	46	65.8	526.4
Virtex-II XC2V8000	2429	48.2	1543.4	46	58.4	467.2
Virtex-5 XC5VFX70T	2113	63.3	2025.3	46	103.5	828.0

Table 3.1: Implementation comparison of two different hash algorithms using Xilinx tools and FPGAs.

When designs under test are large enough, and the number of test vectors make the spectrum of possible faults huge, it could be necessary to proceed with a more sophisticated hash function or even a cryptographic hash function to keep null the collision probability. The problem here is the resources required and also the availability of hardware descriptions for such algorithms. The resource requirements from RIPEMD-160 for example, seem to be prohibitive in case only one configurable device must be shared by the DUT and the signature block.

The RIPEMD-160 algorithm can be fast enough if data flow coming from the DUT is arranged properly. The pipeline architecture implemented to obtain Table 3.1 must be carefully used to take advantage of a high data throughput rate. This fact makes less friendly the use of this kind of hash functions.

It can be good idea to follow a strategy based upon the concept of "divide and conquer" to keep using simpler hash functions. There exists the possibility of dividing the whole design in several pieces to have more than only one dictionary. It could be also of interest to have the dictionary of only part of the design. This strategy can be combined with selective focusing in a nuclear microprobe for example.

3.6 Remarks on This Chapter

A preliminary use of signature analysis was done in [83] at the laser facilities of the University of Salamanca. In that work, software simulation was used to get error patterns after fault injection. These kind of experiments on digital circuits irradiated dynamically, with the circuit running in normal operation, motivated the use of signature analysis.

The first conference paper introducing part of the methodology described in this thesis is [70]. The fault dictionary generation by fault injection is introduced using a hash function to code the outputs from digital circuits. Experimental results were provided lately to validate the methodology and are discussed in subsequent chapters in this thesis.

CHAPTER 4

Fault Injection Campaigns

"If a tree falls in the forest and no one is around to hear it, does it make a sound?"
– Philosophical Thought Experiment

Contents

4.1 Introduction **83**
4.2 Beam Test Mode of Operation **84**
4.3 Fault Dictionary Generation **86**
4.4 Workload Suitability for Fault Diagnosis **89**
4.5 Remarks on This Chapter **92**

4.1 Introduction

This chapter describes the integration of a signature module in the work flow of an specific hardware emulator as FT-UNSHADES, used for the generation of dictionaries of faults, by mean of fault injection campaigns on different digital designs. The process corresponds to the left branch on the complete flowchart of the whole methodology, as can be seen in figure 4.1.

Four different digital designs have been used here to generate fault dictionaries that could be used for diagnosis in radiation experiments. One of these designs is used in further chapters for radiation testing.

Faults dictionaries are analyzed statistically to evaluate the suitability of the workload for fault diagnosis, which depends on the number of faults that are exactly identified by a single signature. This criterion for workload evaluation before

the radiation experiment is a contribution of this thesis, providing a way to determine the set of test vectors more appropriate for a radiation experiment, in order to avoid fault masking and to guarantee a high proportion of errors being perfectly diagnosed.

4.2 Beam Test Mode of Operation

Another contribution of this thesis is the Beam Test Mode of Operation of the FT-UNSHADES [3] emulator, developed to use the emulator as an ATE in a radiation experiment. The work flow for this new mode of operation is exactly the same than that for the generation of the fault dictionary, so that it is introduced in this chapter.

In normal operation, FT-UNSHADES implements two replicas of the design in the system FPGA, one to be injected and the other for cycle-by-cycle comparison. Fault injection is performed by read-modify-write of the FPGA configuration bits by mean of partial reconfiguration of the FPGA. During each run of the test vectors, the system clock stops and freezes the circuit on the injection selected clock cycle, then the system reads from the configuration memory of the FPGA, and sends the bit-stream to a host application running in a computer. The data is then modified to inject a bit-flip in a desired FF by software, and afterward it is downloaded to the FPGA to resume the run. In the event of an error detection by comparison with a golden replica, the test is paused and results are sent back to the computer, that proceeds with a new injection run. For each injection, the system resets the circuit and starts again all the steps until all the faults have been injected.

Figure 4.2 represents the original process flow of FT-UNSHADES with the extra modules required for signature generation. The hash function can be easily synthesized in the system FPGA to digest the outputs coming from the SEU-MUT. The corresponding dictionary is processed in the computer with the recorded signatures and annotations about the clock cycle and injected FF for each run.

It is relevant to note that, in the context of signature analysis, cycle-by-cycle comparison is not relevant since failures are expected to be identified by the corresponding signature at the end of the workload. This implies that the system FPGA doesn't need the GOLDEN-MUT in figure 4.2, saving a considerable amount of resources.

In the Beam Test Mode of Operation, the test is never paused but executed completely. The outputs from the design being injected are processed in a signature module and the resulting signature is then read by the computer. The way to proceed with the fault injector is to perform *blind* systematic campaigns for full coverage of registers and time, required to get a comprehensive dictionary of

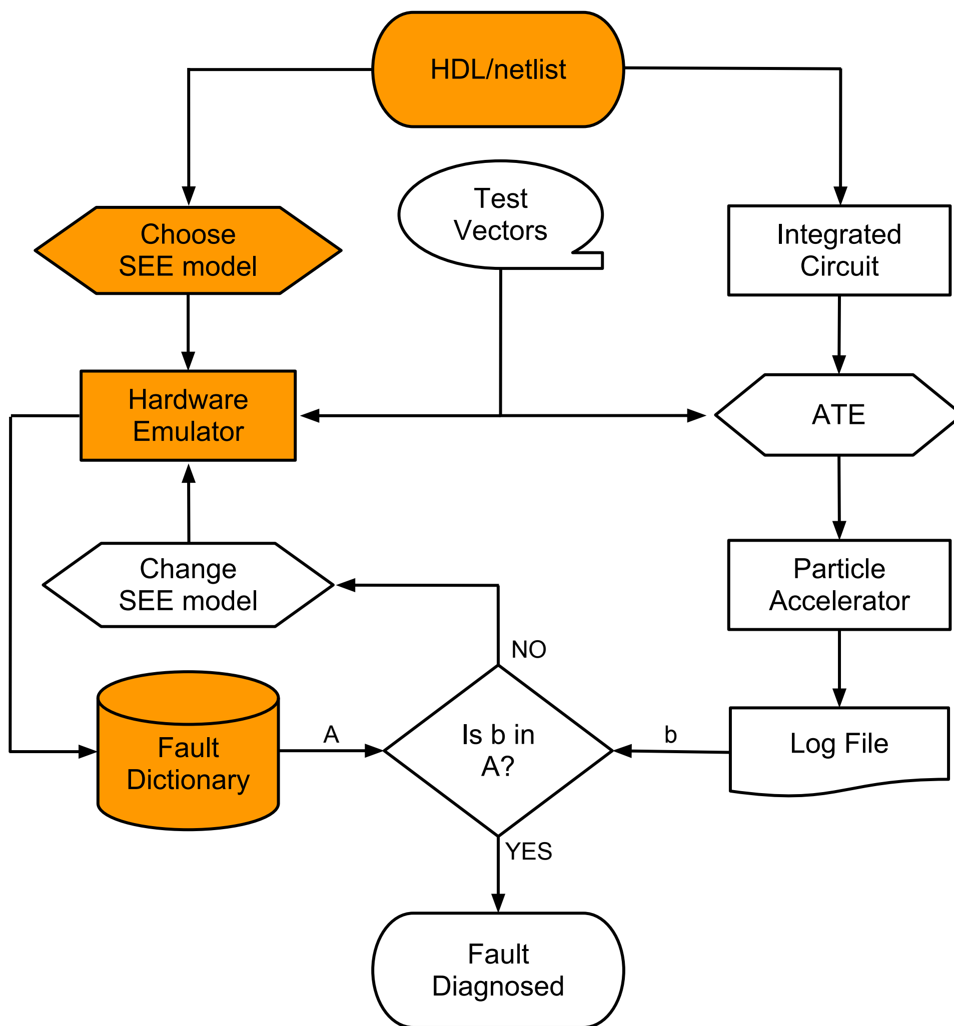


Figure 4.1: Flow chart of the proposed approach. The left branch corresponds to the fault dictionary generation.

faults. In future versions of the emulator, the fault dictionary should be stored in hardware to avoid transactions with the application running on the computer.

In a radiation experiment, the SEU-MUT block corresponds to an integrated circuit operating under radiation, so that there are no need to implement an instance of the circuit in the system FPGA. The inputs for the Module Under Test are applied using the expansion ports of the hardware emulator and so the outputs from the circuit are read by the system FPGA. The outputs from the irradiated circuit are then hashed and signatures recorded as in the dictionary generation. Appendix B is related to the hardware associated to the Beam Test Mode of Operation, that together with the FTU emulator constitute the custom ATE developed for the case study in chapter 5.

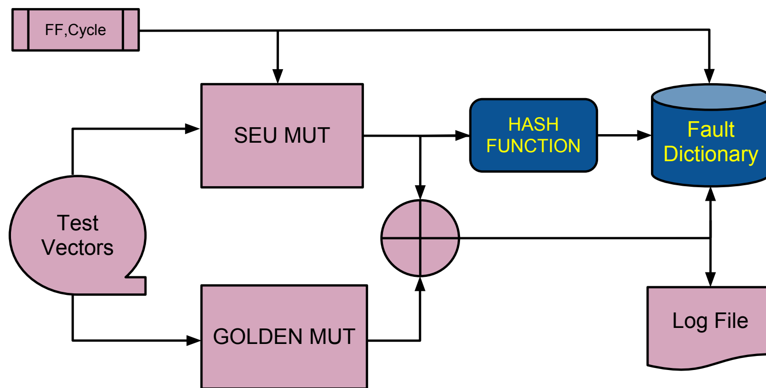


Figure 4.2: Block Diagram showing integration of the Hash Function in the classic architecture of FTUNSHADES

4.3 Fault Dictionary Generation

The methodology proposed in this thesis is intended to provide ASIC designers with an experimental procedure to evaluate mitigation strategies in the final device, performing a real particle accelerator test. Error propagation depends on the stimuli at the primary inputs and can only be observed and studied from the patterns generated at the primary outputs. In dynamic tests, failures induced by SEE are normally present for several cycles at the circuit output ports. This output pattern depends on the nature, location and clock cycle of the fault affecting to any internal FF of the circuit. The output data flow is completely deterministic for a given set of input stimuli, if an unique initial logic state can be guaranteed. This means that if the fault does not remain latent inside the circuit, there is an unique sequence of logic values at the primary outputs (or output pattern) provoked by the presence of the fault.

It is clear that an output pattern is not always causally related to an unique given single event, having place in a given location and time. Hence a 1-to-1 relation can not be always established as would be desirable. In these cases, more than one event leads to the same output pattern and disambiguation is impossible by only observing the circuit outputs. This effect is known as *aliasing* by some authors [90]. Anyway, only a bunch of events will be sharing exactly the same output pattern and the origin of the failure can be reduced to that number of SEE candidates, which is a valuable information for designers.

In the presented technique, the whole stream of data coming from the circuit outputs is hashed to generate a unique short signature in real time. Then, as long as a given output stream is related to an internal upset, it can be related to a unique keyword or signature. Latent, silent or masked faults will produce an error-free data stream corresponding to a *golden* signature.

During fault dictionary generation, the whole stream of data coming from the circuit outputs is digested to generate a unique short hash signature in real time. Hashes are the keys to find in the fault dictionary the corresponding origin of the observed error, in terms of clock cycle and affected FF.

In the real test in a particle accelerator, a hash function module is placed connected to the integrated circuit outputs to compute a hash code cycle by cycle, until the end of the workload. The final value is the signature of the corresponding run. This procedure is repeated continuously while the device is under radiation to get a *golden hash* when no faults are present, or an anomalous or *faulty hash* when errors appear at the outputs.

Massive fault injection campaigns over all the registers in the design and clock cycles in the test allow to generate fault dictionaries where the location and time of the injected faults are labeled by the signature values, and stored in a database or fault dictionary. The need for speeding up such comprehensive campaigns make it necessary to use a hardware emulator to achieve the highest fault injection rates. For this thesis, the platform for emulation FT-UNSHADES [3] has been slightly modified to include the beam test mode of operation and the hash module. As a mature tool, it has proven to be versatile for many works involving fault injection in digital circuits.

Autonomous hardware emulators like [57] could be very valuable for the generation of fault dictionaries due to the high fault rates achieved in systematic campaigns. For speeding up these campaigns, a hardware emulator can save the state of the system in a given clock cycle to resume from this state with the next fault, avoiding to execute all the previous cycles before the injection of the fault. This feature is compatible with hash signature generation since the hash calculation can be resumed from the last-cycle hash.

Figure 4.1 describes the proposed approach connecting the main steps in it. As can be seen, there are two independent process flows starting from the hardware

description or netlist of a digital design. The process to the left is the emulation flow, and the process to the right is the experimental flow. As said before, hardware emulation is in charge of performing massive fault injection campaigns in all the registers of the design and in all the possible clock cycles of the test. The basic model for fault injection is the bit-flip model. Test vectors in the middle of the flow chart are shared for the two process flows. The fault injection campaign generates a fault dictionary indexed by a hash value corresponding to each and every single run.

The process to the right is the experimental work flow, starting with a silicon device fabricated from the original HDL description or netlist of the digital design. The ATE is configured and prepared to operate in the accelerator facility to control the target circuit, taking into consideration all the physical and electrical constraints imposed by the beam line. ATE is in addition retrieving all the data from the circuit under test generating in real time the hashes that are recorded in log files.

The model used to inject faults in the design netlist is the bit-flip, corresponding to the change in the logic state of a single FF. This model is appropriate for testing soft errors affecting one or more bits due to the strike of an ionizing particle. Other physical errors can be modeled based on the bit-flip model.

The presented technique is suitable to be used for any precomputed error model. Single Event Upset is represented in this context as a bit-flip in a FF. Single Event Transients are modeled as a worst case where the propagated pulsed is registered by one or several FFs simultaneously. FT-UNSHADES platform supports an extension that evaluates transients [2]. Multi Bit Upset effects can be studied by the injection of several simultaneous bit-flip in those FFs that can be paired by layout proximity, after a detailed study of the circuit.

The HDL implementation of the hash function is connected seamless and non-intrusively in the work flow of FT-UNSHADES. Also, the size of the hash function implemented in the system FPGA is negligible as can be seen in Table 3.1 for FNV.

As commented in previous sections, the FNV hash function has 8 inputs and generates a hash word of 32 bits. This means that, if we had a number n of 8-bit output ports in our design, we would need to attach n FNV hash blocks in parallel and then bring together the 32-bit output ports of each one to produce a unique $n \times 32$-bit hash.

4.3.1 SEU Dictionaries

The fault model to generate a SEU dictionary is a single bit-flip. The most simple fault dictionary can be generated by injecting bit-flips in each and every single FF

in the design to be tested. Depending on the experimental conditions and tested circuit, SEU errors can be the only soft errors appearing.

4.3.2 SET Dictionaries

SET dictionaries can be generated by defining suitable fault models for this type of single event effects. There are some approaches like [2] using FT-UNSHADES to analyze the capture of transients from the cone of logic. Another approach for having fault models for SET dictionaries consists of combining the gate and register transference levels of a circuital description, to model the capture of transients by the sequential logic. This method for multilevel emulated-based fault injection approach is described in [30].

4.3.3 MBU Dictionaries

Multi-Bit Upsets occur due to direct ionization by irradiation at small grazing angles of incidence or by nuclear reactions at any angle by irradiation with protons, alpha particles or neutrons. Ion track can affect several neighbor flip-flops in their sensitive volumes flipping their logical states. A fault model for MBU consists of injecting multiple bit-flips at the same clock cycle in neighbor flip-flops. It is then necessary to have information about the circuit layout to define fault models for MBU.

Figure 4.3 from [106], is a simulation on how a proton lead to MBU by a nuclear reaction near the surface of the semiconductor.

Of course, the use of fault dictionaries other than SEU dictionaries implies the definition of fault models, what can be a tedious work. In this context, for SET or MBU radiation tests, it is recommendable to have high focusing of the ion beam in order to irradiate a relatively small area to simplify the fault dictionary calculation.

4.4 Workload Suitability for Fault Diagnosis

The hashes at the end of the test vectors are recorded and post-processed to filter and collapse identical hashes, resulting in a statistical analysis about fault aliasing. In fault campaigns, a first run is performed without fault injection. The golden hash is then recorded for further comparison. All the subsequent runs in the campaign having the same golden signature are marked as “harmless injection”, and therefore not included in the fault dictionary. After removing harmless injection, all the faults producing the same hash are brought together to form a *group of equivalence*. Of course, the smaller the size of these groups the more unambiguous the relation between faults and hashes is. This analysis has been performed

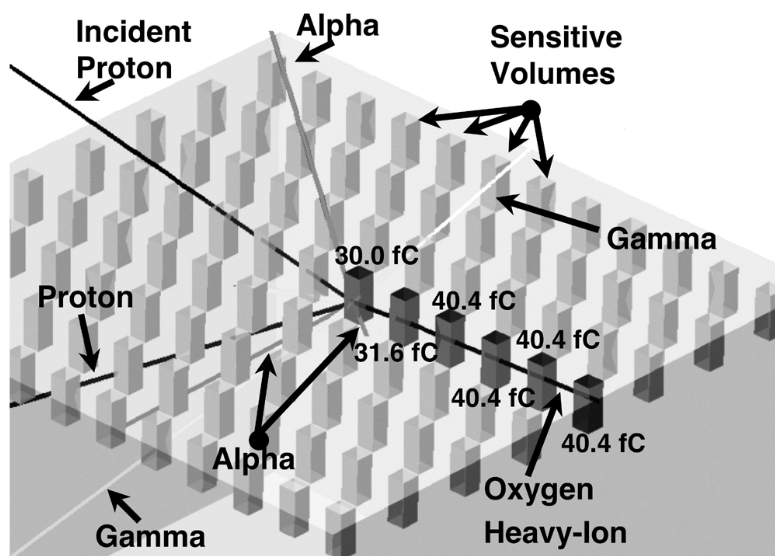


Figure 4.3: Multi Bit Upset simulation by proton irradiation, triggering a nuclear reaction resulting in direct ionization of several sensitive volumes by an oxygen ion [106].

on 4 different designs to observe the statistics in the hash-to-fault ratios or fault aliasing. Hence, this analysis provide a figure of merit for the quality of the test vectors attending to fault aliasing. Some basic information about these case studies is shown in Table 4.1.

Figures 4.4 and 4.5 correspond to fault dictionaries obtained for several programs running on an implementation of the Intel 8051 processor. In this case, faults were induced systematically on the CPU Control Unit at random clock cycles. In all the cases, approximately the 85% of the faults injected were masked or latent faults, and only the remaining 15% were visible on the primary outputs. The fault injection campaigns consisted of 20000 runs in all the cases, which turns to be a reasonable number of injections for representative statistics. Pie charts in figures 4.4 and 4.5 were obtained from this 15% of observed errors. These charts represent the aliasing in the observed errors. The number of test vectors ranges from 166608 vectors for the Elliptical Filter program to 1364936 vectors for the Viterbi algorithm. As can be observed, the worst case for exact diagnosis (1-to-1) corresponds to the “Matrix Product”. In this case, almost 70% of the observed errors are expected to be exactly diagnosed. In the other cases, this percentage is close to 85%. Another relevant information from this charts is the percentage of errors with 4 or more possible candidate faults, which would be desirable to be as low as possible. Again the “Matrix Product” presents the worst case, with 28% of errors with 4 or more candidate faults.

Design	FF	RAM (Kb)	Test Vectors	Inputs	Outputs
INTEL 8051	2403	36	166608 / 1364936	32	32
CORDIC	1179	0	1000	19	32
SPACEWIRE	196	36	5409	29	31
JONIC	67	0	1127	5	4

Table 4.1: HDL Synthesis summary for the digital designs used for fault dictionary generation

Figure 4.6 shows the corresponding results for a Cordic processor, a Spacewire codec and JONIC, the test vehicle targeted in chapter 5. The Cordic and Spacewire fault dictionaries were not generated covering the whole fault spectrum, and so they are not representative of the complete fault dictionary. Nevertheless, a huge amount of faults (100000) were randomly selected to have good statistics for these cases (as for the Intel 8051). Close to 80% of the faults injected resulted in visible errors for the Cordic processor. The percentage of visibility in the Spacewire codec was 30%.

Exhaustive fault injection was performed only on JONIC, the design to be exposed to the beam. In this case, a campaign with 70886 faults resulted in an error visibility of 48%. The corresponding pie charts show that roughly half of the errors at the outputs are caused by an unique fault injection in a given clock cycle and FF, and approximately 75% of the failures are coming from three possible faults or less in all the cases studied.

These statistical analyses provide a *figure of merit* for the visibility and diagnosability of test vectors and can be very valuable when facing a dynamic radiation experiment in a particle accelerator. Fault aliasing can be used as a figure of merit for a given workload, in such a way that a set of test vectors could be discarded if not complies with a minimum percentage of perfectly diagnosable failures. However, it is not the purpose of this methodology to provide the most representative set of stimuli or test pattern as in the case of Automated Test Pattern Generation algorithms. The statistical analysis in this section provides a figure of merit for a set of test vectors if there are several sets available.

It is also remarkable that some of the digital designs tested in this section (CORDIC, JONIC) are all feed-forward architectures with no internal states, memory blocks or feed-back loops. Regarding fault aliasing, feed-forward designs are expected to be the worst case examples. Certainly, in these designs faults do not remain in the design for too long, and outputs are expected to be quite similar for different fault injections. The data flow from primary outputs is expected to be more variable when faults remain in the design long time due to memories

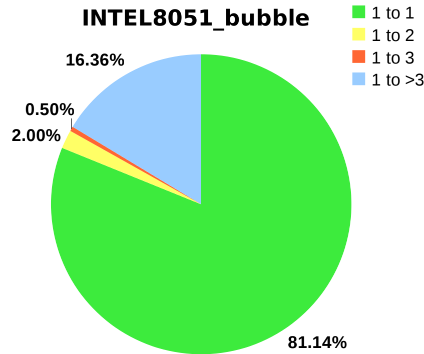
or feed-back loops, thus reducing aliasing, as can be observed in the Intel 8051 processor.

4.5 Remarks on This Chapter

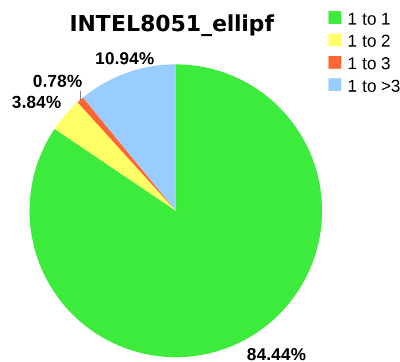
This Chapter is crucial to understand the main contributions of this thesis. Here, the beam test mode of operation for FT-UNSHADES has been introduced, which is extensible to other ATEs. The generation of fault dictionaries indexed by hash codes is an excellent way for hardware fault injectors to assist in dynamic radiation tests of digital parts. Before moving to the accelerator facility, a fruitful work can be done with the fault injector in relative low time (compared to the time it takes the full setup of the beam test). The statistical analysis of fault dictionaries provides a way to qualify different sets of test vectors for a radiation experiment. First, attending to the visibility of the test vectors, and second, attending to the probability to diagnose soft errors.

A number of published works about FT-UNSHADES [76, 77] have contributed to establish the background to develop the main ideas in this chapter. Part of the statistical analysis have been published here [70].

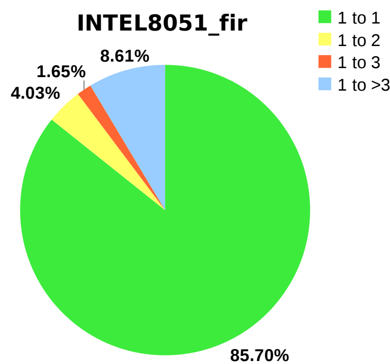
It is worth mentioning that, the need for massive fault injection campaigns have fostered the development of FTU2. An introductory paper about FTU2 was published recently [69].



(a) Bubble Algorithm.

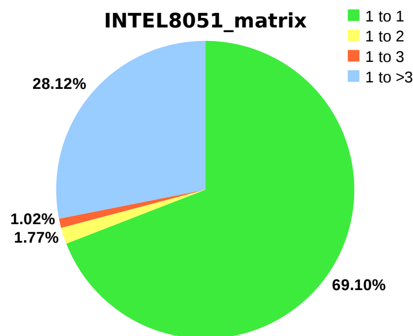


(b) Elliptical Filter.

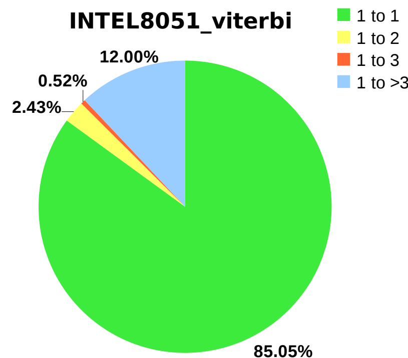


(c) Finite Impulse Response (FIR) Filter

Figure 4.4: (1/2) Failure percentage distribution attending to the number of possible faults involved for different programs running on the Intel 8051 processor. First sector to the right corresponds to failures diagnosed without uncertainty (1 to 1). The number of possible faults involved for a given failure is represented increasing clockwise up to 1 to >3.

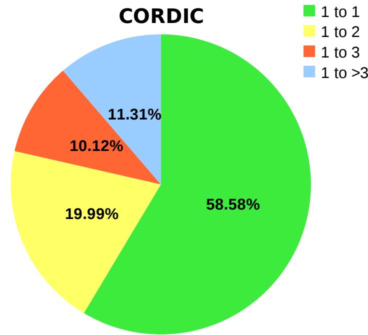


(a) Matrix Product.

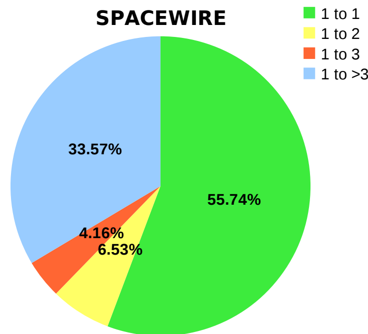


(b) Viterbi Algorithm.

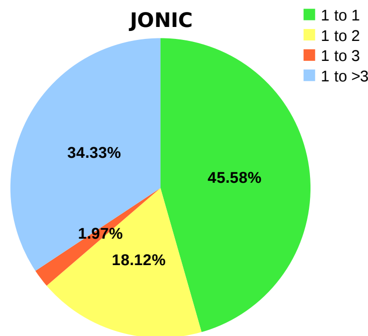
Figure 4.5: (2/2) Failure percentage distribution attending to the number of possible faults involved for different programs running on the Intel 8051 processor. First sector to the right corresponds to failures diagnosed without uncertainty (1 to 1). The number of possible faults involved for a given failure is represented increasing clockwise up to 1 to >3.



(a) CORDIC processor. Almost 60% of the failures are related to a single fault.



(b) SPACEWIRE codec. More than half of the failures are perfectly determined. Failures related to 4 or more faults are around 33%.



(c) JONIC test vehicle. Almost 50% of failures can be related to a single fault.

Figure 4.6: Failure percentage distribution for different designs attending to the number of possible faults involved. First sector to the right corresponds to failures diagnosed without uncertainty (1 to 1). The number of possible faults involved for a given failure is represented increasing clockwise up to 1 to >3.

CHAPTER 5

Experiment at the Particle Accelerator

"Le macchine, diceva, sono effetto dell'arte, che è scimmia della natura, e di essa riproducono non le forme ma la stessa operazione."
– Guglielmo da Baskerville, "Il nome della rosa"

Contents

5.1 Introduction 97
5.2 Automatic Test Equipment 98
5.3 Microbeam Chamber 100
5.4 Vehicle Under Test 101
5.5 Signal Integrity Issues 105
5.6 Projectile Selection 109
5.7 Final Test Considerations 111
5.8 Remarks on This Chapter 113

5.1 Introduction

Previous radiation experiments in particle accelerator and laser facilities like [85, 107] were always done using a reduced version of the FT-UNSHADES emulator also known as FTUSB, which is not a fault injection tool, but a portable platform for real testing. The problem with this platform is in the system FPGA used to implement the design under test. In FTUSB, this FPGA is a Xilinx Spartan 3 so

the maximum size available for the design to be tested is reduced when compared to the Xilinx Virtex 2 in FT-UNSHADES. Moreover, the application managing the test in FTUSB is slightly different to that in FT-UNSHADES and so, it is not straightforward to compare signatures from the radiation experiment to that in the fault dictionaries generated in a hardware fault injector like FT-UNSHADES.

It is worth noting that the computer and hardware emulator used for fault injection in the previous chapter, are now arranged to be the ATE in the radiation experiment described in this chapter. This way, the time it takes to prepare the instrumentation for the accelerator test is shortened.

This chapter introduces the custom test equipment deployed in the radiation experiments carried out for this thesis corresponding to the block ATE in the flow chart of Figure 5.1. The ATE (in Figure 5.2) is also referred to as *test fixture* by some authors [92]. Also, the experimental setup necessary for the experiments according to the floor plan in Figure 5.3 is detailed. Later in this chapter, the Device Under Test (JONIC) for the radiation experiment is thoroughly described.

The experiments are designed for signature analysis and so are in accordance to Section 3.6 “Data Collection” in JESD89A where it is suggested to record failure signatures if possible.

5.2 Automatic Test Equipment

The FT-UNSHADES emulator has been slightly modified to be a hardware and software test platform at the accelerator facility using the expansion ports in the board, so a new in-beam mode for the tool, never exploited before, have been developed as mentioned in the previous chapter. This way, the same test procedure used to generate the fault dictionaries is followed during the radiation test. For this purpose, the stimuli applied to both design replicas inside the system FPGA are now sent through the expansion ports to the ASIC, also used to receive the incoming signals from the radiated device. These signals are hashed in the proper block inside the FPGA (the same used for fault injection) and the hash result recorded in a log file.

The same set of test vectors are applied continuously while the device is under radiation, with timeouts between runs. During these short timeouts, it is not possible to block the beam so it is necessary to initialize the circuit previous to the next run. The aim is to have representative numbers and avoid side effects like multiple impacts.

Temperature measurement close to the device could also be measured including extra circuitry to the testbed inside the chamber, however for the present work it was not considered necessary. Target power consumption was measured externally with a precision amperemeter connected to the power line.

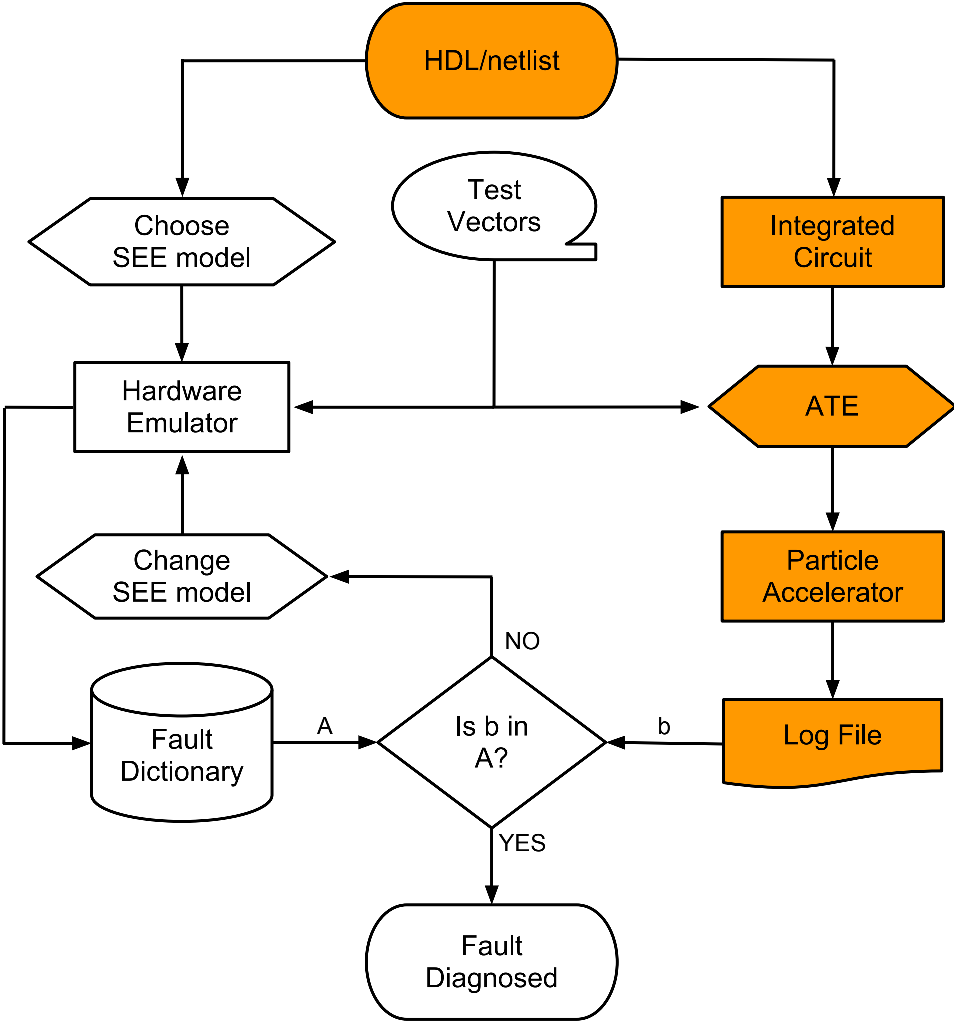


Figure 5.1: Flow chart of the proposed approach. The right branch corresponds to the radiation experiment.

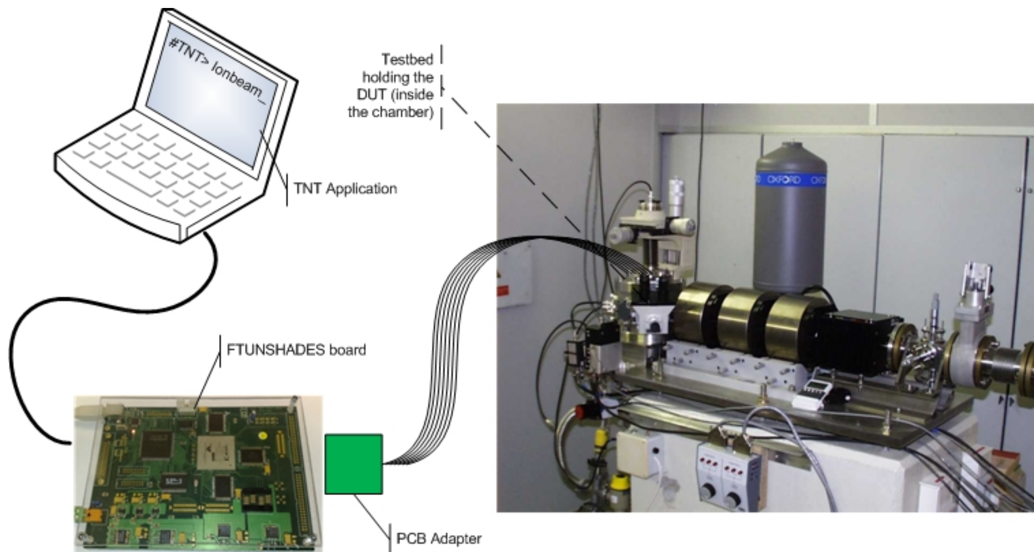


Figure 5.2: Automated Test Equipment (ATE) / Test Fixture.

Outputs from the target are hashed in real time in the system FPGA and recorded to a log file (see Appendix C for an example of log file). Most of the hashes recorded are expected to be the gold one (fault free). Anomalous hashes can be extracted and searched in the fault dictionary for coincidences. If an anomalous hash matches with one on the dictionary, the proper location and clock cycle of the internal fault is then available.

More detailed information about extra circuitry and VHDL considerations for the emulator are included in Appendices A and B.

5.3 Microbeam Chamber

The beam experiment was carried out at the CNA facilities [19, 73] of the University of Sevilla. The accelerator was the 3MV Linear Electrostatic Tandem Van der Graaf, with ions supplied from a SNICS-II ion source. Beam track features a 90º bending magnet and a switching magnet to select between 7 end lines including an external beam line and a nuclear microprobe.

For high beam focusing, we used the nuclear microprobe featuring 3 quadruple magnets close to the target vacuum chamber [74]. High focusing allows to restrict the area of irradiation, improving the control of the experiment and the validation of the results, as discussed later. On the other hand, focusing may help to reduce the size of the fault dictionary by injecting only in the area of interest. Picture in Figure 5.4 is a close view of the microprobe vacuum chamber without the sealing

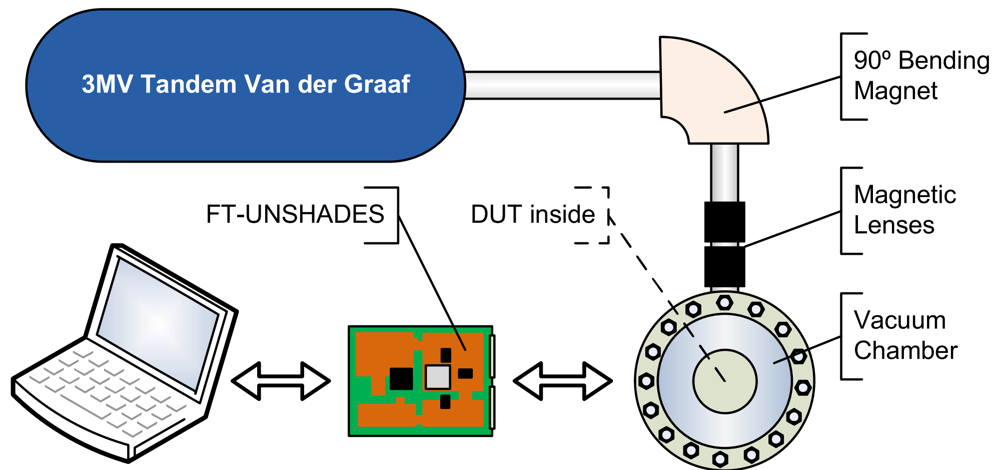


Figure 5.3: Floor plan of the experimental setup at the CNA facilities.

top plate.

A Silicon particle detector was used to measure the particle flux before placing the target in the beam path and a microscope attached to a video cam was used to visually establish the area of interest to be irradiated.

Once the target is placed in front of the beam line, blocking the silicon detector, it is no longer available. The flux is then measured indirectly from the X-ray produced by the bulk Silicon atoms in the target device. For this purpose, a Si(Li) detector is placed 135º away from the beam line. The board holding the target device in Figure 5.5 (see Appendix B) is attached to the top plate of the vacuum chamber so that the target device stay centered when the chamber is covered. A pair of manual positioners allow to move precisely the target in the plane perpendicular to the beam line, aiming the area of interest with the assistance of the microscope and the video cam.

5.4 Vehicle Under Test

JONIC Test Vehicle is a 130nm CMOS digital ASIC in a 40-lead DIL package consisting of six identical instances of a custom design, sharing the input ports and power rails. The design was conceived as a digital dummy target to be operated under radiation with no specific functionality and designed to enhance observability of SEE. The layout of the design is showed in Figure 5.6 and the whole die with the six replicas is in Figure 5.7.

Following is the functional description of the test vehicle corresponding to the block diagram of Figure 5.8. The structure to the left in Figure 5.6 is a 32bit shift register (SR). The individual FFs are observed clearly in an array of 4x8 in the

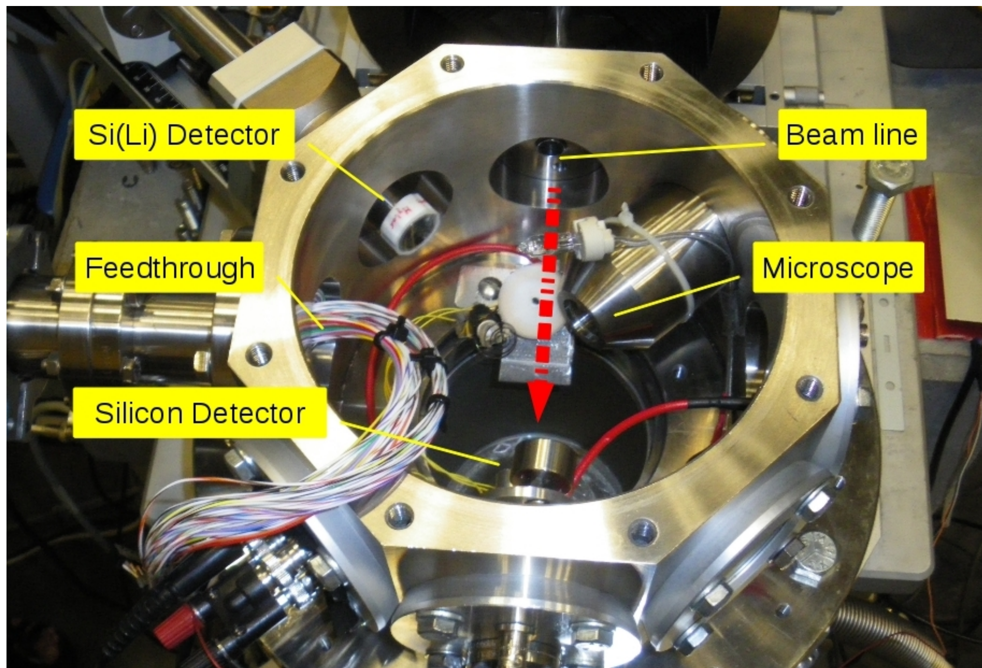


Figure 5.4: Vacuum chamber at the nuclear microprobe line of the 3MV Tandem Van der Graaf Accelerator.

layout with a total size of 30x62 μm. The outputs of the 32 FFs are connected to two identical cones of logic consisting of a matrix of XOR gates. After the cone of logic there are 4x8-bit parallel-to-serial (PS) registers connected as seen in the block diagram. The PS registers are loaded in parallel (dotted bus in Figure 5.8) every 32 clock cycles, and the rest of the time are shifting a parity bit calculated every clock cycle from the content of the SR.

The clock for the 32-bit SR is managed in such a way that it is stopped during a clock cycle after 32 consecutive cycles of serial load, to dump data in parallel to the PS registers through the cone of logic[1]. An input signal enables the PS registers to be loaded in parallel during the cycle when the SR is frozen. The input test vectors are built up from 4 signals: the clock and data input for the SR, and the clock and enable signals for the PS registers.

The blue box in the block diagram of the design is a SET detector to find transient pulses in the parity bit due to impacts on the cone of logic. Basically, it is made by comparing the outputs from two FFs clocked in the rising and falling edge of the clock signal respectively, to detect changes in the parity bit during a single clock cycle. This block is out of the scope of this work, but output coming

[1]For the SET detector to work properly, a specific workload must be used. For arbitrary input stimuli as in the present case study, the putput from the SET detector is meaningless.

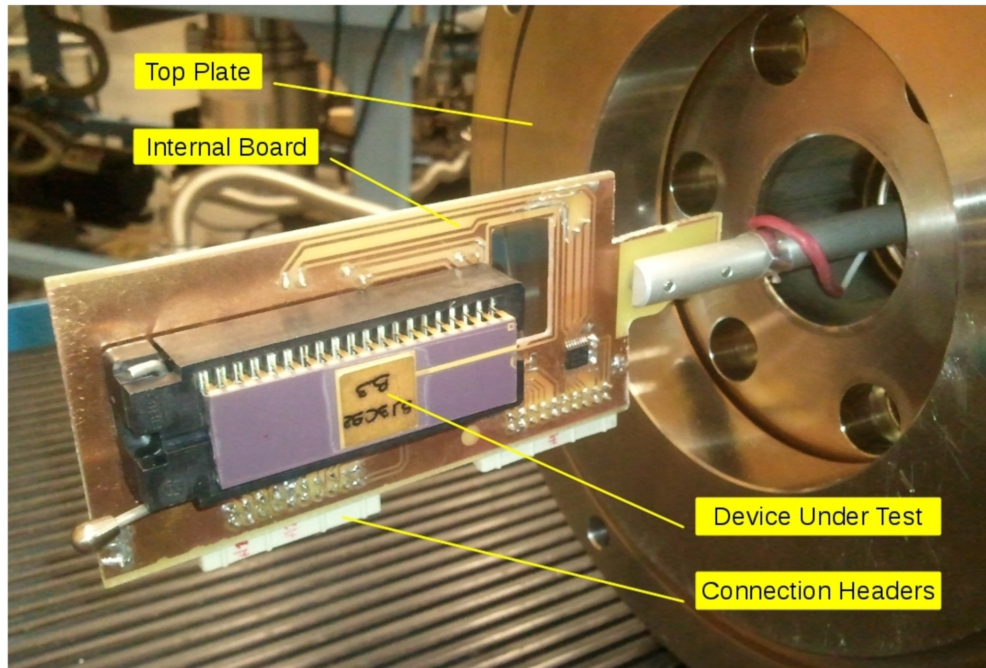


Figure 5.5: Top plate of the vacuum chamber at the nuclear microprobe line and target integrated circuit. The golden lid covering the silicon die is removed before irradiation.

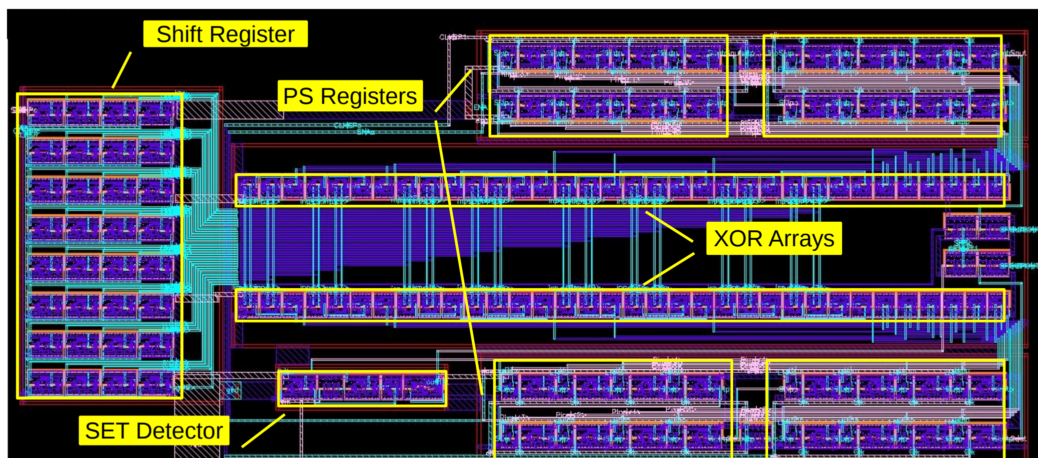


Figure 5.6: Layout of one of the six replicas of the circuit under test.

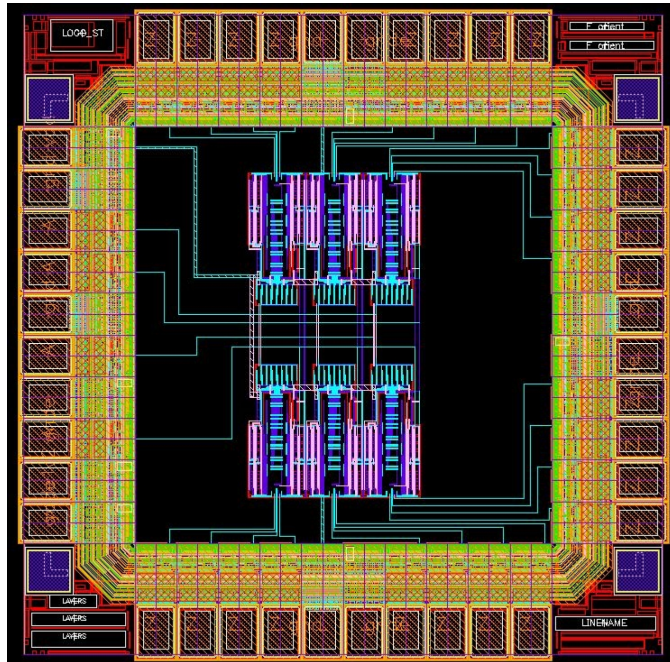


Figure 5.7: JONIC. View of the target die featuring six replicas of the circuit showed in Figure 5.6.

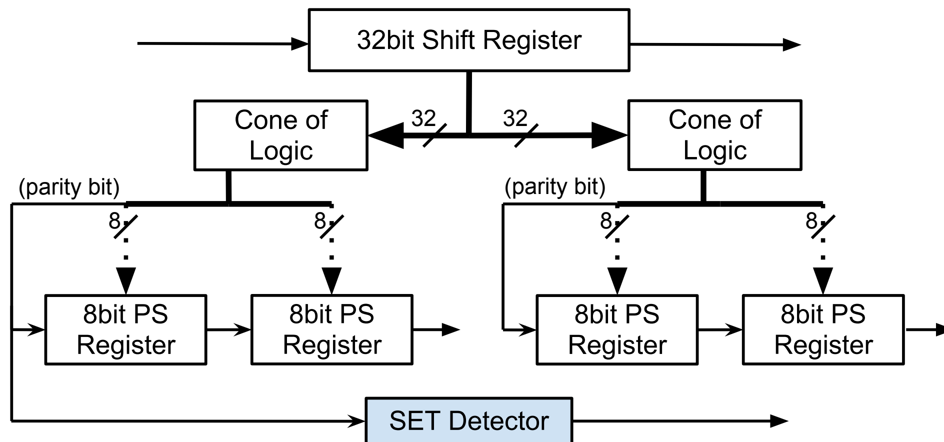


Figure 5.8: Block diagram of the design under test. See text for a detailed description.

from it has been used for the hash calculation due to it is affected when faults are injected in the SR. So, the 4 outputs considered for the hash calculation are coming from the main SR, the PS registers and the SET detector. As long as the FNV hash function needs 8 inputs, the 4 outputs from the circuit were replicated.

Putting the design in a well-known initial state is mandatory for a technique based on signature analysis. In this case, the design has no global asynchronous reset net to avoid unwanted global resets from glitches induced by radiation. Instead, a start up sequence is applied to put the circuit in a known initial state previous to start the hash calculation.

The emulator clock frequency is twice the frequency of the design, so during the fault injection campaign in FT-UNSHADES it is possible to choose to inject in both halves of the design clock cycle. For this case study, a workload with 1127 emulator clock cycles is applied including the start up sequence to reset the whole circuit. A systematic fault injection campaign covering all the FFs and clock cycles in the workload consists of 70886 runs. For a 32-bit hash function there are theoretically a probability of 0.5 to find one collision for 2^{16}(=65536) messages digested[2]. Considering that most of the runs in the fault injection lead to no error at the outputs, the 32-bit version of the FNV hash function is enough to keep low the probability of collision for this case study. A fault dictionary was obtained in few hours from FT-UNSHADES previous to the radiation experiment at the particle accelerator.

Previous experiences and simulation works done for a 0.5 microns CMOS technology resulted in a bit-flip threshold for the LET corresponding to approximately 16 MeV Oxygen or 12 MeV Magnesium ions [87]. For this experiment on a 0.13 microns CMOS technology, Carbon ions were accelerated to 13 MeV to have enough LET to flip the logical state of the FFs. The flux of particles was reduced to 130-160 counts per second (cps) to avoid TID effects during the test.

Focusing is necessary to reduce the beam exposition area, controlling exposition to better understand experimental results. Focusing capabilities of the nuclear microprobe depend on the kinetic energy of the ion. For 13 MeV Carbon ions, magnetic lenses at the end of the beam line are able to focus to the SR area in Figure 5.6, this way limiting faults to this area. All the anomalous hashes detected during the experiment should be related to faults injected in the 32-bit SR.

5.5 Signal Integrity Issues

The configuration of the automated test equipment for the accelerator experiment includes the assessment of signal integrity issues. Remote control of the target

[2]Each message digested is equivalent to a run.

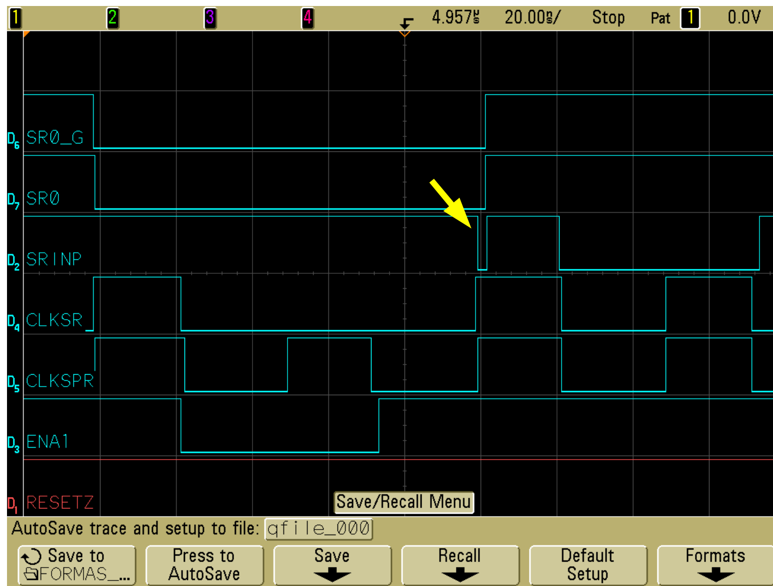


Figure 5.9: Waveforms of input stimuli. A glitch is showed in the input data of the SR.

chip introduces some electrical artifacts mainly due to cabling, the feedthrough in the microprobe and other hardware (see Figure 5.4).

During the assembly of the ATE before moving to the test beam facility, a complete check of performance was done including the bunch of cables, the detachable feedthrough block and the PCBs that hold the DUT and interface the emulator (See the items named “PCB adapter” in Figure 5.2 and “Internal Board” in Figure 5.5). The final layout of these PCBs is included in appendix B. The aim of *in-room* testing was to get working the complete test setup out of the beam.

In absence of radiation, everything during the test bench must run properly. The challenge then was getting golden signatures from the DUT with 100% of confidentiality, guaranteeing no data corruption from the instruments. The golden signature was known from the previous fault injection campaign and also from VHDL simulation. As can be seen, signature analysis is applied here for electronic testing in this stage of the experiment, ensuring that all the data coming out the DUT are valid.

The first issue relating the data channel between the emulator and the device under test was the different voltage ratings. The FPGA banks connected to the expansion ports in the emulator board are referenced to 3.3 V, while the DUT core and input/output buffers are powered to 1.2 V. There are no possibility to change the voltage reference of the output buffers for the FPGA in the emulator, so voltage translation was considered.

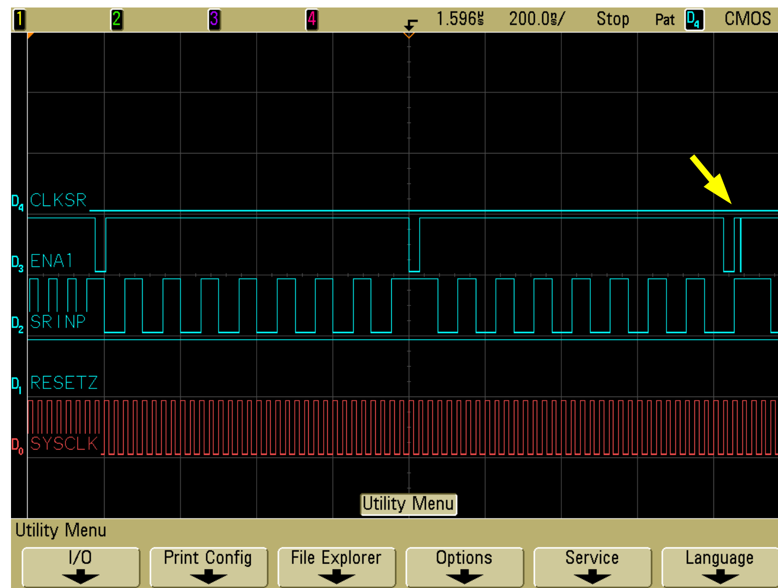


Figure 5.10: Waveforms of input stimuli. A glitch appears in the signal "ENA" that controls the parallel load of the PS registers.

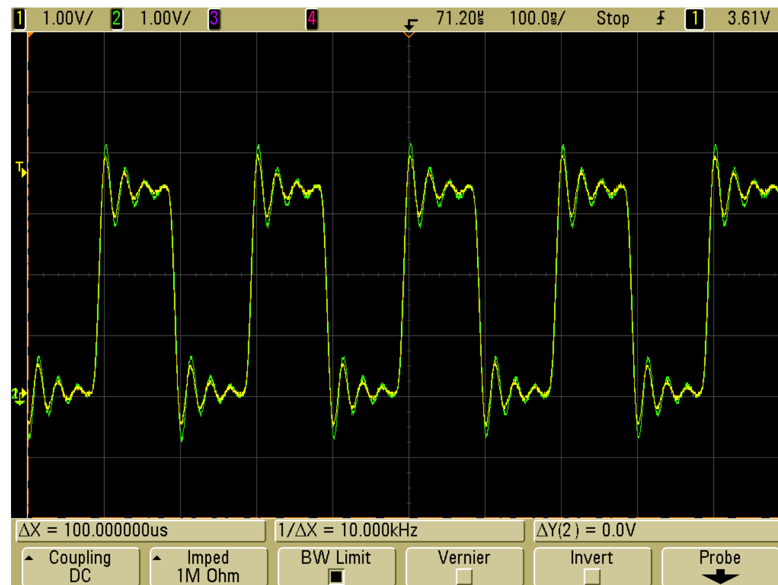


Figure 5.11: Ringing in the clock signal in the DUT side.

5.5.1 Emulator-Side Voltage Translation

A first approach consisted of using voltage conversion at the emulator side. The 8-bit bidirectional voltage-level translator TXB0108 was used for this task, but this device introduces too much delay in the switching, and data was corrupted even for low bit rates (10 MHz). A suitable alternative with better switching characteristics is the SN74AVC4T774.

Voltage conversion at the emulator-side implies that signals referenced to 1.2 V must travel all along the data channel. The problem in a noisy environment is that noise margin is narrow and eventually it becomes a problem for signal integrity. After assembling the complete ATE, the first tests of operation went wrong, with the emulator recording signatures very different from the expected golden signature. The main problem detected was a number of glitches in some signals at the DUT-side. Figures 5.9 and 5.10 show two screenshots from the oscilloscope displaying glitches in two inputs of the DUT. After a number of tests without success, DUT-side voltage translation was considered in order to provide wider noise margin in the data transfer.

5.5.2 DUT-Side Voltage Translation

The major concern here may be related to operation in a radiation environment. However, focusing guarantees no ions out of the beam spot. It was not expected a significant gamma-ray or X-ray background inside the chamber during the test due to the low energy of the incoming ions, so extra circuitry could be placed safely in the inner PCB for the radiation experiment.

With this configuration, signals are referenced to 3.3 V (the voltage reference at the FPGA I/O banks in the emulator) and voltage is translated close to the DUT, inside the microprobe chamber. The noise margin for 3.3 V signaling is wider than 1.2 V and ensures better immunity to noise. However, another problem was observed immediately after the output ports of the DUT, showing no glitches but data appearing several cycles before expected. The analog probe of the oscilloscope showed problems of ringing in the clock signals arriving the DUT that was identified to be the reason for extra clock cycles, inducing anticipation in the circuit response. Ringing can be observed in Figure 5.11 for the 3.3 V clock cycle before voltage translation to 1.2 V.

To mitigate ringing in the clock line, a resistance of 100 Ohm was placed close to the DUT input clock terminating the line in parallel. This modification in the PCB filtered the rapid transitions and overshooting in the clock edges. After this little change, the test started working as expected, recording the gold signature with no flaws. The test was run repetitively for long periods to check the probability of finding errors due to the setup, in absence of radiation. No error arose.

5.6 Projectile Selection

Among the particle accelerators available at the CNA facilities, the most suitable for the proposed experiment is the 3 MV linear tandem Van der Graaf accelerator. This is a low energy accelerator used traditionally for IBA techniques. It means that the maximum energy and ions available are not sufficient to reach the LET values from outer space heavy ions, so the use of this accelerator is restricted to research and pre-validation studies of components.

For radiation experiments involving soft errors, the LET threshold for the technology under study must be achievable from the cocktail of ions and energies available. Device simulation [67] and related experiments [89] demonstrated that the linear accelerator at the CNA facilities was able to flip the logic for a 0.5 μm CMOS technology. Therefore, soft errors on a 130 nm CMOS technology should be feasible in this accelerator due to a significantly lower characteristic critical charge.

The LET threshold and complete SEU cross sections for commercial 130 nm CMOS SRAM memory modules are obtained in [25] experimentally and by simulation. The results on this paper are applicable to our case study due to the similarity in the technology process and scale of integration. The foundry is STMicroelectronics in both cases. These results can be considered as a well basis to make an educated guess on the LET threshold value. From this work, a LET over roughly 2 MeV·cm^2/mg should be enough to flip a standard SRAM cell in the DUT.

To achieve a LET value over the threshold, a set of simulations were carried out in SRIM[114] over an approximated layer model of the DUT for different projectiles. The layer model was extracted from the technical notes and datasheets provided by the foundry (STMicroelectronics 130 nm VLSI HCMOS9 GP) and is built up of[3]:

- Aluminium: 880 nm.
- Silicon Nitride: 1010 nm.
- Silicon Oxide: 5650 nm
- Si-Bulk. Silicon >500nm.

Several ion species and energies were tried following the guidelines in [74]. SRIM simulations for a Carbon ion accelerated to 13 MeV resulted on an associated LET of approximately 4.5 MeV·cm^2/mg in the silicon active region as

[3]The exact layer stack up is confidential and only total thicknesses are provided for each compound.

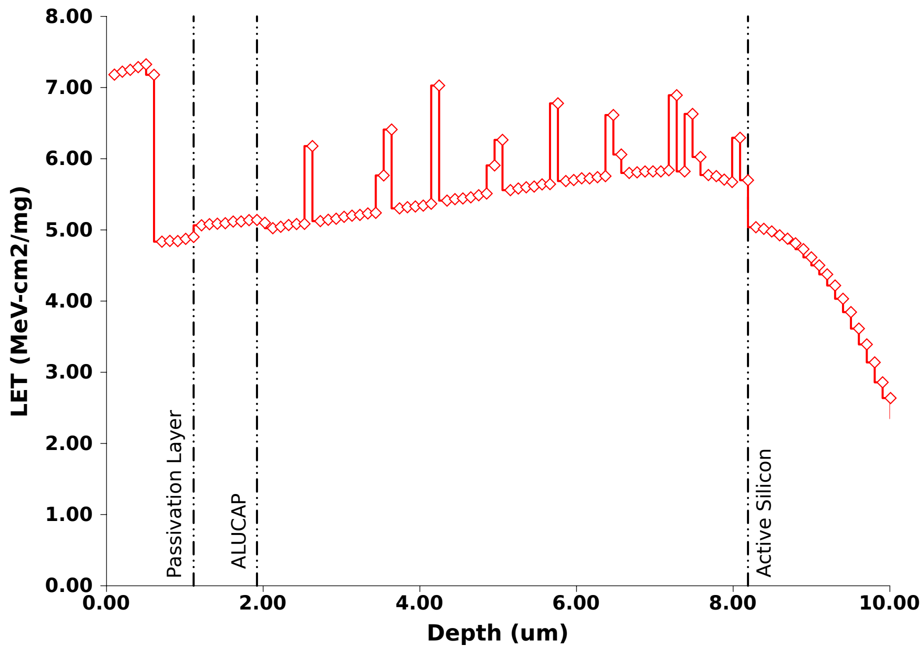


Figure 5.12: Simulated ionizing LET profile for 13 MeV Carbon ions striking a layered model of JONIC device.

showed in Figures 5.12 and 5.13, which is well above the estimated LET threshold of 2 MeV. Figure 5.13 is a detail of the LET profile on the active silicon. The vertical dotted line to the right is approximately the theoretical maximum collection depth (approx. 1 μm) for a 130 nm CMOS technology, from [6]. Charge generated deeper in the bulk silicon does not contribute to the total collected charge.

The total ionizing dose will be negligible and can be even reduced by limiting the total fluence during the beam sessions. The relatively low LET value associated to the Carbon ion also guarantees almost non-existing displacement damage directly related to NIEL, which is roughly 0.1$\%$ of the total energy loss [31].

13 MeV Carbon ions can be conveniently focused by the powerful magnetic lenses in the nuclear microprobe. Moreover, soft errors are likely to occur without physical damage or degradation on the target as predicted by SRIM simulations. For all these considerations, 13 MeV Carbon ions were selected for the radiation experiment.

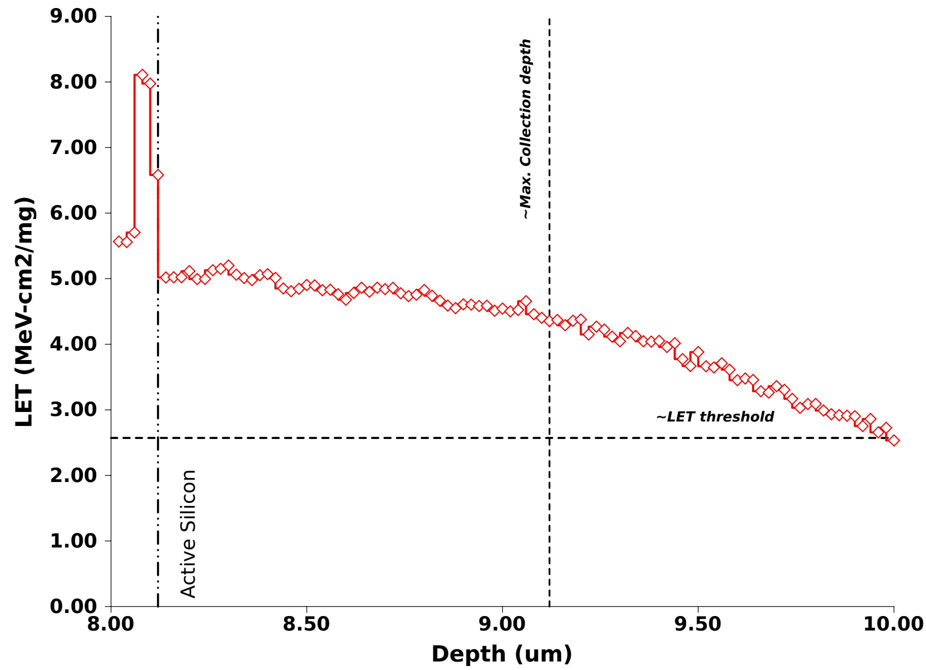


Figure 5.13: Detail of the ionizing LET profile in Fig. 5.12 to show direct ionization in the active region of the silicon device.

5.7 Final Test Considerations

5.7.1 Multi-Bit Upset

Before the experiment, an exhaustive fault dictionary for SEU errors was generated, covering the whole circuit design. Moreover, also a fault dictionary for MBU errors was generated in the area of the Shift Register, where the beam was focused approximately. From the layout of the Shift Register, it is possible to consider fault dictionaries including two or more bit-flips simultaneously in adjacent FFs. Nevertheless, MBU errors are not likely to occur in our case study with 13 MeV Carbon ions at normal incidence.

For radiation experiments including the possibility of introducing tilt angles in the beam, MBU fault dictionaries can provide hashes registered during the test that are not present in the SEU dictionary. Figure 5.14 represents the layout of the targeted shift register with the distribution of the FFs. The colored tiles represent bit-flip patterns related to ions striking with a given tilt angle. The lower the tilt angle, the higher the number of FFs to consider for MBU calculation. For example, the orange pattern would correspond to a small grazing angle of incidence, affecting potentially to 4 FFs in the shift register.

Normal incidence of particles with a high cross-section for nuclear reactions can also lead to MBU due to secondary ionization from the products of the nuclear reaction. Again, it is not the case of 13 MeV Carbon ions.

Hardware emulators are normally able to inject more than only one fault in a given clock cycle. FT-UNSHADES has no limit in the number of faults to be injected simultaneously.

5.7.2 SET Errors

The combinational cone of logic in JONIC consists of 5 stages of XOR logic gates. Referring to works on pulse induced pulse broadening effect as for example [18, 68], only few picoseconds would add to the width of the initial voltage pulse. Broadening is not a concern in short combinational paths for low clock frequencies.

Voltage pulse width after the ion strike is expected to be lower than two hundred of picoseconds, and following [60], the probability of this transient to be sampled at the PS registers (see Figure 5.8) is given by [4]:

$$P_{latch} = \frac{T_{SET}}{T_{Clock}} \tag{5.1}$$

For a pulse width of 200 ps and a clock period of 333 ns (3 Mhz), the probability of latching a SET is lower than 0.1 %. This probability (P_{latch}) must be multiplied by the SET cross-section and integrated for any possible pulse width to find out the probability of having SET errors [60].

In principle, there was no necessity for a SET dictionary since DUT's clock frequency was very low (3 MHz) to have significant probability of having SET errors.

5.7.3 Timing

Time dimension must be also controlled while in the beam to have a reasonable probability of causing SEE before damage from accumulated dose appears. During the experiment, the workload is applied to the target device continuously to detect errors dynamically. Depending on the ATE features, there could be time windows during the beam time when the target is running and timeouts when the target is not active but also receiving radiation. These timeouts are necessary to record data or to prepare the next run. Of course, static SEE from these timeouts must be removed during the reset sequence when a new run is underway. For the present case study, with a single run test time of 169 μs, and a timeout of 64.6

[4] Considering a pulse width larger than the setup+hold times of the FF.

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

Figure 5.14: Floor plan of the 32-bit Shift Register under radiation. Colored tiles represent possible patterns for MBU fault injection in order to get MBU fault dictionaries.

ms, the idle time represent close to 99,7% of the beam time. Supposing a rather homogeneous beam and, for a flux of particles of around 150 cps, there are an interval of approximately 6.6 ms between impacts so it is improbable to have more than one ion striking in the same run. Indeed, it is even rare to have just one strike while the target is running.

5.8 Remarks on This Chapter

This chapter introduces the experimental setup for the radiation test in a particle accelerator. The first challenge was to customize the FT-UNSHADES hardware emulator to turn it into a test fixture. Some extra home-made hardware has

been developed and extra VHDL code to make available the Beam Test Operation Mode.

The 3MV Linear Electrostatic Tandem Van der Graaf accelerator was proved capable for testing soft errors by simulation and experimentally [66, 89]. Previous radiation experiments at the Cyclotron accelerator were also very valuable [107]. Projectile selection was done by SRIM simulation using a complete layer model of the DUT.

The vehicle under test was designed in the Dept. of Electronics Engineering of the University of Seville and have been fabricated in a 130 nm CMOS process. Layout availability is not mandatory, but it helps during the setup of the experiment.

Signal integrity in the communication channel with the DUT is very important to guarantee no errors due to electronics. Transmission line termination should be always considered, even for low frequencies (at least for the clock signal).

Finally in this chapter, a number of test recommendations are done. Particularly, a work done on PIPB effect by the author of this thesis [68], is taken into consideration to evaluate the likeliness of having SET errors during the experiment.

CHAPTER 6

Discussion of the Results

"That's all i have to say about that."
– Forrest Gump

Contents

6.1 Introduction 115
6.2 Experimental Results 115
6.3 Side Effects 118

6.1 Introduction

The results discussed in this chapter, and also the soft error detection and diagnosis methodology, which is the main contribution of this thesis, has been submitted to the IEEE Transactions on Nuclear Sciences. The work is actually under peer review.

6.2 Experimental Results

The experiment consisted of recording dynamically a number of logs with the hash signatures of different runs while the DUT was under radiation. The most of the time, the recorded signatures was the expected one or golden signature, showing no anomalies on the outputs during the whole run. However, during the beam time, some anomalous signatures were recorded indicating the presence of errors at the circuit outputs. Following is an example extracted from a log file recorded during irradiation with a flux of 130 cps:

```
# hash = h2325C245
# hash = h2325C245
# hash = h2325C245
# hash = h99F16C9B
# hash = h2325C245
# hash = h2325C245
# hash = h2325C245
```

The hash value h2325C245 is the expected golden signature for an error-free run, but the hash value h99F16C9B is an anomalous signature and must be found in the fault dictionary to be validated as SEU. In such case, it is possible to classify the error and diagnose the origin.

In fact, signature h99F16C9B is catalogued in the fault dictionary and the error can be diagnosed. In this case, the bit-flip took place on the FF no. 30 in the SR at the clock cycle 733 of the workload, as showed in the following extract of the fault dictionary:

```
# RUN 10623
# Selected clk cycle for SEU insertion:
# 733
# Selected reg for SEU insertion:
# */REG_s30/qs
# Elapsed time: 0.031250
# hash = h99F16C9B
```

The same procedure is applicable to other 24 anomalous signatures obtained dynamically in several different flux conditions to obtain the diagnostic of all the errors that appeared during the radiation experiment. All the signatures were perfectly determined except by one shared by 13 candidate faults. The hash code h921B46F4 is associated in the fault dictionary to 13 different FFs flipped at the clock cycle 1088. For the set of test vectors tried in the beam sessions, Figure 6.1 is representing the number of SEUs diagnosed in the different FFs of the 32-bit SR. Figure 6.2 is representing the distribution of the diagnosed faults along the workload. All the anomalous hashes detected correspond to faults affecting the SR, this way demonstrating that the beam was perfectly focused on this region of the die (see fig. 5.6). No errors were detected originated at the PS registers or at the registers of the SET detector block, as they were out of the beam spot. Figure 6.3 shows the area distribution of SEU error population in the layout of the SR. This graph suggests that the beam was not perfectly centered in the layout area of the SR, but a little bit shifted.

In the event that the anomalous hashes were not present in the SEU-like fault dictionary, then another model for fault injection must be used in order to generate

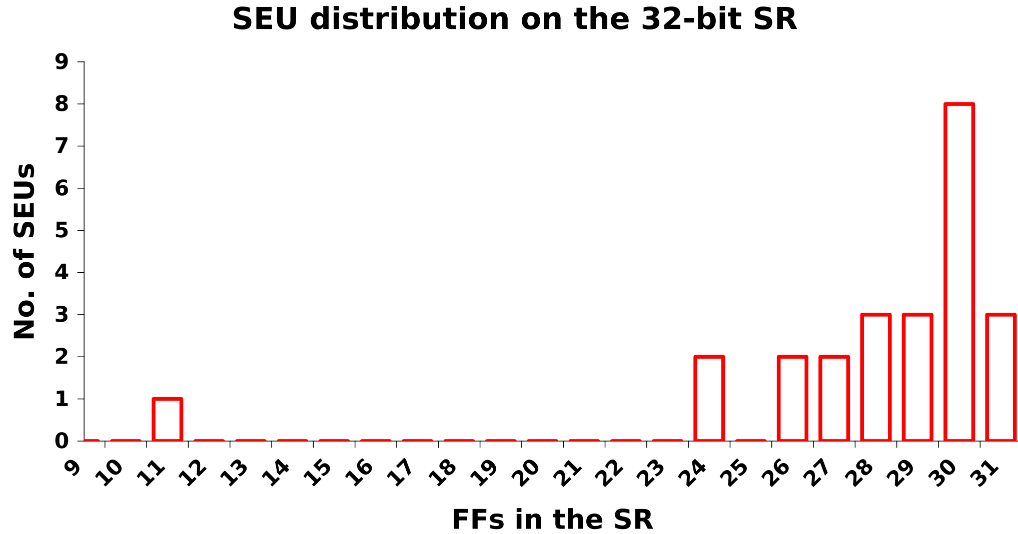


Figure 6.1: Chart representing the FFs affected during irradiation of the 32b Shift Register

a different dictionary. Depending on several experimental variables like angle of incidence, it can be interesting to generate a fault dictionary injecting MBU faults. Also SET errors should be considered for deep submicron technologies clocked at high frequencies.

To get full traceability of each SEU, errors can be reproduced and analyzed cycle by cycle using the latent-damage feature of FT-UNSHADES to observe error propagation across target circuitry.

The proposed approach has been proven to be successful determining the origin of errors observed at the outputs of the target circuit. The case study in this thesis proves the methodology and is applicable to more complex circuits. The unique requirement is the reset condition, since the hash must be calculated from a well-known initial state. Time consumption for the generation of the fault dictionary could be a limitation depending on the size of the design, the number of test vectors and the performance of the fault injector. Optimizing fault injection campaigns by using fast hardware emulators can definitely impact the time required for fault dictionary generation.

Knowing the exact time and location when the fault appeared can be used for designers to better understand the efficiency of a variety of mitigation topologies deployed in a unique device as in [93], tested dynamically. This methodology can also be useful when using radiation testing for the evaluation of selective hardening techniques.

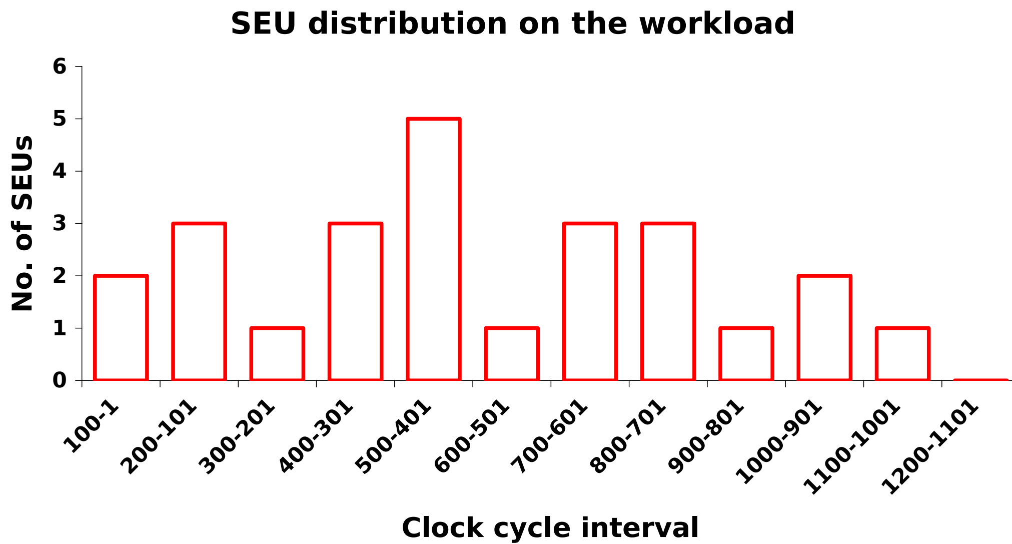


Figure 6.2: Chart representing the interval of clock cycles in the workload affected during irradiation of the 32-bit Shift Register.

Using FPGAs both during the fault injection campaign and as part of the ATE allows to imlement different hash functions adapted to the needs of the design to be tested. Additionally, the hash function module can be tiny in comparison with the CUT, so even powerful hash functions only require few resources from the FPGA.

6.3 Side Effects

This section is dedicated to explain side effects during the experiment. The log files recorded during the radiation experiment showed an *unexpected double golden signature*. It can be seen in the following fragment from a log file:

```
...
# hash = h2325C245
# hash = h2325C245
# hash = h2325C245
# hash = hC3B66675
# hash = hC3B66675
# hash = hC3B66675
# hash = hC3B66675
...
```

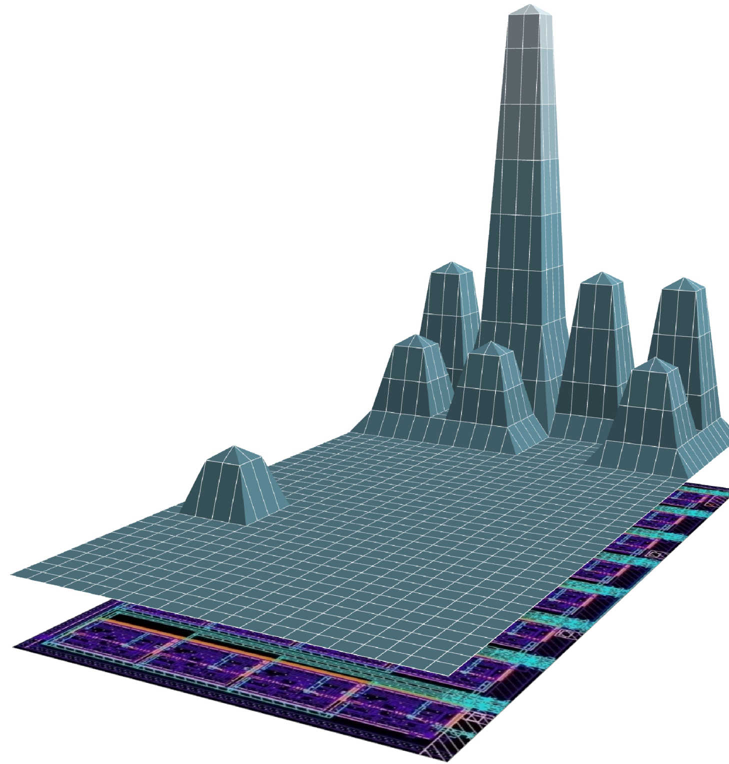


Figure 6.3: 3D graph showing the map of faults in the 32-bit SR. The picture in the base plane is the actual layout of the shift register.

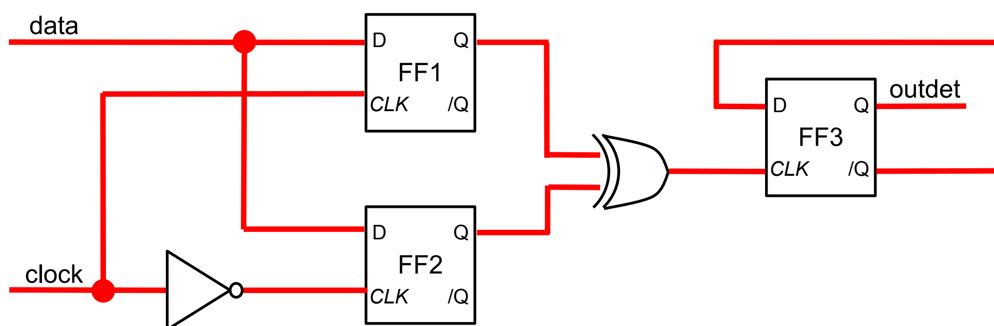


Figure 6.4: Block diagram of the SET detector in JONIC.

The log file recorded two different hashes (h2325C245 and hC3B66675), both related to error-free executions of the test vectors. These two golden signatures appear due to two different initial states of the DUT. After power up, the logic state of each single FF in the design is unknown and so the DUT start a reset sequence to put the system in a well known state. There is no global reset net in the design, but a sequence of input data loaded to the shift register and the PS registers. However, the output FF from the SET detector block, described in Figure 6.4, is not writable and can not be initialized. This FF provide the *outdet* output that is used for the hash function to generate hash values.

The SET detector block is used for hash calculation since it introduces variability in the output data flow. However, SET detection was out of the scope of the experiment. As stated by Eq. 5.1, for SET detection higher frequencies are required in order to have a reasonable probability of having this kind of soft errors.

The idea behind the design of the SET detector block is sampling an input signal from the cone of logic at double data rate (triggering FF1 and FF2 at rising and falling clock edges respectively). This way, it is possible to monitor data changing whithin a clock cycle. In the event of a transient voltage excursion sampled by the falling edge of the clock, the XOR gate after FF1 and FF2 changes from '0' to '1'. This change triggers the FF3 output register. FF3 is wired in a feedback configuration from the '/Q' output port.

Depending on the initial state of FF3 after power on, the resulting golden hash codes would be #h2325C245 or #hC3B66675 as can be easily found by simulation. During the beam experiment, no powering cycles were applied so it wasn't expected to find different gold signatures in a given log file. However, eventually during the beam test, the golden signature swapped randomly as can be seen in the extract of log file showed previously.

The explanation for this effect is in the timeouts between runs. As stated in previous chapters, ions remain hitting the DUT between runs presumably inducing static soft errors. Of course, this soft errors are removed from the design by the reset sequence before starting a new run. However, as discussed previously, there is no way to force a initial value in FF3. In the event of an ion striking FF1, FF2 or FF3 during the timeouts, it would be likely to have a bit-flip in one of these FFs. It can be seen that a bit-flip in any of these FFs leads to a change in the state of FF3 and so in the initial state of the circuit for the next run.

The record of different golden signatures in the log file demonstrates that, even though the beam was approximately focused to the 32-bit Shift Register area, some ions reach the area of the SET detector. Figure 5.6 shows the proximity between the SR and the SET detector.

This effect was not predicted before the radiation experiment, but it is easily understood after signature analysis.

CHAPTER 7

Conclusions and Future Work

“What is not started today is never finished tomorrow."
– Johann Wolfgang von Goethe

Contents

7.1 Conclusions of the Present Work 121
7.2 Future Work 122

7.1 Conclusions of the Present Work

The primary objective of this thesis was to make hardware fault injection techniques a valid method to assist in the standard radiation test work flow. Fault injection techniques are used commonly to find vulnerabilities in the early design stages of electronic circuits.

In this thesis, the capacity of hardware fault injectors to perform massive fault injection campaigns in a relative short time, has been exploited to develop a methodology based on signature analysis. Particularly, the characterization and diagnosis of soft errors on digital electronic circuits due to radiation induced upsets. For signature analysis, fault injection is required in order to achieve a complete dictionary of errors. For this task, a mature hardware emulator (FT-UNSHADES) was used successfully with minor modifications to include a hash algorithm.

Using the techniques developed and tested in this thesis, the following objectives have been reached:

- An original technique for soft error characterization. The problem with failure detection in radiation test experiments is the validation and classification

of the observed error. The complexity inherent to particle accelerator facilities and radiation setups, introduces a uncertainty in the observed effects of radiation. The introduction of signature analysis provides a method to guarantee the nature of the observed error. Different fault models lead to different fault dictionaries. Theoretically, SEU, SET or MBU faults can be recognized after checking the registered signature if a complete dictionary was generated by fault injection.

- An original technique for fast diagnosis of observed errors. Radiation hardening by design and fault mitigation techniques can benefit from exact diagnosis of observed errors during radiation tests. Dynamic testing of digital components is the best way to test the robustness of an irradiated circuit under operation. Dynamic testing makes it hard to diagnose observed errors since outputs can be wrong for several clock cycles. Signature generation provides a way to code failure behaviour in a single signature, simplifying the diagnosis of the failure in terms of affected FF and instant of occurrence.

- A statistical procedure for the qualification of test vectors for radiation tests. The selection of a set of test vectors for a radiation test is not direct. An inadequate set of test vectors could mask errors that, in other case, would reveal weakness in the design. Hence, the stimuli for a radiation test have to provide visibility to the experiment. Attending to visibility, the statistical treatment of the fault dictionaries gives the percentage of observed errors. Furthermore, attending to signature aliasing, it is possible to extract the probability of having exact diagnosis of the observed errors.

- A simple method to improve existing test fixtures. Applying signature analysis to radiation experiments requires the introduction of a signature module in the test fixture. Platforms based on FPGA devices are particularly flexible to incorporate new hardware blocks. The HDL description of a hash function can be introduced easily to process data coming from the DUT outputs.

The effectiveness of these contributions have been demonstrated in a real radiation experiment in this thesis.

7.2 Future Work

As said in section 1.7, the optimization of the methodology introduced in this thesis is out of the scope of this thesis. Therefore, there is a number of improvements and work to do regarding several aspects of the methodology.

- A first step forward in the work developed consists of testing larger designs in a radiation test. The drawback with large designs is in the time to get a complete fault dictionary, since the number of required fault injections can be huge. This can be overcome with faster emulators or by focusing on the radiation experiment. Regarding beam focusing, it is possible to plan a microbeam experiment to focus the ion beam on a portion of the whole circuit layout. The fault dictionary can be then constrained to the FFs in the area to be irradiated.

- The use of the methodology developed in this thesis is, in principle, constrained to digital designs with a hardware description or netlist available. However, for Commercial Off-The-Shelf components it is possible to model a functionally identical netlist from a similar known architecture. A future work on this direction will show the main drawbacks in the technique when applied to COTS. It is worth noting that, for accurate error diagnosis, it is necessary to know the netlist of the DUT. In other case, the radiation engineer has no way to assess the origin of errors.

- There is a natural extension of this work involving radiation tests of SRAM FPGAs. When targeting these devices in a radiation test, the most susceptible elements are the configuration bits. The rate of faults in the user design FFs is negligible and soft errors occur almost uniquely in the configuration memory. For this reason, it is common in SRAM FPGA devices a process known as *scrubbing*. It consists of a periodic partial or total reconfiguration of the FPGA to overwrite corrupted bits. Before performing scrubbing, it is possible to readback the configuration bits. Data can be then processed to have a signature corresponding to the configuration memory. This way, signature analysis can be applied to the whole configuration memory or only to few frames. With this information, it would be possible to selectively apply correction to the configuration memory.

- The forthcoming platform FTU2 will be a perfect test bench for the methodology developed in this thesis. The new hardware emulator features an autonomous mode to speed-up massive fault injection campaigns. In this platform, a complete process of optimization will be done to test more complex designs.

- There are still a number of tests of feasibility before signature analysis applied to dynamic radiation experiments on integrated circuits becomes industrialized. Outputs from the DUT are generally accesible to the observer. Therefore, as a first upgrade, actual text fixtures/ATEs can be equipped with a signature module to hash data coming from the DUT. All runs in the beam

line can be then recorded for future analysis. A second step would consist of acquiring a hardware emulator for the generation of fault dictionaries. Hardware emulators are based mostly in FPGAs and are easily customizable, allowing the introduction of new modules in the standard work flow.

Appendices

APPENDIX **A**

VHDL Considerations for Turning FT-UNSHADES into an ATE

This appendix is a brief guide outlining the necessary changes in the FT-UNSHADES emulator firmware to make it available the Beam Test Mode of Operation described in section 4.2. Changes are related to files *ftunshades.vhd*, *ftunshades.ucf* and *debug_controller.vhd*. The file *top_hash_FNV.vhd*[1] in Appendix D must be included to the Xilinx ISE project "Design for Test Environment (DTE)".

- *ftunshades.vhd*. This module have to instantiate as much FNV blocks as necessary, depending on the number of output ports in the MUT instance. Some output and input ports must be added to the top entity to send stimuli and capture the outputs from the irradiated device. It is necessary to provide an input port to switch between beam test or fault injection modes of operation. Depending on this mode, the FNV block is connected to either top entity input ports (radiation test) or SEU-MUT instance output ports (fault dictionary generation).

- *ftunshades.ucf*. The output ports added to the top instance in *ftunshades.vhd* must be connected to the corresponding FPGA pin. These pins must be connected to the expansion socket in the board.

- *debug_controller.vhd*. Normal operation of the fault injector implies stopping the run when an error is detected at the outputs. The only change to do in this module allows the emulator to run freely until the end of the test vectors, whenever an error is detected or not. This way the complete data flow from the outputs is hashed. For this purpose, the signal *monitor_msk* must be initialized to 0.

[1]This entity is the corresponding to the FNV hash function. Other hash functions will require the corresponding VHDL entity.

APPENDIX B

PCB Designs for the Microprobe End-Line

This appendix describes the functionality of the extra hardware required for the experiment at the nuclear microprobe.

A testbed PCB is required as sample holder to attach the DUT. This board features a 40-lead Zero Insertion Force (ZIF) socket for JONIC. Voltage translation is achieved by means of 4 SN74AVC4T774 voltage shifter in a SOIC package. Two connectors with 20-pins are placed in one side for connecting communication and power cables. A small window (white rectangle in figures B.1 and B.2) was cut in the middle of the board to measure beam fluence. For this purpose, a silicon detector was placed in front of the beam, in the vacuum chamber. After this measure, the sample holder is shifted upwards to put the silicon die in the way of the beam.

A PCB adapter is required to be attached to the emulator board. The purpose of this board is to provide a pair of sockets for a bunch of flying cables to the feedthrough in the vacuum chamber. This board also features a voltage regulator to get a 1.2 V power source from the 3.3 V supplied by the emulator board. The 1.2 V power source supplies current to the DUT and SN74AVC4T774 integrated circuits inside the vacuum chamber.

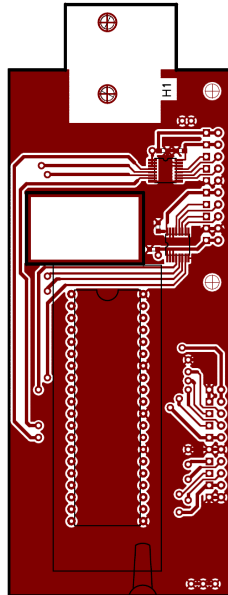


Figure B.1: Top layer of the testbed holding the DUT inside the microprobe's vacuum chamber. Not to scale.

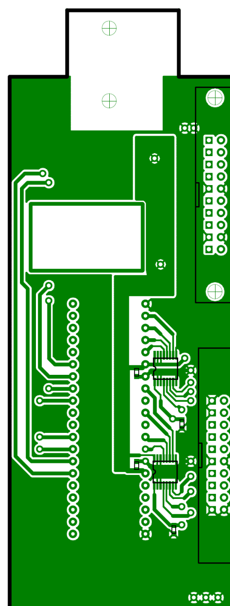


Figure B.2: Bottom layer of the testbed holding the DUT inside the microprobe's vacuum chamber. Not to scale.

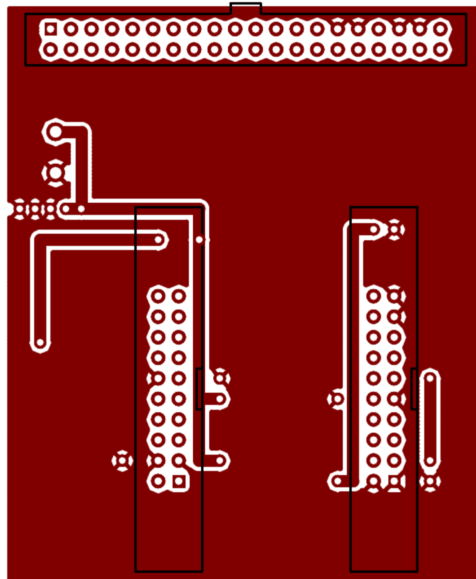


Figure B.3: Top layer of the pcb adapter for FT-UNSHADES connection to the microprobe's hermetically-sealed feedthrough. Not to scale.

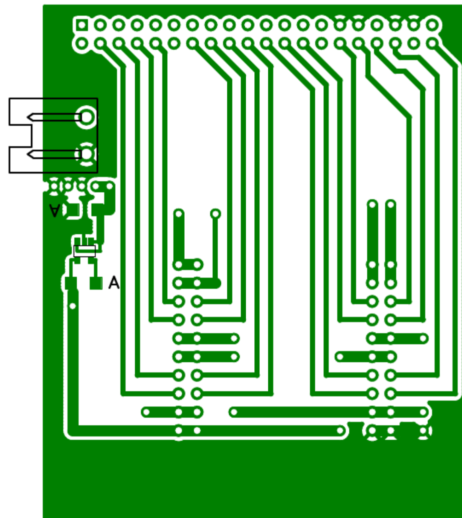


Figure B.4: Bottom layer of the pcb adapter for FT-UNSHADES connection to the microprobe's hermetically-sealed feedthrough. Not to scale.

APPENDIX C

Log Files from Radiation Experiments

This appendix is a fragment from a log file generated during radiation experiments at the CNA. Here, the faulty hashes different to the expected one are h921B46F4, h0A39E3EA, h9C236547 and hB357EF41.

```
# FTUnshades Test aNalysis Tools
# Release version: 2.3.2
# Build Date: Sep 19 2009
# AICIA-GIE
# Date: 10/26/2011 18:11:46
#
# Comm port successfully opened:
#    Number of devices detected: 1
#     #    ID
#    ----------
#     0    FTU-n12/3000
set directory="C:\Documents and Settings\RadUS-1\Desktop\b-Microprobe_130nm\dte"
set vectfile=ftunshades_memory_coe2_FULL.dat
loadbit
# Design name: ftunshades_top.ncd
# Design date: 2011/09/27, 13:23:08
# Bitfile device type: 2v3000ff1152
# File sent: 1313222 Bytes
loadll
loadvect
# Loading test vectors
# Number of test vectors: 1127
define SR=[SEU_MUT/*REG_s* SEU_MUT_O_sr0c_i]
define hash=[hashfunct/hash<*>]
define mux=[MUX_SEL]
define sp1=[SEU_MUT*regsp1* seu_mut*spout1c*]
define sp2=[SEU_MUT*regsp2* seu_mut*spout2c*]
define sp1_g=[GOLD_MUT*regsp1* GOLD_MUT*spout1c*]
define sp2_g=[GOLD_MUT*regsp2* GOLD_MUT*spout2c*]
set maxruns=100
set quiet=1
function ionbeam
# Starting function definition (ENDF to terminates)
```

```
pulserst
pulsersm
readb hash
endf
keyboard
ionbeam
# hash = h2325C245
ionbeam
# hash = h2325C245
repeat 700 ionbeam
# hash = h2325C245
# hash = h2325C245
# hash = h2325C245
# hash = h2325C245
# hash = h2325C245
# hash = h2325C245
# hash = h2325C245
..................
# hash = h2325C245
# hash = h921B46F4  ***Failure detected!***
# hash = h2325C245
..................
# hash = h2325C245
# hash = h0A39E3EA  ***Failure detected!***
# hash = hC3B66675
..................
# hash = h2325C245
# hash = h9C236547  ***Failure detected!***
# hash = h2325C245
..................
# hash = h2325C245
# hash = h2325C245
# hash = h2325C245
# hash = h2325C245
# hash = hC3B66675
# hash = hC3B66675
# hash = hC3B66675
# hash = hC3B66675
# hash = hC3B66675
# hash = hC3B66675
# hash = hC3B66675
# hash = hC3B66675
# hash = hB357EF41  ***Failure detected!***
# hash = hC3B66675
# hash = hC3B66675
# hash = hC3B66675
# hash = hC3B66675
..................
# hash = hC3B66675
# hash = hC3B66675
# hash = hC3B66675
quit
# Comm port succesfully closed
```

APPENDIX D

VHDL Entity for FNV

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

--********FNV-1 HASH ALGORITHM***********
--    hash = offset_basis
--    for each octet_of_data to be hashed
--     hash = hash * FNV_prime
--     hash = hash xor octet_of_data
--    return hash
--**************************************

entity top_hash_FNV is
    Port ( clk : in  STD_LOGIC;
           reset_n : in  STD_LOGIC;
           message_in : in  STD_LOGIC_VECTOR (7 downto 0);
           hash : out  STD_LOGIC_VECTOR (31 downto 0));
end top_hash_FNV;

architecture Behavioral of top_hash_FNV is
signal offset_basis, FNV_prime,i_hash_0, i_hash_1, n_hash_3 : STD_LOGIC_VECTOR (31 downto 0);
--signal i_hash_1 : STD_LOGIC_VECTOR (63 downto 0);
signal i_hash_2 : STD_LOGIC_VECTOR (63 downto 0);

begin
offset_basis <= x"811c9dc5";
FNV_prime <= x"01000193";
--offset_basis <= x"14650FB0739D0383";
--FNV_prime <= x"00000100000001B3";

--FNV1a:
i_hash_1(31 downto 8) <= i_hash_0(31 downto 8);
i_hash_1(7 downto 0) <= i_hash_0(7 downto 0) XOR message_in;
i_hash_2 <= i_hash_1*FNV_prime;
n_hash_3 <= i_hash_2(31 downto 0);

--FNV1:
--i_hash_1 <= i_hash_0*FNV_prime;
```

```
--i_hash_2 <= i_hash_1(31 downto 0);
--n_hash_3(31 downto 8) <= i_hash_2(31 downto 8);
--n_hash_3(7 downto 0) <= i_hash_2(7 downto 0) XOR message_in;

sync: process (clk,reset_n)
begin
if reset_n = '0' then
hash <= (others=>'0');
i_hash_0 <= offset_basis;
elsif (clk'event and clk = '1') then
i_hash_0 <= n_hash_3;
hash <=  n_hash_3;
end if;
end process;

end Behavioral;
```

APPENDIX E

VHDL Package for RIPEMD-160

```
--Package File for RIPEMD_160 Core
--
-- Purpose: This package defines supplemental types, subtypes,
-- constants, and functions
--
-- Notes: All values for constants and definitions are taken from the paper:
--"RIPEMD-160, a strengthened version of RIPEMD",
--H. Dobbertin, A. Bosselaers, B. Preneel.


library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.NUMERIC_STD.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
-- *********************START PACKAGE DECLARATION**********************
package RIPEMD_160 is

-- TYPES
type block_in is array (15 downto 0) of std_logic_vector (31 downto 0);
type md_words is array (4 downto 0) of std_logic_vector (31 downto 0);
type array_of_integers is array (15 downto 0) of integer range 0 to 15;

-- CONSTANTS
-- LEFT
constant K_LEFT_1 : std_logic_vector(31 downto 0) := (others => '0');
constant K_LEFT_2 : std_logic_vector(31 downto 0) := x"5A827999";
constant K_LEFT_3 : std_logic_vector(31 downto 0) := x"6ED9EBA1";
constant K_LEFT_4 : std_logic_vector(31 downto 0) := x"8F1BBCDC";
constant K_LEFT_5 : std_logic_vector(31 downto 0) := x"A953FD4E";

-- RIGHT
constant K_RIGHT_1 : std_logic_vector(31 downto 0) := x"50A28BE6";
constant K_RIGHT_2 : std_logic_vector(31 downto 0) := x"5C4DD124";
constant K_RIGHT_3 : std_logic_vector(31 downto 0) := x"6D703EF3";
constant K_RIGHT_4 : std_logic_vector(31 downto 0) := x"7A6D76E9";
constant K_RIGHT_5 : std_logic_vector(31 downto 0) := (others => '0');

-- INITIAL VALUES
constant H0 : std_logic_vector(31 downto 0) := x"67452301";
```

```
constant H1 : std_logic_vector(31 downto 0) := x"EFCDAB89";
constant H2 : std_logic_vector(31 downto 0) := x"98BADCFE";
constant H3 : std_logic_vector(31 downto 0) := x"10325476";
constant H4 : std_logic_vector(31 downto 0) := x"C3D2E1F0";

-- ROL STEPS
constant rol_round1_a :
array_of_integers := (11, 14, 15, 12, 5, 8, 7, 9, 11, 13, 14, 15, 6, 7, 9, 8);
constant rol_round2_a :
array_of_integers := ( 7,  6,  8, 13,11, 9, 7,15,  7, 12, 15,  9,11, 7,13,12);
constant rol_round3_a :
array_of_integers := (11, 13,  6,  7,14, 9,13,15, 14,  8, 13,  6, 5,12, 7, 5);
    constant rol_round4_a :
array_of_integers := (11, 12, 14, 15,14,15, 9, 8,  9, 14,  5,  6, 8, 6, 5,12);
constant rol_round5_a :
array_of_integers := ( 9, 15,  5, 11, 6, 8,13,12,  5, 12, 13, 14,11, 8, 5, 6);

constant rol_round1_b :
array_of_integers := ( 8,  9,  9, 11,13,15,15, 5,  7,  7,  8, 11,14,14,12, 6);
    constant rol_round2_b :
array_of_integers := ( 9, 13, 15,  7,12, 8, 9,11,  7,  7, 12,  7, 6,15,13,11);
constant rol_round3_b :
array_of_integers := ( 9,  7, 15, 11, 8, 6, 6,14, 12, 13,  5, 14,13,13, 7, 5);
constant rol_round4_b   :
array_of_integers := (15,  5,  8, 11,14,14, 6,14,  6,  9, 12,  9,12, 5,15, 8);
constant rol_round5_b   :
array_of_integers := ( 8,  5, 12,  9,12, 5,14, 6,  8, 13,  6,  5,15,13,11,11);

-- PERMUTATION FUNCTIONS
    function PERMUTATION_RHO (constant i : in integer)
return integer;

function PERMUTATION_PI (constant i : in integer)
return integer;

-- BOOLEAN FUNCTIONS

function BOOLFUNCT_1   ( signal X : in std_logic_vector(31 downto 0);
signal Y : in std_logic_vector(31 downto 0);
signal Z : in std_logic_vector(31 downto 0))
return std_logic_vector;

function BOOLFUNCT_2   ( signal X : in std_logic_vector(31 downto 0);
signal Y : in std_logic_vector(31 downto 0);
signal Z : in std_logic_vector(31 downto 0))
return std_logic_vector;

function BOOLFUNCT_3   ( signal X : in std_logic_vector(31 downto 0);
signal Y : in std_logic_vector(31 downto 0);
signal Z : in std_logic_vector(31 downto 0))
return std_logic_vector;

function BOOLFUNCT_4   ( signal X : in std_logic_vector(31 downto 0);
signal Y : in std_logic_vector(31 downto 0);
signal Z : in std_logic_vector(31 downto 0))
return std_logic_vector;

function BOOLFUNCT_5   ( signal X : in std_logic_vector(31 downto 0);
signal Y : in std_logic_vector(31 downto 0);
signal Z : in std_logic_vector(31 downto 0))
return std_logic_vector;
```

```
-- OTHER FUNCTIONS
function PERMUT_BLOCK ( signal input : in block_in;
signal permut_sel : in std_logic)
return block_in;

end RIPEMD_160;
-- *********************END PACKAGE DECLARATION**********************

-- *********************START PACKAGE BODY***************************
package body RIPEMD_160 is

-- PERMUTATIONS
function PERMUTATION_RHO (constant i : in integer)
return integer is
begin
case i is
when 0  => return 7;
when 1  => return 4;
when 2  => return 13;
when 3  => return 1;
when 4  => return 10;
when 5  => return 6;
when 6  => return 15;
when 7  => return 3;
when 8  => return 12;
when 9  => return 0;
when 10 => return 9;
when 11 => return 5;
when 12 => return 2;
when 13 => return 14;
when 14 => return 11;
when 15 => return 8;
when others => return 16;
end case;
end PERMUTATION_RHO;

   function PERMUTATION_PI (constant i : in integer)
return integer is
begin

return (9*i + 5) mod 16;

end PERMUTATION_PI;

-- FUNCTIONS
  function BOOLFUNCT_1  ( signal X : in std_logic_vector(31 downto 0);
signal Y : in std_logic_vector(31 downto 0);
signal Z : in std_logic_vector(31 downto 0))
return std_logic_vector is
begin

return X XOR Y XOR Z;

end BOOLFUNCT_1;

function BOOLFUNCT_2  ( signal X : in std_logic_vector(31 downto 0);
signal Y : in std_logic_vector(31 downto 0);
signal Z : in std_logic_vector(31 downto 0))
return std_logic_vector is
begin

return (X AND Y) OR (NOT X AND Z);
```

```
end BOOLFUNCT_2;

function BOOLFUNCT_3  ( signal X : in std_logic_vector(31 downto 0);
signal Y : in std_logic_vector(31 downto 0);
signal Z : in std_logic_vector(31 downto 0))
return std_logic_vector is
begin

return (X OR NOT Y) XOR Z;

end BOOLFUNCT_3;

function BOOLFUNCT_4  ( signal X : in std_logic_vector(31 downto 0);
signal Y : in std_logic_vector(31 downto 0);
signal Z : in std_logic_vector(31 downto 0))
return std_logic_vector is
begin

return (X AND Z) OR (Y AND NOT Z);

end BOOLFUNCT_4;

function BOOLFUNCT_5  ( signal X : in std_logic_vector(31 downto 0);
signal Y : in std_logic_vector(31 downto 0);
signal Z : in std_logic_vector(31 downto 0))
return std_logic_vector is
begin

return X XOR (Y OR NOT Z);

end BOOLFUNCT_5;

function PERMUT_BLOCK ( signal input : in block_in;
signal permut_sel : in std_logic)
return block_in is

variable aux : block_in;
begin

if permut_sel = '0' then
for i in 0 to 15 loop
aux(i) := input(PERMUTATION_RHO(i));
end loop;
return aux;
else
for i in 0 to 15 loop
aux (i) := input(PERMUTATION_PI(i));
end loop;
return aux;
end if;

end PERMUT_BLOCK;

end RIPEMD_160;
-- *********************END PACKAGE BODY******************************


-- *********************START OF ENTITIES*****************************
----------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
```

```
use IEEE.NUMERIC_STD.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;
library TESIS;
use TESIS.ripemd_160.all;

entity core_block is
generic (
k : std_logic_vector(31 downto 0) := K_LEFT_1;
rol_round: array_of_integers := rol_round1_a
);
port(
clk : in std_logic;
reset_n : in std_logic;

--input block with 16 32-bit words and input message digest with 5 32-bit words.
--message digest: a(t-1) = md_in(0)
--b(t-1) = md_in(1)
--c(t-1) = md_in(2)
--d(t-1) = md_in(3)
--e(t-1) = md_in(4)

input  : in block_in;
md_in  : in md_words;
count16 : in integer range 0 to 15;
perm_sel : in std_logic;
-- output block with PERMUTED 16 32-bit words and output message digest with 5 32-bit words.
--message digest: a(t) = md_out(0)
--b(t) = md_out(1)
--c(t) = md_out(2)
--d(t) = md_out(3)
--e(t) = md_out(4)

output : out block_in;
md_out : out md_words

);
end core_block;

architecture behavioral of core_block is

signal p_md_out : md_words;
signal aux1, aux2, aux3 : std_logic_vector(31 downto 0); -- aux signals for b_t calculation
signal aux4 : bit_vector(31 downto 0);

begin

--a_t <= e_t-1
p_md_out(0) <= md_in(4);

--b_t <= e_t-1 + (ft(bt-1,ct-1, dt-1) +a_t-1 + Xi + Kj) rol s
aux1 <= BOOLFUNCT_1 (md_in(1), md_in(2), md_in(3));
aux2 <= (input(count16) + k) + md_in(0);
aux3 <= aux1 + aux2;
aux4 <= to_bitvector(aux3) rol rol_round(count16);
p_md_out(1) <= md_in(4) + to_stdlogicvector(aux4);
```

```
--c_t <= b_t-1
p_md_out(2) <= md_in(1);

--d_t <= c_t-1 rol 10
p_md_out(3) <= to_stdlogicvector(to_bitvector(md_in(2)) rol 10);

--e_t <= d_t-1
p_md_out(4) <= md_in(3);

async : process (input, perm_sel)
begin
if perm_sel='0' then
for i in 0 to 15 loop
output(i) <= input(PERMUTATION_PI(i));
end loop;
else
for i in 0 to 15 loop
output(i) <= input(PERMUTATION_RHO(i));
end loop;
end if;
end process;

sync : process (reset_n, clk)
begin
if (reset_n='0') then
for i in 0 to 4 loop
md_out(i) <= (others=>'0');
end loop;
elsif (clk'event and clk='1') then
for i in 0 to 4 loop
md_out(i) <= p_md_out(i);
end loop;
end if;
end process;
end behavioral;
```

Bibliography

[1] O. Adriani, G. C. Barbarino, G. A. Bazilevskaya, R. Bellotti, M. Boezio, E. A. Bogomolov, M. Bongi, V. Bonvicini, S. Borisov, S. Bottai, A. Bruno, F. Cafagna, D. Campana, R. Carbone, P. Carlson, M. Casolino, G. Castellini, L. Consiglio, M. P. De Pascale, C. De Santis, N. De Simone, V. Di Felice, A. M. Galper, W. Gillard, L. Grishantseva, G. Jerse, A. V. Karelin, M. D. Kheymits, S. V. Koldashov, S. Y. Krutkov, A. N. Kvashnin, A. Leonov, V. Malakhov, L. Marcelli, A. G. Mayorov, W. Menn, V. V. Mikhailov, E. Mocchiutti, A. Monaco, N. Mori, N. Nikonov, G. Osteria, F. Palma, P. Papini, M. Pearce, P. Picozza, C. Pizzolotto, M. Ricci, S. B. Ricciarini, L. Rossetto, R. Sarkar, M. Simon, R. Sparvoli, P. Spillantini, Y. I. Stozhkov, A.Vacchi, E. Vannuccini, G. Vasilyev, S. A. Voronov, Y. T. Yurkin, J. Wu, G. Zampa, N. Zampa, and V. G. Zverev. The discovery of geomagnetically trapped cosmic-ray antiprotons. *The Astrophysical Journal Letters*, 737(2):L29, 2011. 1.1

[2] M.A. Aguirre, V.. Baena, J.. Tombs, and M.. Violante. A new approach to estimate the effect of single event transients in complex circuits. *Nuclear Science, IEEE Transactions on*, 54(4):1018 –1024, aug. 2007. 4.3, 4.3.2

[3] M.A. Aguirre, J.N. Tombs, F. Muñoz, V. Baena, H. Guzmán, J. Nápoles, A. Torralba, A. Fernandez-Leon, F. Tortosa-Lopez, and D. Merodio. Selective protection analysis using a seu emulator: Testing protocol and case study over the leon2 processor. *Nuclear Science, IEEE Transactions on*, 54(4):951 –956, aug. 2007. (document), 3.1, 3.2, 4.2, 4.3

[4] M.A. Aguirre. Página web de la familia Unshades: Unshades, Unshades-2 y FT-Unshades, 2004. http://www.gte.us.es/˜aguirre/Web_unshades/index. 1.6

[5] M. Alderighi, F. Casini, M. Citterio, S. D'Angelo, M. Mancini, S. Pastore, G.R. Sechi, and G. Sorrenti. Using flipper to predict irradiation results for virtex 2 devices. In *Radiation and Its Effects on Components and Systems (RADECS), 2008 European Conference on*, pages 300 –305, sept. 2008. 1.5.2

[6] Oluwole A. Amusan. *Analysis of Single Event Vulnerabilities in a 130nm CMOS Technology*. Master's Thesis, 2006. 2.2.2, 5.6

[7] L. Antoni, R. Leveugle, and B. Feher. Using run-time reconfiguration for fault injection applications. *Instrumentation and Measurement, IEEE Transactions on*, 52(5):1468 – 1473, oct. 2003. 1.5.2

[8] J. Arlat, M. Aguera, L. Amat, Y. Crouzet, J.-C. Fabre, J.-C. Laprie, E. Martins, and D. Powell. Fault injection for dependability validation: A methodology and some applications. *IEEE Transactions on Software Engineering*, 16:166–182, 1990. 1.4.1, 1.5

[9] G. Barbottin and A. Vapaille. *Instabilities in Silicon Devices: New insulators, devices, and radiation effects*. Instabilities in Silicon Devices. Elsevier, 1999. 1.2.4

[10] H. J. Barnaby. Total-ionizing-dose effects in modern cmos technologies. *Nuclear Science, IEEE Transactions on*, 53(6):3103 –3121, dec. 2006. 1.2.3

[11] Robert C. Baumann and Eric B. Smith. Neutron-induced 10b fission as a major source of soft errors in high density srams. *Microelectronics Reliability*, 41(2):211 – 218, 2001. 2.3.3

[12] R.C. Baumann. Radiation-induced soft errors in advanced semiconductor technologies. *Device and Materials Reliability, IEEE Transactions on*, 5(3):305 – 316, sept. 2005. 1.2.4

[13] A. Benso and P. Prinetto. *Fault injection techniques and tools for embedded systems reliability evaluation*. Frontiers in electronic testing. Kluwer Academic Publishers, 2003. 1.4, 2.2.1

[14] A. Benso, M. Rebaudengo, L. Impagliazzo, and P. Marmo. Fault-list collapsing for fault-injection experiments. In *Reliability and Maintainability Symposium, 1998. Proceedings., Annual*, pages 383 –388, jan 1998. 1.5.1.1

[15] F. Bezerra, R. Velazco, A. Assoum, and D. Benezech. Seu and latch-up results on transputers. In *Radiation and its Effects on Components and Systems, 1995. RADECS 95., Third European Conference on*, pages 340 –345, sep 1995. 2.1

[16] D. Binder, E. C. Smith, and A. B. Holman. Satellite anomalies from galactic cosmic rays. *Nuclear Science, IEEE Transactions on*, 22(6):2675 –2680, dec. 1975. 1.2.4

[17] J. Boue, P. Petillon, and Y. Crouzet. Mefisto-l: a vhdl-based fault injection tool for the experimental assessment of fault tolerance. In *Fault-Tolerant Computing, 1998. Digest of Papers. Twenty-Eighth Annual International Symposium on*, pages 168 –173, jun 1998. 1.4.3

[18] V.F. Cavrois, V. Pouget, D. McMorrow, J.R. Schwank, N. Fel, F. Essely, R.S. Flores, P. Paillet, M. Gaillardin, D. Kobayashi, J.S. Melinger, O. Duhamel, P.E. Dodd, and M.R. Shaneyfelt. Investigation of the propagation induced pulse broadening (pipb) effect on single event transients in soi and bulk inverter chains. *Nuclear Science, IEEE Transactions on*, 55(6):2842 –2853, dec. 2008. 5.7.2

[19] Centro Nacional de Aceleradores. Cna, 2011. http://www.cna.us.es. 5.3

[20] Kwang-Ting Cheng, Shi-Yu Huang, and Wei-Jin Dai. Fault emulation: A new methodology for fault grading. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 18(10):1487 –1495, oct 1999. 1.5, 1.5.2

[21] P. Civera, L. Macchiarulo, M. Rebaudengo, M.S. Reorda, and A. Violante. Exploiting fpga for accelerating fault injection experiments. In *On-Line Testing Workshop, 2001. Proceedings. Seventh International*, pages 9 –13, 2001. (document), 1.9, 1.5.1

[22] P. Civera, L. Macchiarulo, M. Rebaudengo, M. Sonza Reorda, and M. Violante. An fpga-based approach for speeding-up fault injection campaigns on safety-critical circuits. *J. Electron. Test.*, 18:261–271, June 2002. (document), 1.5.1.1, 1.10

[23] CLPU. Centro de láseres pulsados ultracortos ultraintensos, 2012. http://www.clpu.es/. 1

[24] T. Colladant, O. Flament, A. L'Hoir, V. Ferlet-Cavrois, C. D'Hose, and J. du Port de Potcharra. Study of transient current induced by heavy-ion in nmos/soi transistors. *Nuclear Science, IEEE Transactions on*, 49(6):2957 –2964, dec 2002. 1.2.4

[25] V. Correas, F. Saigne, B. Sagnes, J. Boch, G. Gasiot, D. Giot, and P. Roche. Innovative simulations of heavy ion cross sections in 130 nm cmos sram. *Nuclear Science, IEEE Transactions on*, 54(6):2413 –2418, dec. 2007. 5.6

[26] David de Andrés, José Albaladejo, Lenin Lemus, and Pedro Gil. Fast runtime reconfiguration for seu injection. In Mario Dal Cin, Mohamed Kaâniche, and András Pataricza, editors, *Dependable Computing - EDCC 5*, volume 3463

of *Lecture Notes in Computer Science*, pages 230–245. Springer Berlin / Heidelberg, 2005. 10.1007/11408901_18. 1.5.2

[27] C. Detcheverry, C. Dachs, E. Lorfevre, C. Sudre, G. Bruguier, J.M. Palau, J. Gasiot, and R. Ecoffet. Seu critical charge and sensitive area in a submicron cmos technology. *Nuclear Science, IEEE Transactions on*, 44(6):2266 –2273, dec 1997. 2.2.1

[28] P.E. Dodd and F.W. Sexton. Critical charge concepts for cmos srams. *Nuclear Science, IEEE Transactions on*, 42(6):1764 –1771, dec 1995. 1.2.4

[29] A. Douin, V. Pouget, D. Lewis, P. Fouillat, and P. Perdu. Electrical modeling for laser testing with different pulse durations. In *On-Line Testing Symposium, 2005. IOLTS 2005. 11th IEEE International*, pages 9 – 13, july 2005. 2.2.1

[30] L. Entrena, M. Garcia-Valderas, R. Fernandez-Cardenal, A. Lindoso, M. Portela, and C. Lopez-Ongil. Soft error sensitivity evaluation of microprocessors by multilevel emulation-based fault injection. *Computers, IEEE Transactions on*, 61(3):313 –322, march 2012. 4.3.2

[31] ESA-GSP. Displacement damage effects. *WP1 Study Report-Literature Survey and Pre-Assessment of Methods*, ESA Contract No.15157/01/NL/PA, 2002. 5.6

[32] ESA. Handbook of mitigation techniques against radiation effects for asics and fpgas, 2012. http://bit.ly/PQ4kQ2. (document), 1.1, 1.2, 1.3

[33] P. Fernandez-Martinez, J.M. Mogollón, S. Hidalgo, F.R. Palomo, D. Flores, and M.A. Aguirre. Simulation methods for ionizing radiation single event effects evaluation. In *Electron Devices, 2009. CDE 2009. Spanish Conference on*, pages 144 –147, feb. 2009. (document), 1.4, 1.8, 2.5

[34] F.Faure, P. Peronnard, R. Velazco, and R. Ecoffet. Thesic+, a flexible system for see testing. In *Proceedings of RADECS Workshop, Padova (Italy)*, pages 231–234, September 19-20, 2002. 1.4.1

[35] I. González and L. Berrojo. Supporting fault tolerance in an industrial environment: the amatista approach. In *On-Line Testing Workshop, 2001. Proceedings. Seventh International*, pages 178 –183, 2001. 1.4.3

[36] D. González. The SEUs Simulation Tool (SST), functional description, Apr 2004. European Space Agency (ESA). Document Reference TEC-EDM/DGG-SST2. 1.4.3

[37] I. Grenthe. *Nobel lectures, chemistry, 1996-2000*, volume 8. World Scientific Pub Co Inc, 2003. 1.1

[38] Steve Guccione, Delon Levi, and Prasanna Sundararajan. Jbits: Java based interface for reconfigurable computing. 1999. 1.5.2

[39] H. Guzmán-Miranda, M.A. Aguirre, and J. Tombs. Noninvasive fault classification, robustness and recovery time measurement in microprocessor-type architectures subjected to radiation-induced errors. *Instrumentation and Measurement, IEEE Transactions on*, 58(5):1514 –1524, may 2009. 1.6.1

[40] H. Guzmán-Miranda, L. Sterpone, M. Violante, M.A. Aguirre, and M. Gutiérrez-Rizo. Coping with the obsolescence of safety- or mission-critical embedded systems using fpgas. *Industrial Electronics, IEEE Transactions on*, 58(3):814 –821, march 2011. 1.6.1

[41] T. Heijmen, D. Giot, and P. Roche. Factors that impact the critical charge of memory elements. In *On-Line Testing Symposium, 2006. IOLTS 2006. 12th IEEE International*, July 2006. 1.2.4, 2.2.1

[42] Tino Heijmen. Radiation-induced soft errors in digital circuits – a literature survey, 2002. 2.1

[43] Hewlett-Packard. Hewlett-packard journal, 1977. http://bit.ly/x4EYCx. 3.1

[44] Andrew Holmes-Siedle and Len Adams. *Handbook of Radiation Effects*. Oxford university Press, 2007. 1.2.1

[45] Jin-Hua Hong, Shih-Arn Hwang, and Cheng-Wen Wu. An fpga-based hardware emulator for fast fault emulation. In *Circuits and Systems, 1996., IEEE 39th Midwest symposium on*, volume 1, pages 345 –348 vol.1, aug 1996. 1.5

[46] Brad L. Hutchings, Michael J. Wirthlin, and I Overview. Implementation approaches for reconfigurable logic applications. In *In International Workshop on Field-Programmable Logic and Applications*, pages 419–428. Springer, 1995. 1.5.2

[47] E. Ibe, K. Shimbo, H. Taniguchi, T. Toba, K. Nishii, and Y. Taniguchi. Quantification and mitigation strategies of neutron induced soft-errors in cmos devices and components. In *Reliability Physics Symposium (IRPS), 2011 IEEE International*, pages 3C.2.1 –3C.2.8, april 2011. 2.3.3

[48] JEDEC-JESD88D. Dictionary of terms for solid state technology, 2007. http://www.jedec.org/standards-documents/dictionary. 1.2.4

[49] A. H. Johnston and M. P. Baze. Mechanisms for the latchup window effect in integrated circuits. *Nuclear Science, IEEE Transactions on*, 32(6):4017 –4025, dec. 1985. 1.2.4

[50] Insoo Jun, M.A. Xapsos, S.R. Messenger, E.A. Burke, R.J. Walters, G.P. Summers, and T. Jordan. Proton nonionizing energy loss (niel) for device applications. *Nuclear Science, IEEE Transactions on*, 50(6):1924 – 1928, dec. 2003. 2.3.2

[51] F. Kaddour, S. Rezgui, R. Velazco, S. Rodriguez, and J.R. De Mingo. Error rate estimation for a flight application using the ceu fault injection approach. In *On-Line Testing Workshop, 2002. Proceedings of the Eighth IEEE International*, page 195, 2002. 1.4.2

[52] G.A. Kanawati, N.A. Kanawati, and J.A. Abraham. Ferrari: a tool for the validation of system dependability properties. In *Fault-Tolerant Computing, 1992. FTCS-22. Digest of Papers., Twenty-Second International Symposium on*, pages 336 –344, jul 1992. 1.4.2

[53] F. Lima, C. Carmichael, J. Fabula, R. Padovani, and R. Reis. A fault injection analysis of virtex fpga tmr design methodology. In *Radiation and Its Effects on Components and Systems, 2001. 6th European Conference on*, pages 275 – 282, sept. 2001. 1.4

[54] F. Lima, S. Rezgui, L. Carro, R. Velazco, and R. Reis. On the use of vhdl simulation and emulation to derive error rates. In *Radiation and Its Effects on Components and Systems, 2001. 6th European Conference on*, pages 253 – 260, sept. 2001. 1.5

[55] Books Llc. *Hash Functions: Hash Function, Pearson Hashing, Rolling Hash, Perfect Hash Function, Fowler-Noll-Vo Hash Function, Zobrist Hashing*. General Books LLC, 2010. 3.4

[56] C. López-Ongil, M. García-Valderas, M. Portela-García, and L. Entrena-Arrontes. An autonomous fpga-based emulation system for fast fault tolerant evaluation. In *Field Programmable Logic and Applications, 2005. International Conference on*, pages 397 – 402, aug. 2005. (document), 1.12, 1.13

[57] C. López-Ongil, Mario García-Valderas, Marta Portela-García, and Luis Entrena. Autonomous fault emulation: A new fpga-based acceleration system for hardness evaluation. *Nuclear Science, IEEE Transactions on*, 54(1):252 –261, feb. 2007. 4.3

[58] Henrique Madeira, Mário Z. Rela, Francisco Moreira, and João G. Silva. Rifle: A general purpose pin-level fault injector. In *EDCC*, pages 199–216, 1994. 1.4.1

[59] R. Maia, L. Henriques, D. Costa, and H. Madeira. Xceptiontm - enhanced automated fault-injection environment. In *Dependable Systems and Networks, 2002. DSN 2002. Proceedings. International Conference on*, page 547, 2002. 1.4.2

[60] T. Makino, D. Kobayashi, K. Hirose, D. Takahashi, S. Ishii, M. Kusano, S. Onoda, T. Hirao, and T. Ohshima. Soft-error rate in a logic lsi estimated from set pulse-width measurements. *Nuclear Science, IEEE Transactions on*, 56(6):3180 –3184, dec. 2009. 5.7.2, 5.7.2

[61] D.G. Mavis and P.H. Eaton. Soft error rate mitigation techniques for modern microcircuits. In *Reliability Physics Symposium Proceedings, 2002. 40th Annual*, pages 216 – 225, 2002. 2.2.3

[62] J.S. Melinger, S. Buchner, D. McMorrow, W.J. Stapor, T.R. Weatherford, A.B. Campbell, and H. Eisen. Critical evaluation of the pulsed laser method for single event effects testing and fundamental studies. *Nuclear Science, IEEE Transactions on*, 41(6):2574 –2584, dec. 1994. 1.4.1

[63] S.R. Messenger, E.A. Burke, G.P. Summers, M.A. Xapsos, R.J. Walters, E.M. Jackson, and B.D. Weaver. Nonionizing energy loss (niel) for heavy ions. *Nuclear Science, IEEE Transactions on*, 46(6):1595 –1602, dec. 1999. 2.2.2

[64] S.R. Messenger, E.A. Burke, M.A. Xapsos, G.P. Summers, R.J. Walters, Insoo Jun, and T. Jordan. Niel for heavy ions: an analytical approach. *Nuclear Science, IEEE Transactions on*, 50(6):1919 – 1923, dec. 2003. 2.2.2

[65] H. E. Michail, V. N. Thanasoulis, G. A. Panagiotakopoulos, A. P. Kakarountas, and C. E. Goutis. Efficient Pipelined Hardware Implementation of RIPEMD-160 Hash Function. In Ardil, C, editor, *Proceedings of World Academy of Science Engineering and Technology, vol 28*, volume 28 of *Proceedings of World Academy of Science Engineering and Technology*, pages 108–111, 2008. Conference of the World-Academy-of-Science, Engineering and Technology, Rome, ITALY, APR 25-27, 2008. 3.5.1

[66] J.M. Mogollón, F.R. Palomo, M.A. Aguirre, J. Nápoles, and H. Guzmán-Miranda. Simulation methodology for the validation of low energy particle accelerators as fault injection tools. In *Proceedings of the 14th European Test Symposium*, may 2009. 5.8

[67] J.M. Mogollón, R. Palomo, M.A. Aguirre, J. Nápoles, and H. Guzmán-Miranda. Simulation methodology for the validation of low energy particle accelerators as fault injection tools. In *European Test Symposium (14)*, 2009. 2.2.2, 1, 5.6

[68] J.M. Mogollón, F.R. Palomo, M.A. Aguirre, J. Nápoles, H. Guzmán-Miranda, and E. García-Sánchez. Tcad simulations on cmos propagation induced pulse broadening effect: Dependence analysis on the threshold voltage. *Nuclear Science, IEEE Transactions on*, 57(4):1908 –1914, aug. 2010. (document), 1.2.4, 1.6, 2.2, 2.5, 5.7.2, 5.8

[69] J.M. Mogollón, H. Guzmán-Miranda, J. Nápoles, J. Barrientos, and M.A. Aguirre. Ftunshades2: A novel platform for early evaluation of robustness against see. In *Radiation and Its Effects on Components and Systems (RADECS), 2011 12th European Conference on*, pages 169 –174, sept. 2011. 5, 4.5

[70] J.M. Mogollón, J. Nápoles, H. Guzmán-Miranda, and M. A. Aguirre. Real time seu detection and diagnosis for safety or mission-critical ics using hash library-based fault dictionaries. In *Radiation and Its Effects on Components and Systems, 2011. 12th European Conference on*, sept. 19-23 2011. 3.1, 3.6, 4.5

[71] J.M. Mogollón. *Técnicas para la simulación de los efectos de la radiación cósmica en un dispositivo CMOS*. S.l., 2008. [Recurso electrónico] : proyecto fin de Máster /autor, Juan M. Mogollón García; 30 cm +1 CD-ROM; Univ. Sevilla, ESI – http:// bibing.us.es/proyectos/abreproy.php?proyecto=70054. 2.2

[72] J.M. Mogollón. Página web de RadUS y Ft-Unshades, 2009. http:// walle.us.es/ftunshades. 1.6

[73] Y. Morilla, J. García López, and J.A. Labrador. New facilities for ion irradiation of space application devices at the centro nacional de aceleradores (spain). In *Proceedings of the 8th International Workshop on Irradiation Effects on Semiconductor Devices for Space Applications (). Tsukuba, Japon. Jaxa-Jaea*, pages 199–202, 2008. 5.3

[74] Y. Morilla, M. C. Jiménez-Ramos, J. García López, J. A. Labrador, F. R. Palomo, and I. Ortega-Feliu. Developing the iba equipment to increase the versatility of the cna. *Nuclear Instruments and Methods in Physics Research, Section B: Beam Interactions with Materials and Atoms*, 273:218–221, 2012. 5.3, 5.6

[75] N. Naber, T. Getz, Yong Kim, and J. Petrosky. Real-time fault detection and diagnostics using fpga-based architectures. In *Field Programmable Logic and Applications (FPL), 2010 International Conference on*, pages 346 –351, 31 2010-sept. 2 2010. 3.2

[76] J. Nápoles, H. Guzmán, M. Aguirre, J. N. Tombs, F. Muñoz, V. Baena, A. Torralba, and L. G. Franquelo. Radiation environment emulation for VLSI designs: A low cost platform based on Xilinx FPGA's. In *Industrial Electronics, 2007. ISIE 2007. IEEE International Symposium on*, pages 3334–3338, Vigo, June 2007. 1.6, 4.5

[77] J. Nápoles, H. Guzmán-Miranda, M. Aguirre, J.N. Tombs, J.M. Mogollón, R. Palomo, and A.P. Vega-Leal. A complete emulation system for single event effects analysis. In *Programmable Logic, 2008 4th Southern Conference on*, pages 213 –216, march 2008. 4.5

[78] NASA/GSFC. Proton test guideline development (lessons learned), 2002. http://radhome.gsfc.nasa.gov/radhome/papers/Proton_testing_guidelines_2002.pdf. 1.2, 2.3.2

[79] M. Nicolaidis. Design for soft error mitigation. *Device and Materials Reliability, IEEE Transactions on*, 5(3):405 – 418, sept. 2005. 1.3

[80] B. Nicolescu, N. Gorse, Y. Savaria, E.M. Aboulhamid, and R. Velazco. On the use of model checking for the verification of a dynamic signature monitoring approach. *Nuclear Science, IEEE Transactions on*, 52(5):1555 – 1561, oct. 2005. 3.2

[81] L.C. Noll. Fowler-Noll-Vo hash function, 2011. http://bit.ly/zpP6nZ. 3.4

[82] F.R. Palomo, J.M. Mogollón, M.A. Aguirre, J. Nápoles, C. Mendez, J.R. Vazquez de Aldana, and P. Moreno. Method for estimation of critical charge in laser experiments. In *Radiation and Its Effects on Components and Systems (RADECS), 2008 European Conference on*, sept. 2008. 2.5

[83] F.R. Palomo, J.M. Mogollón, J. Nápoles, H. Guzmán-Miranda, A.P. Vega-Leal, M.A. Aguirre, P. Moreno, C. Mendez, and J.R. Vazquez de Aldana. Pulsed laser seu cross-section measurement using coincidence detectors. In *Radiation and Its Effects on Components and Systems (RADECS), 2008 European Conference on*, pages 147 –151, sept. 2008. 2.5, 3.6

[84] F.R. Palomo, J.M. Mogollón, J. Nápoles, H. Guzmán-Miranda, J.A. Rodriguez, A.P. Vega-Leal, M.A. Aguirre, J. Tombs, C. Mendez, J.R. Vazquez de Aldana, P. Moreno, and L. Roso. Start-up of a pulsed laser test system for single

event effects analysis. In *Proceedings of the 14th European Test Symposium*, may 2009. 2.5

[85] F.R. Palomo, J.M. Mogollón, J. Nápoles, H. Guzmán-Miranda, A.P. Vega-Leal, M.A. Aguirre, P. Moreno, C. Mendez, and J.R.V. de Aldana. Pulsed laser seu cross section measurement using coincidence detectors. *Nuclear Science, IEEE Transactions on*, 56(4):2001 –2007, aug. 2009. (document), 1.7, 2.3.4, 3.2, 3.2, 5.1

[86] F.R. Palomo, P. Fernández-Martínez, J. M. Mogollón, S. Hidalgo, M. A. Aguirre, D. Flores, I. López-Calle, and J. A. de Agapito. Simulation of femtosecond pulsed laser effects on mos electronics using tcad sentaurus customized models. *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, 23(4-5):379–399, 2010. 2.5

[87] F.R. Palomo, J.M. Mogollón, J. Nápoles, and M.A. Aguirre. Mixed-mode simulation of bit-flip with pulsed laser. *Nuclear Science, IEEE Transactions on*, 57(4):1884 –1891, aug. 2010. 2.2, 2.5, 5.4

[88] F.R. Palomo, J.M. Mogollón, Y. Morilla, J. Garcia-Lopez, M.C. Jimenez-Ramos, J.A. Labrador, M.A. Cortes-Giraldo, J.M. Quesada, and M.A. Aguirre. Seu threshold model and its experimental verification. In *Radiation and Its Effects on Components and Systems (RADECS), 2011 12th European Conference on*, pages 420 –425, sept. 2011. 2.5

[89] F.R. Palomo, Y. Morilla, J.M. Mogollón, J. García-López, J.A. Labrador, and M.A. Aguirre. Early works on the nuclear microprobe for microelectronics irradiation tests at the ceici (sevilla, spain). *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, 269(20):2210 – 2216, 2011. 12th International Conference on Nuclear Microprobe Technology and Applications. 2.2.3, 5.6, 5.8

[90] I. Pomeranz, S. Venkataraman, and S.M. Reddy. Fault diagnosis and fault model aliasing. In *VLSI, 2005. Proceedings. IEEE Computer Society Annual Symposium on*, pages 206 – 211, May 2005. 4.3

[91] V. Pouget, H. Lapuyade, D. Lewis, Y. Deval, P. Fouillat, and L. Sarger. Spice modeling of the transient response of irradiated mosfets. *Nuclear Science, IEEE Transactions on*, 47(3):508 –513, jun 2000. 2.2.1

[92] H.M. Quinn, P.S. Graham, M.J. Wirthlin, B. Pratt, K.S. Morgan, M.P. Caffrey, and J.B. Krone. A test methodology for determining space readiness of xilinx sram-based fpga devices and designs. *Instrumentation and Measurement, IEEE Transactions on*, 58(10):3380 –3395, oct. 2009. 5.1

[93] P. Rech, A. Paccagnella, M. Grosso, M.S. Reorda, F. Melchiori, D. Loparco, and D. Appello. Evaluating the impact of dfm library optimizations on alpha-induced seu sensitivity in a microprocessor core. *Nuclear Science, IEEE Transactions on*, 57(4):2098 –2105, aug. 2010. 1.7, 6.2

[94] P. Reyes, P. Reviriego, J. A. Maestro, and O. Ruano. New protection techniques against SEUs for moving average filters in a radiation environment. *IEEE Transactions on Nuclear Science*, 54:957–964, August 2007. 1.4.3

[95] S. Rezgui, J.J. Wang, E.C. Tung, B. Cronquist, and J. McCollum. Comprehensive see characterization of 0.13 um flash-based fpgas by heavy ion beam test. In *Radiation and Its Effects on Components and Systems, 2007. RADECS 2007. 9th European Conference on*, pages 1 –6, sept. 2007. 2.4

[96] S. Rezgui, P. Louris, and R. Sharmin. See characterization of the new rtax-dsp (rtax-d) antifuse-based fpga. *Nuclear Science, IEEE Transactions on*, 57(6):3537 –3546, dec. 2010. 2.4

[97] Satellite News Digest. Satellite outages and failures, 2011. http://www.sat-index.co.uk/failures/. 1.1

[98] A. Satoh and T. Inoue. Asic hardware focused comparison for hash functions md5, ripemd-160, and shs. In *Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference on*, volume 1, pages 532 – 537 Vol. 1, april 2005. 3.4

[99] Sentaurus. *TCAD Tools*. 2007. Synopsys, Inc. 2.2.3

[100] F.W. Sexton, W.T. Corbett, R.K. Treece, K.J. Hass, K.L. Hughes, C.L. Axness, G.L. Hash, M.R. Shaneyfelt, and T.F. Wunsch. Seu simulation and testing of resistor-hardened d-latches in the sa3300 microprocessor. *Nuclear Science, IEEE Transactions on*, 38(6):1521 –1528, dec 1991. 2.4

[101] P. Shivakumar, M. Kistler, S.W. Keckler, D. Burger, and L. Alvisi. Modeling the effect of technology trends on the soft error rate of combinational logic. In *Dependable Systems and Networks, 2002. DSN 2002. Proceedings. International Conference on*, pages 389 – 398, 2002. 2.1

[102] J.R. Srour, C.J. Marshall, and P.W. Marshall. Review of displacement damage effects in silicon devices. *Nuclear Science, IEEE Transactions on*, 50(3):653 – 670, june 2003. 1.2.2

[103] A. K. Sutton, R. Krithivasan, P. W. Marshall, M. A. Carts, C. Seidleck, R. Ladbury, J. D. Cressler, C. J. Marshall, S. Currie, R. A. Reed, G. Niu,

B. Randall, K. Fritz, D. McMorrow, and B. Gilbert. Seu error signature analysis of gbit/s sige logic circuits using a pulsed laser microprobe. *Nuclear Science, IEEE Transactions on*, 53(6):3277 –3284, dec. 2006. 2.3.4

[104] A. Taber and E. Normand. Single event upset in avionics. *Nuclear Science, IEEE Transactions on*, 40(2):120 –126, apr 1993. 2.1

[105] The MOSIS Service. Mosis, 2011. http://www.mosis.com/. 1.7, 2.2.2

[106] A. D. Tipton, J. A. Pellish, R. A. Reed, R. D. Schrimpf, R. A. Weller, M. H. Mendenhall, B. Sierawski, A. K. Sutton, R. M. Diestelhorst, G. Espinel, J. D. Cressler, P. W. Marshall, and G. Vizkelethy. Multiple-bit upset in 130 nm cmos technology. *Nuclear Science, IEEE Transactions on*, 53(6):3259 –3264, dec. 2006. (document), 4.3.3, 4.3

[107] J. Tombs, M.A. Aguirre, R. Palomo, J.M. Mogollón, H. Guzmán, J. Nápoles, A. Rodriguez-Perez, J.A. Rodriguez, A.P. Vega-Leal, Y. Morillas, and J. Garcia. The experience of starting-up a radiation test at the 18mev cyclotron in the spanish national accelerators center. In *Radiation and Its Effects on Components and Systems, 2007. RADECS 2007. 9th European Conference on*, pages 1 –5, sept. 2007. (document), 1.7, 2.3.2, 2.7, 5.1, 5.8

[108] R. Velazco, S. Karoui, and T. Chapuis. Seu testing of 32-bit microprocessors (for space application). In *Radiation Effects Data Workshop, 1992. Workshop Record., 1992 IEEE*, pages 16 –20, 1992. 2.1

[109] R. Velazco, R. Leveugle, and O. Calvo. Upset-like fault injection in vhdl descriptions: A method and preliminary results. In *Defect and Fault Tolerance in VLSI Systems, 2001. Proceedings. 2001 IEEE International Symposium on*, pages 259 –267, 2001. (document), 1.5.1.1, 1.11

[110] Raoul Velazco, Pascal Fouillat, and Ricardo Reis (Editors). *Radiation Effects on Embedded Systems*. Springer, 2007. 1.1

[111] R. Velazco, Ph. Cheynet A., and Bofill R. Ecoffet. Thesic: A testbed suitable for the qualification of integrated circuits devoted to operate in harsh environment. In *IEEE European Test Workshop (ETW'98), Sitges, (Spain)*, pages 89–90, may 1998. 1.4.1

[112] J.T. Wallmark and S.M. Marcus. Minimum size and maximum packing density of nonredundant semiconductor devices. *Proceedings of the IRE*, 50(3):286 –298, march 1962. 1.2.4

[113] P.H. Yannakopoulos, A.P. Skountzos, and M. Vesely. Influence of ionizing radiation in electronic and optoelectronic properties of iii-v semiconductor compounds. *Microelectronics Journal*, (39):732–736, 2008. 2.2.2

[114] J.F. Ziegler, J.P. Biersack, and M.D. Ziegler. *SRIM The Stopping and Range of Ions in Matter*. SRIM Co., 2008. 2.2.2, 5.6

Glossary

AEC Automotive Electronics Council. 36

API Application Programming Interface. 45

ASER Accelerated Soft Error Rate. 53, 54, 64

ASIC Application Specific Integrated Circuit. 35, 50, 54, 63, 66, 86, 98, 101

ATE Automated Test Equipment. 66, 73, 84, 86, 88, 98, 106, 108, 112, 117, 123

BIST Built-in Self-Test. 74

BJT Bipolar Junction Transistor. 55

BPSG Borophosphosilicate Glass. 62

CEU Code Emulated Upset. 38

CMOS Complementary Metal Oxide Semiconductor. 30, 32, 33, 35, 49, 54, 55, 57, 59, 62, 68, 101, 105, 109, 114

CNA National Accelerators Center. 49, 54, 57, 59, 61, 100, 108, 109

COTS Commercial Off the Shelf. 53, 123

cps counts per second. 105

CPU Central Process Unit. 90

CRC Cyclic Redundancy Check. 74

CTR Compile-Time Reconfiguration. 45

CUT Circuit Under Test. 36, 45, 117

DIL Dual in-Line. 101

DRAM Dynamic Random Access Memory. 26, 53

DUT Device Under Test. xi, xii, 37–40, 42, 45, 65, 68, 72, 78, 79, 81, 106, 108, 109, 112, 114, 115, 118, 120, 122, 123, 129

EC Emulation Controller. 43

ESA European Space Agency. 35, 36

FARM Faults-Activation-Readouts-Measurements. 39

FF Flip-Flop. 34, 40, 42, 44–46, 48, 78, 79, 84, 87, 88, 91, 101, 102, 105, 110–112, 116, 118, 123

FI Fault Injection. 37

FIM Fault Injection Manager. 41

FNV Fowler-Noll-Vo Hash Function. 75–79, 88, 102, 105

FPGA Field Programmable Gate Array. 35, 39, 40, 42–48, 66, 68, 70–74, 76, 78, 84, 86, 88, 97, 98, 106, 108, 117, 122, 123, 127

FT-UNSHADES Fault Tolerant University of Seville Hardware Debugging System. 45–49, 73, 83, 84, 87–89, 92, 97, 98, 105, 112, 113, 117, 121, 127

HDL Hardware Description Language. 38–40, 44, 65, 88, 122

HWFI Hardware-Based Fault Injection. 38, 39, 45

IBA Ion Beam Analysis. 54, 108

JEDEC Joint Electron Device Engineering Council. 36, 50, 69

JONIC JONathan Tombs Integrated Circuit. 49, 50, 90, 91, 98, 101, 112, 129

LEO Low Earth Orbit. 25

LET Linear Energy Transfer. 31, 54, 55, 59–61, 64, 105, 108–110

LSS Linhard-Scharff-Schiott. 56

MBU Multiple Bit Upset. 32, 37, 48, 66, 89, 110, 111, 116, 121

MCU Multiple Cell Upset. 32, 37, 48, 50, 66

MOS Metal Oxide Semiconductor. 27, 31

MUT Module Under Test. 47, 84, 86, 127

NIEL Non-Ionizing Energy Loss. 30, 110

OTP One Time Programmable. 66, 68

PCB Printed Circuit Board. 106, 108, 129

PIPB Propagation Induced Pulse Broadening. 33, 55, 63, 114

PS Parallel to Serial. 101, 102, 116

RAM Random Access Memory. 42, 43

RTL Register Transfer Level. 37–39

RTR Run-Time Reconfiguration. 45

RTSER Real Time Soft Error Rates. xi

SAA South Atlantic Anomaly. 25

SBFI Simulation-Based Fault Injection. 38, 39

SEB Single Event Burnout. 31

SEE Single Event Effect. xi, xii, 30, 37, 54, 60–66, 68, 86, 101, 112

SEFI Single Event Functional Interruption. 31

SEGR Single Event Gate Rupture. 31

SEH:SHE Single Event Hard Error. 31

SEL Single Event Latch-up. 31, 32

SEP Single Event Phenomena. 30

SER Soft Error Rate. 35, 36, 53, 54

SET Single Event Transient. 32, 37, 53, 55, 59, 66, 68, 74, 89, 102, 112, 114, 116, 118, 120, 121

SEU Single Event Upset. 31, 32, 34, 38, 50, 53, 55, 58, 59, 62–64, 66, 68, 84, 86, 88, 89, 109–111, 116, 117, 121

SoC System on Chip. 35

SPA Single Photon Absorption. 63

SR Shift Register. 101, 102, 105, 116, 120

SRAM Static Random Access Memory. 31, 34, 35, 39, 44, 50, 53, 55, 59, 62, 66, 70, 109, 123

SRIM Stopping Range of Ions in Matter. 57, 58, 109, 110

SSER System Soft Error Rate. 53

SWFI Software-Based Fault Injection. 39

TCAD Technology Computer Aided Design. 34, 35, 49, 57–59, 68

TID Total Ionizing Dose. 61, 64, 105

TMR Triple Modular Redundancy. 68

TPA Two Photon Absorption. 63

VHDL Very-High Speed Integrated Circuit Hardware Description Language. 39, 63, 78, 79, 100, 106

XRTC Xilinx Radiation Test Consortium. 45