1-1-2010

Fault Detection For ASIC Design Reliability On Resistive Delay Faults And Strength-Based Soft-Errors

Mohammad R.S. Javaheri
Ryerson University

FAULT DETECTION FOR ASIC DESIGN RELIABILITY ON RESISTIVE DELAY FAULTS AND STRENGTH-BASED SOFT-ERRORS

By

Mohammad Reza Samadpour Javaheri

Master of Applied Science

Electrical and Computer Engineering

Ryerson University, Toronto, Canada, 2006

Bachelor of Computer Engineering

Azad University of Tehran, Iran, 1997

A Dissertation

Presented to Ryerson University

In partial fulfillment of the

Requirements for the degree of

Doctorate of Philosophy

In the program of

Electrical and Computer Engineering

Toronto, Ontario, Canada, 2010

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this thesis.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Signature

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Signature

ABSTRACT

Thesis Title:

Fault Detection for ASIC Design Reliability on Resistive Delay Faults and Strength-Based Soft-Errors

Mohammad Reza Samadpour Javaheri

Electrical and Computer Engineering, Doctorate of Philosophy

Ryerson University 2010

Thesis Directed by:

Dr. Reza Sedaghat

Electrical and Computer Engineering Department

Ryerson University

Soft-errors (SEs) and delay faults (DFs) frequently occur in modern high-density, high-speed, low-power VLSI circuits. Therefore, SE hardened design and DF testing are essential. This thesis introduces two novel methods for soft-error detection and delay fault propagation in nanometre technology. A new idea is proposed to propagate those delay faults that are not causing logic failure at the site of the defect, but the delay makes the circuit more prone to soft-errors that manifest the effect of delay faults. This approach propagates the fault from the fault location by mapping a nine-valued voltage model on top of a five-valued voltage model to convert delay faults to static faults. This original idea reduces the complexity of delay fault propagation. This thesis introduces an original approach toward soft-error detection based on the strength violation in the circuit. This research shows that transient pulses of less than threshold voltage will cause soft-errors without altering the logic value at the strike location. This method will increase the Soft-Error Rates (SER) for all existing methods if strength-based Soft-Error detection will be considered. The offered approach uses a novel coding system that carries both logic and strength which applies to certain logic functions that are sensitive to strength variations. A wide range of soft-errors are the result of strength violation in switch-level that have never been investigated before.

ACKNOWLEDGEMENT

I am heartily thankful to my advisor, Dr. Reza Sedaghat, whose encouragement, guidance and support from the initial to the final stage of my research enabled me to achieve my goals.

Also, I offer my regards and blessings to all of those who supported me in any respect during the completion of the PhD.

I would like to thank my family for their support. Above all, I cannot express my full gratitude to my parents, who patiently advised me throughout my life.

I dedicate this thesis to my mother and my deceased father.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

ASER	Accelerated Soft Error Rate
BPSG	Borophosphosilicate glass
CMOS	Complementary Metal-Oxide Semiconductor
CRAM	Chalcogenide Random Access Memory
CU	Core Unit
DF	Delay Fault
DSF	Delay Fault Simulation
DTPG	Deterministic Test Pattern Generator
FC	Fault Coverage
FM	Fault Memory
FPGA	Field Programmable Gate Array
GD	Gold Device
HDL	Hardware Description Language
IDM	Input Data Memory
LFSR	Linear Feedback Shift Register
MAC	Memory Address Counter
MVL5	Multi-Valued Logic 5
MVL9	Multi-Valued Logic 9
OU	Observation Unit
SCCU	Soft-error Coverage Calculation Unit
SE	Soft Error

SER	Soft Error Rate
SET	Single Event Transient
SEU	Single Event Upset
SIU	Soft-error Injection Unit
SOC	System On Chip
SRAM	Static Random Access Memory
TPC	Test Pattern Counter
VLSI	Very Large Scale IC

CHAPTER 1
INTRODUCTION

Soft errors caused by ionizing radiation have emerged as a major concern for current generation of CMOS technologies and the trend is expected to get worse. A method is invented to detect soft-errors cause by transient strikes less than threshold voltage. The novel aspect of proposed approach proves that a significant amount of soft-errors are the result of strength violation in a circuit that never been investigated nor realized by exiting methods. More specifically an algorithm is designed to detect soft-errors at the switch-level caused by current spikes which can affect the driving strength. The offered approach uses a novel coding system to be applied in certain functions that are sensitive to strength variations. It is able to detect even the slight changes in signal strength caused by both cosmic rays and alpha particle from package contamination. Most soft-error detection techniques sense the logic changes in the circuit while this method proves that a wide range of soft-errors are the result of strength violation in switch-level. Experimental results illustrate the importance of accurate simulation methods and stress the effect of driving strength changes in switch-level for soft-error detection in today's technology

1.1. Problem Statement

In nanometer technologies, circuits are increasingly sensitive to various kinds of perturbations. Alpha particles and atmospheric neutrons induce single-event upsets (SEU) that affect memory cells, latches, and flip-flops. In addition, single-event transients (SET) can be initiated in the combinational logic and captured by the latches and flip-flops associated with the logic outputs. Designers cannot control the sources of soft-errors, but their effects can be mitigated through soft-error detection techniques. This thesis presents a unique strength-based soft-error detection method targeting soft-errors caused by transient pulses of magnitude less than logic threshold. All existing soft-error detection models are based on transient pulses greater than threshold voltage that are able to alter the logic value in the circuit. In this thesis, the transient pulses less than threshold voltage are considered as strength-based soft-errors and are referred to as Strength Violation (SV).

On the hand, other in advanced technologies, an increasing proportion of defects manifest themselves as small delay faults. This thesis describes a technique to propagate delay faults caused by resistive bridging in the circuit. Most of today's advanced delay fault algorithms are able to propagate those delay faults which create logic or glitch faults. Here, those delay faults that are not causing logic failure are propagated, but the delay makes the circuit more prone to soft-errors that manifest the effect of delay faults.

1.2. The Method

A novel soft-error detection concept is used, which assumes that voltage fluctuations smaller than logic threshold can eventually result in soft-errors. Advanced switch-level models were

designed to not only mimic important characteristics of transistor-level circuits i.e. bidirectional signal flow, driving strength variations and node capacitances, but also to use driving strengths to model strength violation. The resulting switch-level models eliminate the complexity associated with state-of-the-art transistor level simulators while achieving the desired amount of accuracy and faster simulation. The aim of this thesis is to interpret various parameters used in these strength-based switch models in order to find an efficient way of injecting transients into complex logic circuits. This method injects transient errors at the gates and drains of all the switches inside a given circuit. A detailed fault model is devised to cover different categories of soft-errors based on the magnitude of the injected voltage pulse. An emulation system is implemented to create detailed profiles of each transient error type injected into switch-level implementations of ISCAS'85 benchmarks. The collected data are used to calculate the SER and the results are analysed to show the effectiveness and accuracy of the strength-based detection method. Furthermore, the transient-equivalence technique is applied to minimize the number of injected error types and, thus speed-up the detection process as well as overcome resource overhead.

The proposed method of propagating delay faults caused by resistive bridging defects in the circuit has a revolutionary approach in advanced CMOS testing methodology. When a physical defect leads to excessive delays on signals instead of altering the logic function of the circuit, it is no longer a static defect. Such unknown and unpredictable defect behaviour makes it very difficult to analyze a fault. The dynamic nature of such faults disturbs the timing of the logic propagation. The relation between the logic propagation delay and its eventual effect on the circuit output voltage is determined by performing a switch-level analysis on CMOS primitive

gates. Consequently, the voltage, which is carrying a timing disturbance, should be propagated to the primary output. As the delay size is relatively small, the voltage changes do not cause a logical problem at the gate output making it difficult to trace it to the output. To resolve this problem, a nine-valued voltage model is used on top of a five-valued voltage model to propagate faulty signals. Those faults that are causing logical faults in a nine-valued voltage model are still delay faults in a five-valued voltage model. Dynamic behaviours of resistive defects tend to delay the correct logic state propagation at the gate output. Various factors can contribute to the delay, such as certain fault locations with respect to the input vector the gate is subjected to, defect resistance of the fault, and the technology variation. In most cases, faults in a five-valued voltage model only disturb the logic propagation time without adversely affecting the functional output. As a result, the output voltage fluctuates between ranges of intermediate voltage value. However, the disturbance of the propagation time can greatly affect the functional output in a nine-valued voltage model. By reducing feature sizes, resistive fault occurrences are expected to increase. Hence, their effects on the logic voltage-levels in static CMOS primitive gates can be determined subject to 65nm, 45nm and 32nm technologies.

1.3. The Outcomes

This research proves that a wide range of soft-errors are the result of strength violation in switch-level. The experimental results reported for ISCAS'85 benchmarks show an average rate between '0.7' and '0.88'. The presented work goes beyond the normal logic change based on soft-error detection techniques currently used and towards the design of more precise and efficient soft-error detection algorithms. The emulation-based soft-error detection achieved significant speed-up of the order of 10^6 as compared to a customized simulation-based method.

On the other hand, a Fault Coverage (FC) of 37% to 100% has been achieved for resistive short delay faults on several ISCAS'85 and ISCAS'89 benchmark circuits. The number of switches and injected files and, eventually, the delay fault coverage have been calculated using the proposed method. The fault coverage reported above has been obtained for those resistive faults that are not causing logic faults on the site of defect.

Software simulation and hardware emulation were implemented to evaluate and maintain the proposed methods. Although there are no comparable publications in the area of deep submicron, it is clear that the results presented here indicate noteworthy fault coverage and CPU time.

1.4. Summary of Contributions

This thesis focuses on resistive delay faults and soft-error detection in advanced nanotechnology. The first part of this thesis is geared toward resistive short and open delay fault detection. As the soft-errors manifest the effect of delay faults in the first part, the second part of this thesis focuses on soft-error detection. The overall research contributions of this thesis are summarised as follows:

- A novel idea is proposed to propagate those delay faults that are not causing logic failure at the location of defect, but the delay makes the circuit more prone to soft-errors as they manifest the effect of delay faults. This algorithm can be implemented for varied

applications, i.e. switch level min-max mode grading of robust/non-robust delay test vectors and analysis of dynamic hazards for delay fault diagnosis for general switch-level circuits.

- An arithmetic algorithm is created capable of addressing all possible physical problems in switch-level for CMOS technology-based circuits of any size. This algorithm is capable of propagating delay faults to the primary output. Even when the delay size is relatively small, the algorithm maps a MVL9 on top of a MVL5 in order to propagate those delays that are not causing logical failure at the fault location.

- A novel approach was invented to detect soft-errors based on strength violation in the circuit. For strength-based soft-error detection, a novel coding system and logic functions were developed that are sensitive to strength variations. This approach proves that a wide range of soft-errors are the result of strength violation in switch-level and have never been investigated before.

- Delay Fault Simulation (DFS) software was also developed to inject faults into the circuit and measure the fault coverage. The effect of non-logical delay faults on soft-errors was proven to show how delay faults manifest soft-errors.

- A soft-error model for strength violation is presented. This model enables the detection of soft-errors due to transient pulses of magnitude less than logic threshold. Soft-errors are classified into 23 different types based on the logic level and strength level of the injected voltage pulse.

- A new architecture was designed to enable the implementation of the proposed switch-level model on the field-programmable gate array. The emulation system was designed to create

detailed profiles of each transient error type injected into the switch-level implementations of ISCAS'85 benchmarks.

- The effectiveness and accuracy of the strength-based detection method was measured by running simulation/emulation for the proposed theory on soft-error detection. The speed-up of 10^6 was achieved by FPGA-based emulation as compared to simulation-based detection methods.

Figure 1 shows the thesis contribution in a glance.

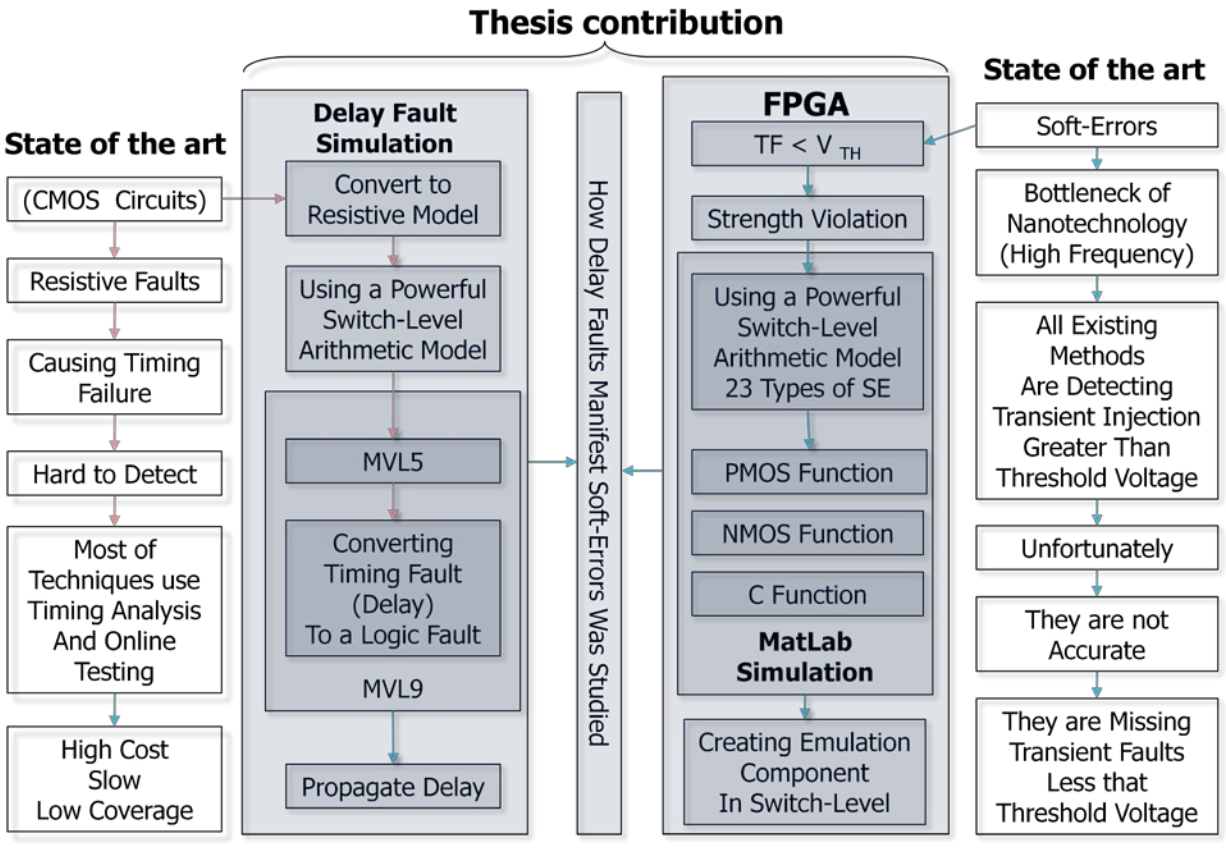


Figure 1 - Thesis contribution diagram.

1.5. Organization of Thesis

The chapters of the thesis are organized as follows: Chapter 1 introduces the thesis and its contributions. Chapter 2 discusses the state-of-the-art of soft-errors and resistive delay faults. Chapter 3 explains the novel idea proposed for resistive delay faults. This chapter also describes the dynamic effect of resistive faults on propagation delay and output voltage in nanometre technology. Chapter 4 demonstrates the effect of delay faults on soft-errors and proves how delay faults manifest soft-errors. Chapter 5 opens a new domain in soft-error detection science by introducing a strength-based soft-error. This chapter also presents a novel switch-level model for soft-error detection. Chapter 6 discusses soft-error injection techniques in the different locations of the circuit. The simulation results are presented in Chapter 7. Chapter 8 explains the emulation architecture used for strength-based soft-error detection on FPGA. Chapter 9 presents the experimental results obtained for the ISCAS benchmark series. Chapter 10 concludes this thesis and discusses the future prospects of this research, ending with a list of cited publications and references.

CHAPTER 2
STATE OF THE ART REVIEW

This chapter is undertaking a literature review on Resistive Delay Faults and Soft-errors, and will discuss roots of defects and their impacts on the circuit for each area separately. This chapter is also discussing the latest related research studies and their solutions for Resistive Delay Faults and Soft-Errors.

2.1. Overview

Present testing techniques for VLSI circuits face many exciting and complex challenges. In the era of large systems embedded in a single system-on-chip (SOC) and fabricated in continuously shrinking technologies, it is important to ensure correct behaviour of the whole system. Electronic design and test engineers have to deal with these complex and heterogeneous systems (digital, mixed-signal, memory), but few have the possibility to study the whole field in a detailed manner. In high-density, high-speed and low-power VLSIs, soft-errors (SEs) and delay faults (DFs) occur frequently. Therefore, SE hardened design and DF testing are essential. This

thesis proposes two revolutionary fault detection methods for the most critical VLSI testing areas of “Resistive Delay Faults” and “Soft-Errors”.

2.2. Resistive Delay Faults

The need for testing timing defects is further expected to grow with the current design trend towards deep submicron devices. After a long period of the prevailing belief that high stuck-at fault coverage is sufficient to guarantee high quality of shipped products, industry is now forced to rethink other types of testing. Delay testing has been a topic of extensive research both in industry and in academia for more than a decade. As a result, several delay fault models and numerous testing methodologies have been proposed. Delay Fault Testing for VLSI Circuits presents a selection of existing delay testing research results combining introductory material with state-of-the-art techniques that address some of the current problems in delay testing. Delay Fault Testing for VLSI Circuits covers some basic topics such as fault modeling and test application schemes for detecting delay defects.

2.3. Resistive Short and Open Delay Faults

CMOS fabrication of digital integrated circuits includes defects that cannot be represented using conventional idealistic stuck-at or bridging fault models. Unfortunately, such defects represent a significant percentage of faults in complex digital circuits. A fault occurs when two nodes are unintentionally connected together without resistance. In reality, parasitic resistance “R”, capacitance “C”, and inductance “L” are always associated with the defects in very large-

scale integration circuits [7][8]. Due to various deep submicron effects, a circuit may fail to operate at the desired clock frequency. Timing failure analysis is a technique used to locate the source of timing failures. The resolution and the hit rate of the candidates, which are reported by the delay-fault diagnosis process, will determine the efficiency of timing failure analysis. The resolution is defined as the ratio of the number of real fault sites to the total number of the reported candidates. Unfortunately, even the most recently published delay-fault diagnosis methodologies suffer from poor resolution or low scalability. In [9] the authors have introduced a method for resistive open and bridging faults. They have mentioned that their *"result accounts only for logical conditions and the actual coverage value can be computed once the transistor-level analysis has been performed."* However, this thesis presents a new idea to propagate those delay faults that are not causing logical conditions, but lower the frequency of operations and also increase the soft-error rates in CMOS technology.

In another recent publication [10], the testability of small delays due to resistive opens are analyzed considering the process variations. A statistical methodology to estimate the fault coverage of these defects is proposed. Using the proposed methodology, the Statistical Fault Coverage (SFC) of resistive opens producing small delays is evaluated for some ISCAS benchmark circuits. The authors mention *"SFCs higher than 50% are obtained for the circuits C432, C499 and C1908. One of the reasons for these values may be the small delay is propagated through all the possible paths starting from open location. This propagation is limited by the MAX function. SFCs lower than 50% is obtained for the circuits C2670 and C3540."*

When a physical defect leads to excessive delays on signals instead of altering the logic function of the circuit, it is no longer a static defect. Such unknown and unpredictable defect

behaviours make it very difficult to analyze a fault. The dynamic nature of such faults disturbs the timing of the logic propagation. In this approach, the timing failures, which are caused by short and open resistive faults inside the gates, will affect the level of the voltage at the gate output. The relation between the logic propagation delay and its eventual effect on the circuit output voltage is determined by performing a switch-level analysis on CMOS primitive gates. Consequently, the voltage, which is carrying timing disturbance, should be propagated to the primary output. As the delay size is relatively small, the voltage changes do not cause a logical problem at the gate output making it difficult to trace it to the output. To resolve this problem, a nine-valued voltage model is used on top of a five-valued voltage model to propagate faulty signals. The main concept is that those faults that cause logical faults in a nine-valued voltage model are still delay faults in a five-valued voltage model. Dynamic behaviours of resistive defects tend to delay the correct logic state propagation at the gate output.

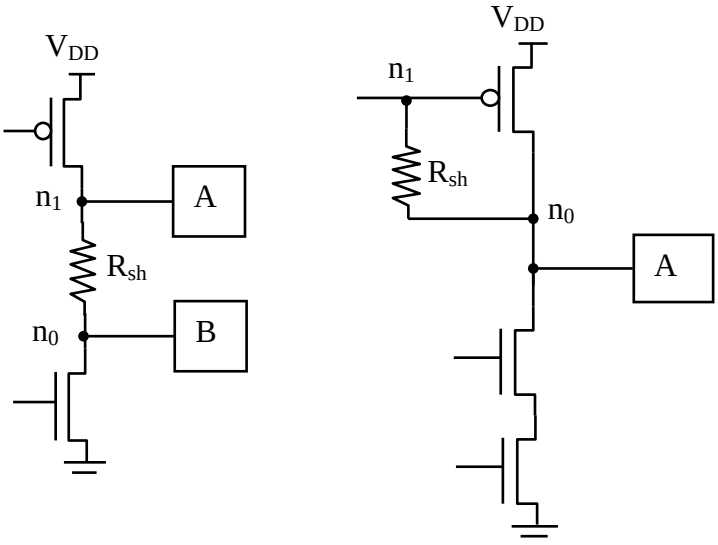


Figure 2 - Switch-Level view of resistive fault in CMOS.

Various factors can contribute to the delay such as certain fault locations with respect to the input vector the gate is subjected to, defect resistance of the fault, and the technology variation.

In most cases, faults in a five-valued voltage model only disturb the logic propagation time without adversely affecting the functional output. As a result, the output voltage fluctuates between ranges of intermediate voltage value. However, the disturbance of the propagation time can greatly affect the functional output in a nine-valued voltage model. Figure 2 illustrates resistive faults in CMOS circuits where nodes n_0 and n_1 are connected through the resistor R_{sh} which is a resistive short defect.

2.4. Soft-Errors

Although gate-level modeling is the most common approach to study the impact of SETs through simulation, it is difficult to accurately predict their effects [1], as the gate-level description does not represent the actual characteristics of VLSI design [2]. Secondly, it is difficult to capture the complex analog behavior of the propagation of transients at gate level [3] since many internal nodes sensitive to particle radiation are not accessible at gate-level. On the other hand, electrical level simulation is very time-consuming and is not feasible for complex designs. The switch-level is an abstraction level between the gate level and the electrical level and offers many advantages. By operating directly on the transistor network, switch-level simulators can reliably model many important phenomena in MOS circuits, such as bidirectional signal propagation, charge-sharing and variations in driving strengths. Most of the switch-level models used thus far neglect the stray capacitance associated with the transistor nodes [1][4]. In this work, switch-level models are used which take into account bidirectional signal propagation, variations in driving strengths due to different levels of voltages induced by transient pulses, and the effects of node capacitances. This makes these advanced switch-level models more realistic

and closer to actual transistor behavior. Current soft-error modeling methods are based on the assumption that an SET may propagate to the next stage only if the voltage generated by a particle strike at a node is more than $V_{DD}/2$, i.e., if the voltage change at node crosses the logical threshold [5]. Previous research shows that transient pulses which do not have sufficient strength (amplitude) to alter the logic state of the node can cause another effect called a soft delay. A soft delay introduces inherent delays in the combinational logic by changing the driving strength of the signal and can ultimately result in the latching of wrong data at the output [6]. Thus, voltage pulses of smaller amplitude as compared to logic threshold cannot be neglected. This research shows that some of the voltage pulses less than logic threshold can propagate through different transistor stages depending on the location of the struck node, input values and the charge strength of the node capacitance, and can finally cause a logic change at the output. The amplitude of the voltage pulse (indicating the amount of voltage generated by the injected charge due to particle strike) is modeled using driving strengths based on a 5-bit coding system. Using this approach in advanced switch-level models, the effects of the transient injection location on the soft-error rate, the accuracy of the results, and the speed of simulation are analyzed. This leads to the conclusion that the best results in terms of soft-error rate, accuracy and speed of simulation will be achieved if strength-based soft-error detection is used.

2.5. Soft-Error Faults

Radiation-induced soft-errors are an increasingly important reliability issue in integrated circuit technologies. A bit error is called a soft-error if the data is corrupted but the device itself is not damaged. In contrast, a permanent device failure is called a hard error. While the number

of bits that are sensitive to soft-errors tends to grow, the soft-error probability per bit stays constant or increases. Therefore, the trend at system level is for an increase in the soft-error rate (SER) [8]. The development of advanced methodologies to characterize and improve the SER of nanometre technologies is, therefore, essential to assure product reliability.

Soft-errors due to alpha particle radiation are common in integrated circuits, particularly in latches and memory elements. This source is not limited to cosmic rays. On-chip solder bumps produce alpha particles as they contain lead. Hence, storage nodes are subjected to more probable single-event upsets (soft-errors) due to alpha particles than in the past. This may cause the storage node to flip and corrupt its contents. These charged particles can come directly from radioactive materials and cosmic rays or indirectly as a result of high-energy particle interaction with the semiconductor. As the result, a pulse of current with a usual duration of 5-500 ps may charge or discharge a circuit node. The collected charge may be strong enough to alter the data state of a node [12] as well as the node strength. If the node is driven, as in the case of static CMOS, the node may recover quickly. If it is a domino node, a register, latch, SRAM, CRAM or any other type of memory cell, the wrong value may persist until the node is written again.

Today's deep sub-micron devices are already very susceptible to errors induced by neutrons and alpha particles. Shrinking geometries are making the problem increasingly worse with each new generation of technology. Previous generations of 5 V CMOS technology had noise margins of a couple of volts, while newer nanometre technologies have only a few tenths of a volt noise margin [13][14]. Since soft-errors may occur at any time, the conventional post-manufacturing test approach is not useful for measuring SER. Characterization testing and on-line testing are employed instead [15]. Under the non-accelerated characterization test methodology, the test

vectors are either applied to a single IC for billions of device-hours (impractical to implement), or to many such devices for a comparatively shorter period, which could incur a significantly high cost. An alternative is to irradiate the device to increase the soft-error probability followed by measuring the accelerated soft-error rate (ASER) [16]. However, the SER-ASER conversion is inaccurate [17] and poorly understood for combinational logic. Acceleration by lowering supply voltage is also reported [18]. Soft-error causes can be classified in three major groups as follows:

2.5.1. Alpha Particles

The alpha particles are emitted by traces of radioactive elements (such as thorium and uranium) present in the packaging materials of the device. These alpha particles manage to penetrate the die and generate a high density of holes and electrons in its substrate, which creates an imbalance in the device's electrical potential distribution that causes stored data to be corrupted.

The alpha particles emitted by the device package can have energies of up to 8 MeV. It takes about 3.6 eV to generate an electron-hole pair in the substrate, so an 8 MeV alpha particle can generate 2.5 million electron-hole pairs within 2-3 microns of the alpha particle track.

The potential well of a memory cell that contains a “0” is filled with electrons (inversion mode), while that of a memory cell that contains a “1” is devoid of electrons (depletion mode). When an alpha particle hits the substrate and generates holes and electrons, the holes will be pulled toward the substrate supply while the electrons will be pulled towards the potential well.

An empty well can fill up with enough electrons (assuming that enough electron-hole pairs were generated by the alpha particle) to have its stored information reversed from”1” to “0”. Cells that already have electron-filled wells in the first place are not affected by alpha particles.

2.5.2. Cosmic Rays

Once the electronics industry determined how to control package contaminants, it became clear that other causes were also at work. James F. Ziegler led a program of work at IBM which culminated in the publication of a number of papers [19] demonstrating that cosmic rays also could cause soft-errors. Indeed, in modern devices, cosmic rays may be the predominant cause. Although the primary particle of the cosmic ray does not generally reach the earth's surface, it creates a shower of energetic secondary particles. At the earth's surface, approximately 95% of the particles capable of causing soft-errors are energetic neutrons with the remainder composed of protons and pions [20]. This flux of energetic neutrons is typically referred to as "cosmic rays" in soft-error literature. Neutrons are uncharged and cannot disturb a circuit on their own but undergo neutron capture by the nucleus of an atom in a chip. This process may result in the production of charged secondaries, such as alpha particles and oxygen nuclei, which can then cause soft-errors.

2.5.3. Thermal Neutrons

Neutrons that have lost kinetic energy until they are in thermal equilibrium with their surroundings are an important cause of soft-errors for some circuits. At low energies, many neutron capture reactions become much more probable and result in fission of certain materials

creating charged secondaries as fission-byproducts. For some circuits, the capture of a thermal neutron by the nucleus of the B-10 isotope of boron is particularly important. This nuclear reaction is an efficient producer of an alpha particle, Li-7 nucleus and gamma ray. Either of the charged particles (alpha or Li-7) may cause a soft-error if produced in very close proximity, approximately 5 micrometers, to a critical circuit node. The capture cross section for B-11 is 6 orders of magnitude smaller and does not contribute to soft-errors [21].

Boron has been used in Borophosphosilicate glass (BPSG), the insulator in the interconnection layers of integrated circuits, particularly in the lowest one. The inclusion of boron lowers the melt temperature of the glass providing better reflow and planarization characteristics. In this application, the glass is formulated with a boron content of 4% to 5% by weight. Naturally occurring boron is 20% B-10 with the remainder the B-11 isotope. Soft-errors are caused by the high level of B-10 in this critical lower layer of some older integrated circuit processes. Boron-11, used at low concentrations as a p-type dopant, does not contribute to soft-errors. Integrated circuit manufacturers eliminated borated dielectrics by the 150nm process node, largely due to this problem.

This thesis presents a novel approach for increasing the accuracy of soft-errors detection by switch-level analysis of a circuit for Single Event Upset (SEU). Unlike existing methods that focus on bit flipping, this method proves that current charges resulting from radiation can affect the driving strength and eventually lead to data corruption in the switch-level, referred to here as “strength violation”. The relation between logic and strength propagation and its eventual effect on the circuit output voltage is determined by performing a switch-level analysis on the circuit. An advanced coding system is applied that is able to inject the fault in switch-level and detect

soft-errors even if the impulse duration time is relatively small and not able to be detected with most conventional methods.

As explained above, this chapter was discussing the latest related research studies and their solutions for Resistive Delay Faults and Soft-Errors. Most of today's advanced delay fault algorithms are able to propagate those delay faults which create logic or glitch faults. Next chapter will offer an accurate model to propagate delay faults that are not causing logic failure.

CHAPTER 3
RESISTIVE SHORT AND OPEN DELAY FAULT

In advanced technologies, an increasing proportion of defects manifest themselves as small delay faults. Most of today's advanced delay fault algorithms are able to propagate those delay faults which create logic or glitch faults. An algorithm is proposed for circuit fault diagnosis in deep sub-micron technology to propagate the actual timing faults as well as those delay faults that eventually create logic faults to the primary outputs. Unlike the backtrack algorithm that predicts the fault site by tracing the syndrome at a faulty output back into the circuit, this approach propagates the fault from the fault site by mapping a nine-valued voltage model on top of a five-valued voltage model. In such a forward approach, accuracy is greatly increased since all composite syndromes at all faulty outputs are considered simultaneously. As a result, the proposed approach is applicable even when the delay size is relatively small. Experimental results show that the number of fault candidates produced by this approach is considerable.

3.1. Dynamic Effect of Resistive Faults on Output Voltage

It is assumed that existing bridging fault models [22][23] describe shorts between logical nodes with a short resistance of 0 Ω. Many studies regarding the delay defect synthesis have been conducted in gate–level fault modeling [24] and fault diagnosis [25]. Generally, the voltage degradation caused by resistive physical defects is accounted for as intermediate node voltage [26]. There has been little focus on the area of propagation of the additional delay to the circuit output. This section discusses how to calculate the voltage changes at gate output that cause a delay rather than a logic fault. Later in this thesis this delay will be propagated to the output.

3.1.1. Switch-Level Fault Modeling

The switch-level delay fault described in [27], involves the simulations based on fixed capacitive load. For a precise analysis, parasitic resulting from the MOSFET are included for delay estimation due to resistive faults [28]. For 32nm technology, the NMOS and PMOS on-resistance values are calculated using equation (2.1) and equation (2.2). These formulas are explained in [28].

$$R_{ON} = R_N = \frac{V_{DD}}{\frac{K'_N \times W}{2L}(V_{DD} - V_{THN})^2} \qquad (2.1)$$

$$R_{ON} = R_P = \frac{V_{DD}}{\frac{K'_P \times W}{2L}(V_{DD} - V_{THP})^2} \qquad (2.2)$$

Figure 3 shows the transistor-level circuit for NOR gate. A MOSFET is modeled as a 3-terminal device that acts as a voltage controlled resistance, R_{ON} as shown in Figure 4 [29].

Some of variables and definitions will be used in this chapter are brought as follows:

t_P	Total propagation delay
t_{PHL}	High-to-low propagation time
t_{PLH}	Low-to-high propagation time
R'_N	Equivalent switching resistance in the NMOS network
R'_P	Equivalent switching resistance in the PMOS network
R'	Equivalent circuit resistance including resistive elements from NMOS and PMOS sides respectively
n_P	Number of active or ON PMOS
n_N	Number of active or ON NMOS
N	$n_P + n_N$
C_{LOAD}	Load capacitance

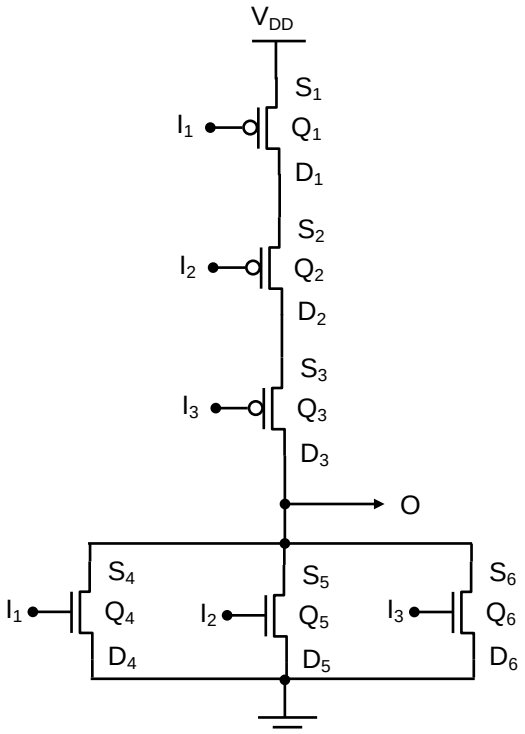


Figure 3 - Transistor level circuit for 3-input NOR Gate.

At the gate-level, faults can only be injected or diagnosed on input and output pins. Variable quantities as follows: n (number of gate inputs), defect resistance (resistive short value (R_{SH}) and resistive open value (R_O)), load capacitance, input combinations and the fault locations.

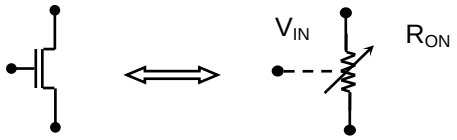


Figure 4 - PMOS and NMOS transistors as voltage controlled resistance.

Figure 5 illustrates the switch-level structure for the 3-input NOR gate. Equation (2.2) shows the relation of R_{ON} with transistor aspect ratio (W/L), operating voltage (V_{DD}) and transconductance (K') [29][30]. Basic concepts of electric circuit analysis are applied for obtaining circuit timing. According to the structure of any CMOS level logic gate, equivalent digital models can be derived by solving the parameters resulting from their serial/parallel connections of PMOS/NMOS or NMOS/PMOS transistor in a gate circuit [30].

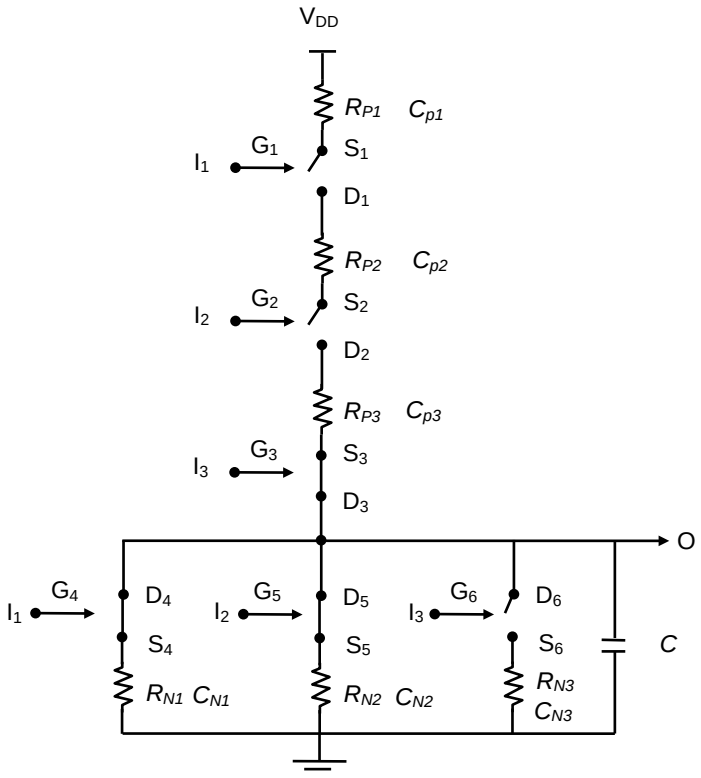


Figure 5 - Switch-level representation for 3-input NOR Gate.

Referring to Figure 5, the transistors on the PMOS side of the circuit are represented by their ON-resistances, R_{Pi} (For $1 \leq i \leq n$). For instance, PMOS1 transistor is R_{P1} and NMOS1 is R_{N1}. The PMOS transistor capacitances are symbolized as C_{Pi} and for NMOS as C_{Ni} ($1 \leq i \leq n$). Further conventions for transistor resistances are done in the same manner. The load capacitance value for a MOSFET can be calculated by combining the oxide and junction capacitance values for each transistor in a gate circuit.

3.1.2. Propagation Delay and Output Voltage

Delay estimation for resistive faults in a 3-input NOR gate is presented in the following subsections. For the all high input cases (for example, I1I2I3 = 111), NMOS transistors are on and PMOS are off. The NMOS side of any gate consists either of a parallel connection of NMOS or a serial connection of NMOS. For n number of NMOS transistors in a logic gate, the high-to-low propagation delay-time for this gate is shown in equation (2.3).

$$\frac{1}{R_N^{'}} = \frac{0.7\ C_{LOAD}}{t_{PHL}} \times \frac{1}{n} \qquad (2.3)$$

Substituting the value for R'N, from equation (2.3) in equation (2.1) is shown in equation (2.4).

$$\frac{\frac{K_N^{'}W}{2L}(V_{OUT} - V_{THN})^2}{V_{OUT}} = \frac{0.7n\ C_{LOAD}}{t_{PHL}}$$

$$\frac{(V_{OUT} - V_{THN})^2}{V_{OUT}} = \frac{0.7n\ C_{LOAD} \times 2L}{t_{PHL} \times\ K_N^{'} \times W}$$

$$V_{OUT} - 2V_{THN} + \frac{V_{THN}^2}{V_{OUT}} = nX_N \qquad (2.4)$$

Where:

$$X_N = \frac{0.7C_{LOAD} \times 2L}{t_{PHL} \times K_N^{'} \times W} \qquad (2.5)$$

Therefore, for a serial NMOS combination

$$V_{OUT} + \frac{V_{THN}^2}{V_{OUT}} = nX_N + 2V_{THN} \qquad (2.6)$$

And for a parallel NMOS connection,

$$V_{OUT} + \frac{V_{THN}^2}{V_{OUT}} = \frac{X_N}{n} + 2V_{THN} \qquad (2.7)$$

Rearranging equation (2.6) to fit the quadratic relation format can have two roots, as shown below,

$$V_{OUT}^2 - (nX_N + 2V_{THN}) \times V_{OUT} + V_{THN}^2 = 0 \qquad (2.8)$$

Reorganizing equation (2.7) makes it a quadratic equation format equation (2.9).

$$V_{OUT}^2 - \left(\frac{X_N}{n} + 2V_{THN}\right) \times V_{OUT} + V_{THN}^2 = 0 \qquad (2.9)$$

Replacing the actual values of X_N, n and V_{THN} into the equations (2.8) and (2.9), the voltage level at the output (V_{OUT}) of a NOR gate is obtained. On the contrary, the propagation delay value for all low input cases will be completely influenced by the PMOS transistors, unlike the all high input case. For the all low input case (for example, $I_1I_2I_3$ = 000), PMOS transistors are on and NMOS are off. The low-to-high propagation delay-time for this gate in serial/parallel PMOS are:

Serial PMOS connection:

$$t_{PLH} = 0.7R_P^{'} \times n \times C_{LOAD} \qquad (2.10)$$

Parallel PMOS connection:

$$t_{PLH} = 0.7(R_P^{'}/n) \times C_{LOAD} \qquad (2.11)$$

A mixed input case (for example, $I_1I_2I_3$ = 001,010,100,110,101 etc.) is a combination of NMOS and PMOS transistors that are on and the rest are off. For this particular category, the numbers of "on" PMOS and "on" NMOS transistors affect the combined delay at the gate output. A calculation of propagation delay from MOSFETs activities both on PMOS and NMOS sides is presented. The variable n has different values for PMOS and NMOS in this case. Identical numbers of active PMOS are termed as n_P and identical number of active NMOS as n_N. For instance, when a primitive gate consists of serially linked PMOS and a parallel formation of NMOS the total propagation delay at the output of that gate can be represented as:

$$t_P = 0.7R^{'} \times C_{LOAD}$$

$$t_P = 0.7(n_P R_P^{'} + R_N^{'}/n_N) \times C_{LOAD} \qquad (2.12)$$

Alternatively, a parallel PMOS connection with a serial NMOS connection is regarded in the following equation:

$$t_P = 0.7R^{'} \times C_{LOAD}$$

$$t_P = 0.7(n_N R_N^{'} + R_P^{'}/n_P) \times C_{LOAD} \qquad (2.13)$$

Output voltage value for all low input and mixed input cases can also be derived in a similar manner.

Actual or static bridging faults have a nominal resistance value mainly in the range of 0 to 500Ω. This problem has been previously analyzed by choosing a fixed resistance value or by applying a locally exhaustive test set at the bridge location [31]. Transistor-level bridging faults can occur internally (intra-gate) or externally (inter-grate). This thesis focuses on the intra-gate bridging faults, which occur inside the transistor level circuit of a gate.

3.2. A Quick Review of Fault Propagation in Transistor-Level

In previous research, [32] a novel fault synthesis algorithm for modelling CMOS circuits with an arithmetic solution for circuit verification and fault synthesis was introduced. This new approach is capable of simulating multiple fault injections into the circuit and speeds up switch-level simulation. Another advantage of this algorithm is its application in the mapping of single and multiple faults from switch-level to gate-level as well as its function as a multi-level model. A unique method to propagate delayed signals that are created by resistive short and open faults

is presented. This method converts a circuit to a graph, finds its arithmetic equation, and eventually, synthesizes faults and generates the outputs

Table 1 - CMOS behaviour lookup table, Multi-Valued Logic 5.

Gate	Source	Drain P (G, S)	Drain N (G, S)
L	L	L	Z
L	H	H	Z
L	Z	Z	Z
L	1	1	Z
L	0	0	Z
L	U	U	Z
H	L	Z	L
H	H	Z	H
H	Z	Z	Z
H	1	Z	1
H	0	Z	0
H	U	Z	U
Z	L	U	U
Z	H	U	U
Z	Z	U	U
Z	1	U	U
Z	0	U	U
Z	U	U	U
1	L	Z	L
1	H	Z	H
1	Z	Z	Z
1	1	Z	1
1	0	Z	0
1	U	Z	U
0	L	L	Z
0	H	H	Z
0	Z	Z	Z
0	1	1	Z
0	0	0	Z
0	U	U	Z
U	L	U	U
U	H	U	U
U	Z	U	U
U	1	U	U
U	0	U	U
U	U	U	U

In CMOS technology, [33] the basic components at switch-level are transistors. N-channel and P-channel transistors (Figure 5) can receive different logical values on their pins. These

logical values are “L”, “H”, “1”, “0”, “Z”, and “U” [8][34][35]. Logic value “L” is weak 0, “H” is weak 1 and “1” or Forcing 1 represents power source. “0” or Forcing 0 represents ground; “Z” represents the state of an isolated or floating connector and may be interpreted as the high impedance. And, finally, “U” represents an intermediate voltage level occurring when “0” and “1” signals are applied simultaneously to a connector and may be interpreted as an unknown signal. Table 1 displays all the possible states of a transistor in a digital circuit in a five-valued voltage model. Later, this look-up table will be used to find the output of the functions.

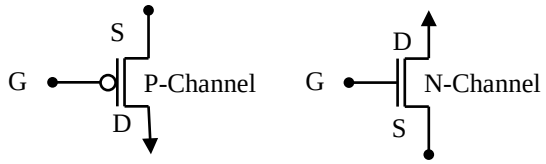


Figure 5 - N-channel and P-channel transistors.

3.2.1. Connection Node

A connection node, shown in Figure 7, is where two or more signals meet each other [36][37] and generate a network. In Figure 7, for example, the drain of a P-channel transistor is connected to the drain of an N-channel transistor each with its own logical value. The outcome is a dominant value for this connection. For instance, if the drain of the P-channel transistor has the value “1” and the drain of the N-channel transistor has the value “Z”, then the dominant value “1” is considered the connection node, which is symbolized as “∇“. Table 2 shows the dominant logic value for connection nodes in a five-valued voltage model.

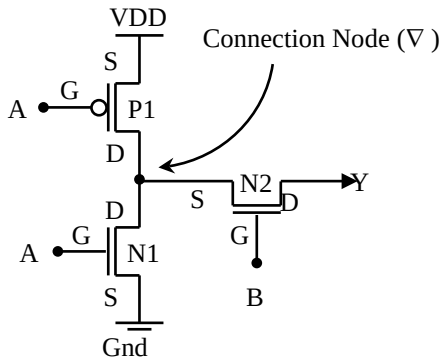


Figure 7 - Connection node.

Table 2 - Connection node equivalent value lookup table, Multi-Valued Logic 5.

∇	L	H	Z	1	0
L	L	U	L	1	0
H	U	H	H	1	0
Z	L	H	Z	1	0
1	1	1	1	1	U
0	0	0	0	U	0

3.2.2. Circuit Graph Model

With an increase in circuit size, it becomes increasingly difficult to analyse and observe circuit behaviour particularly in the presence of a fault [38][39][40][41][42]. In the algorithm presented here, a graph model converts a transistor level model to a graph and is independent of circuit complexity. This graph model simplifies the use of the algorithm and fault injection [32][43][44]. In the graph model, nodes represent a transistor or a connection and the edges represent wires. Also, transistors are not considered as switches, but they are considered as functions with input parameters and a returning value. This feature makes it possible to replace the whole circuit with an arithmetic function. 0 (a) shows a NOT gate circuit with (b) its related

graph. In this graph “P” represents a P-type transistor, “N” represents an N-type transistor, and “∇“shows a connection node. “A” is the primary input signal, and “Y” is the primary output signal, “1” represents Power source and “0” represents ground. “P” and “N” are transistors with “G”, “S” and “D” edges that represent gate, source and drain, respectively, for transistors. The arithmetic equation related to the graph and its functionality is described in detail in [31].

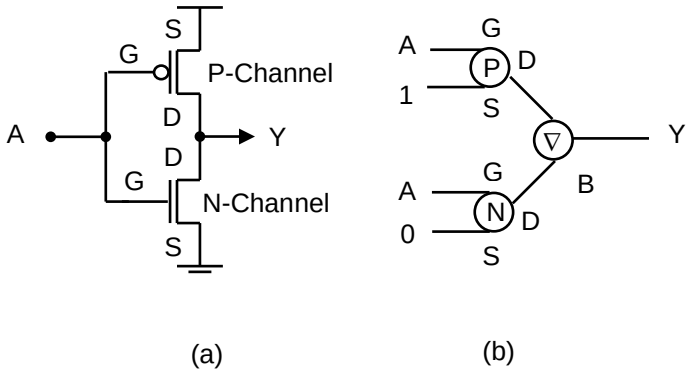


Figure 8 - (a) NOT gate circuit with (b) its related graph.

3.3. Delay Propagation of Resistive Faults in Deep Sub-Microns

Many multiple delay fault based diagnosis methods have been published [45][46][47]. Dastidar and Touba [45] proposed an approach for multiple delay-fault diagnosis based on static timing information. Authors in [46] investigated the effectiveness of n-detection tests to diagnose failure responses caused by multiple stuck-at and bridging faults. The approach presented here is capable of diagnosing multiple delay faults as well as static faults in switch-level and is more accurate and applicable even with a relatively small delay size.

The presented approach uses two models of IEEE Standard 1164-1993: Multi-valued Logic Systems MVL5 and MVL9. Although it is extremely difficult to propagate any fault in a digital circuit when there are no logical changes in signals, this method is able to propagate a delay in MVL5 using a MVL9. Most other algorithms are able to propagate a timing disturbance as soon as there is a logical failure in circuit functionality. However, if there are no logical failures, these methods are useless. To solve this problem, a MVL9 is used on top of a MVL5.

Table 3 - Mapping MVL9 and MVL5.

Nine-valued voltage model	Five-valued voltage model
1	1
H	
X	
X	H
U	U
W	L
W	0
L	
0	

The voltage level in MVL5 has been divided into several levels of voltage based on Verilog strengths for upper and lower boundaries for each logical section in MVL5. In this method logic value ‘‘1” in MVL5 can be mapped to (1 and H) in MVL9 if the signal is strong 1 as per calculation in Section 2.1, or mapped to X if the value is considered to be weak 1. On the other hand, the value of ‘‘0” in MVL5 can be mapped to (0 and L) in MVL9 if the signal is strong 0 as per calculation in Section 2.1 , or W if the value is considered to be weak 0. Table 3 maps logic values from MVL9 to MVL5 and vice versa. As shown in this table logic 0 and 1 in MVL5 are

mapped to three different logics in MVL9 based on the equivalent voltage coverage. The upper boundary of W and lower boundary of X in MVL9 are mapped to L and H in MVL5 accordingly.

Consequently, the logic value ‘‘H” in MVL5 will be mapped to ‘‘X” for all medium (1) and small (1)1 signals in MVL9 as per Section 2, and the logic value ‘‘L” in MVL5 will be mapped to ‘‘L” for all medium (0) and small (0)1 signals in MVL9 as per Section 2. Finally, U or un-initialized signal will be considered the same in both models. As explained in Section 2, in the case of a slow-to-rise or slow-to-fall delay due to a resistive open or short fault in circuit the output voltage of the gate will be affected accordingly. This change is not sufficiently significant to affect a logic change in MVL5 but may change the logic in MVL9. As explained above, this model maps logic values from MVL9 to MVL5 and vice versa.

3.3.1. Arithmetic Equation of the Circuit

The algorithm uses the behaviour of CMOS transistors in digital circuits [48] and describes the circuit in an arithmetic equation. An equation is defined and describes circuit behaviour in detail according to all possible input combinations. In this arithmetic model, each transistor is considered a function such as “P” and “N”. Each function has two arguments as inputs and a returning logical value that is considered for Drain. P (G, S) and N (G, S) are the syntaxes for the functions. The “P” function is used for P-type transistors, and the “N” function is used for N-type transistors. The first argument or “G” is the value of Gate, and the second argument or “S” is the

value of Source for each transistor. The result of the function will be calculated with the logic value at the drain. The value of each function can be derived from a lookup Table 4 in MVL9.

Table 4 - CMOS behaviour lookup table, Multi-Valued Logic 9.

Gate	Source	Drain P (G, S)	Drain N (G, S)	Gate	Source	Drain P (G, S)	Drain N (G, S)
L	L	W	Z	0	L	L	Z
L	H	X	Z	0	H	H	Z
L	Z	Z	Z	0	Z	Z	Z
L	1	1	Z	0	1	1	Z
L	0	L	Z	0	0	0	Z
L	U	U	Z	0	U	U	Z
L	X	X	Z	0	X	X	Z
L	W	W	Z	0	W	W	Z
H	L	Z	W	U	L	U	U
H	H	Z	X	U	H	U	U
H	Z	Z	Z	U	Z	U	U
H	1	Z	H	U	1	U	U
H	0	Z	L	U	0	U	U
H	U	Z	U	U	U	U	U
H	X	Z	X	U	X	U	U
H	W	Z	W	U	W	U	U
Z	L	U	U	X	L	W	W
Z	H	U	U	X	H	W	X
Z	Z	U	U	X	Z	W	Z
Z	1	U	U	X	1	W	H
Z	0	U	U	X	0	W	L
Z	U	U	U	X	U	W	U
Z	X	U	U	X	X	W	X
Z	W	U	U	X	W	W	W
1	L	Z	L	W	L	W	X
1	H	Z	H	W	H	X	X
1	Z	Z	Z	W	Z	Z	X
1	1	Z	1	W	1	H	X
1	0	Z	0	W	0	L	X
1	U	Z	U	W	U	U	X
1	X	Z	X	W	X	X	X
1	W	Z	W	W	W	W	X

For example, given the logical values G = L and S = 1, according to Table 4, these functions will be P (L, 1) = 1 and N (L, 1) = Z. Equation (2.14) is an arithmetic evaluation for the graph in Figure 8 (b).

$$Y = P(G, S) \nabla N(G, S) \qquad (2.14)$$

In Figure 8 (b), the drain of a P-channel transistor is connected to the drain of an N-channel transistor, each with its own logical value. The outcome is a dominant value for this connection. For instance, if the drain of the P-channel transistor has the value “1” and the drain of N-channel transistor has the value “Z”, then the dominant value “1” is considered for this node and symbolized as “∇“. Table 5 shows the dominant logic value for the connection node in MVL9.

Table 5 - Connection node equivalent value lookup table, Multi-Valued Logic 9.

∇	U	X	0	1	Z	W	L	H	-
U	U	U	U	U	U	U	U	U	U
X	U	X	X	X	X	X	X	X	X
0	U	X	0	X	0	0	0	0	X
1	U	X	X	1	1	1	1	1	X
Z	U	X	0	1	Z	W	L	H	X
W	U	X	0	1	W	W	W	W	X
L	U	X	0	1	L	W	L	W	X
H	U	X	0	1	H	W	W	H	X
-	U	X	X	X	X	X	X	X	X

3.3.2. Delay Propagation

The actual delay, which is the result of a resistive short or open fault in digital circuit, will change the voltage that has to pass through a path to the output. To illustrate how this algorithm works, several examples are presented. Figure 9 shows two three-input NOR gates connected to each other. Assume there is a delay at output of “NOR” gate “L” which is caused by a 5 KΩ resistor between drain D_3 and V_{DD}. According to the result that will be discussed later in the simulation result chapter, for 65nm technology, the output voltage of slow-to-rise is 0.85 V for a delay of 0.19 μs.

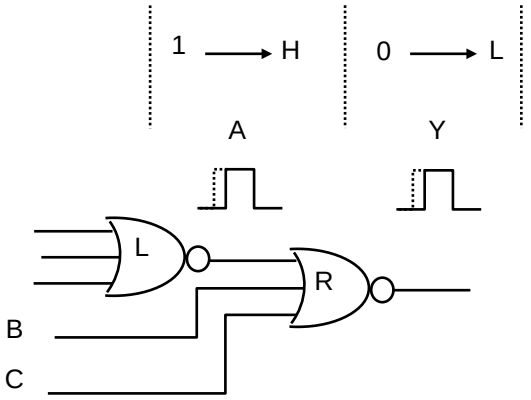


Figure 9 - Logic change in MVL9.

This value is considered as “H” signal in MVL9 instead of “1”. This signal must pass through a three-input “NOR” gate using the circuit’s arithmetic equation. Figure 10 (a) shows the transistor level of a three-input NOR gate and (b) shows the arithmetic model of the circuit. As explained earlier, the output of the circuit “Y” can be calculated by equation (2.15) as follows:

$$Y = P3\,(G, P2\,(G, P1\,(G, S))) \nabla (N1\,(G, S) \nabla \; N2\,(G, S) \nabla N3\,(G, S)) \qquad (2.15)$$

Equation (2.16) represents equation (2.15) after replacing actual signals for gate “G” and source “S”.

$$Y = P3\,(C, P2\,(B, P1\,(A, 1))) \nabla (N1\,(A, 0) \nabla \; N2\,(B, 0) \nabla N3\,(C, 0)) \qquad (2.16)$$

The delayed signal “H”, which is caused by a resistive short fault in “NOR” gate “L” is considered as input signal “A” in the above equation.

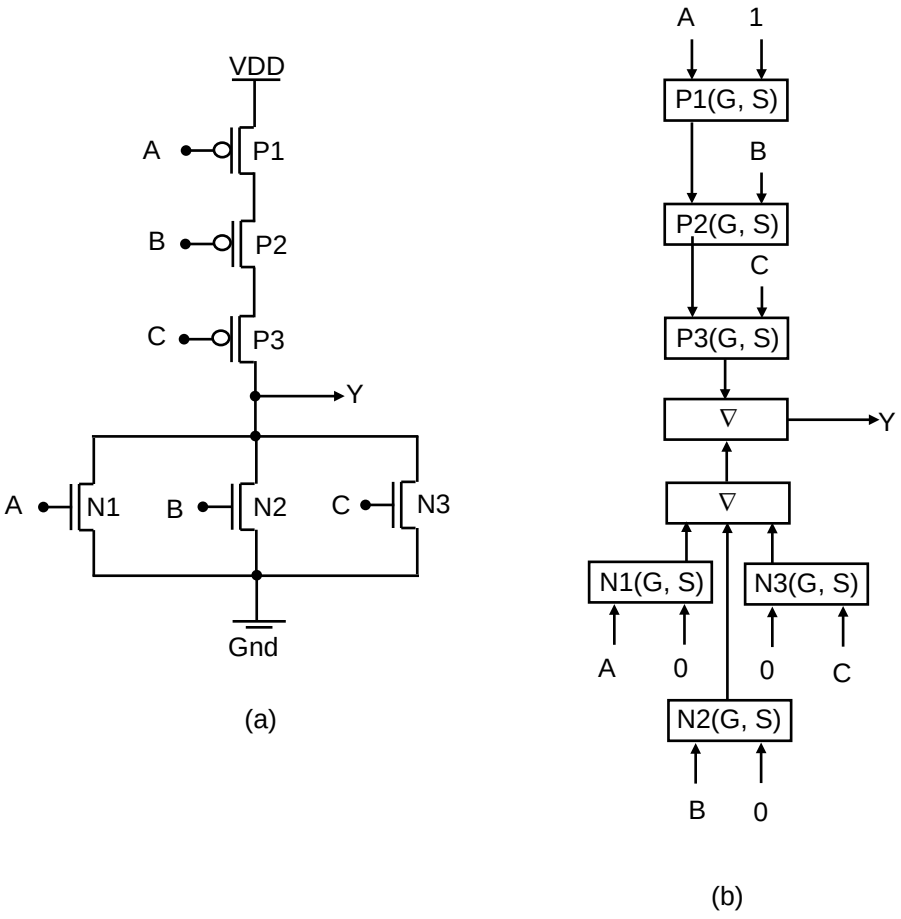


Figure 10 - (a) Transistor level of three-input NOR gate and (b) Arithmetic model.

To propagate this signal to the “Y” output, input “B” and “C” must have the value of “0”. The result is shown in equation (2.17).

$$Y = P3\ (0, P2\ (0, P1\ (H, 1))) \nabla\ (N1\ (H, 0) \nabla\ \ N2\ (0, 0) \nabla\ N3\ (0, 0)) \qquad (2.17)$$

Values of function “P” and “N” can be derived from lookup Table 4. The value of connection node “∇“ is taken from lookup Table 5 and the final value for “Y” will be equal to “L”.

All steps are shown as follows:

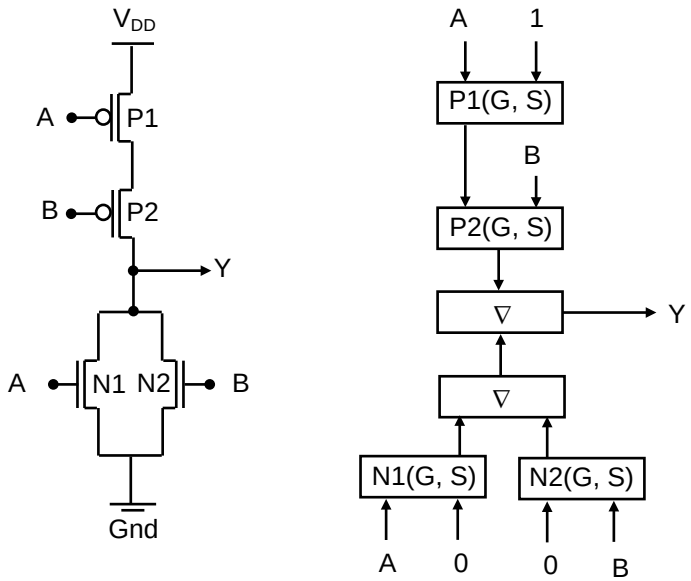
P1 (H, 1) = Z

P2 (0, Z) = Z

N1 (H, 0) ∇ N2 (0, 0) ∇ N3 (0, 0) = L ∇ Z ∇ Z = L

Y = P3 (0, Z) ∇ L = Z ∇ L = L

As shown in Figure 9, when the logic value for signal A changes from “1” to “H” in MVL9 as the result of a resistive fault in the circuit, the logic value for output “Y” changes from “0” to “L”. This change at output represents a delay of 0.37 μs. At the same time, in MVL5 signal “A” and “Y” remains the same value “1” and “0” regardless of presenting a delay fault in the circuit.



(a) (b)

Figure 11 - (a) Transistor level of two-input NOR gate and (b) Arithmetic model.

As mentioned earlier, this arithmetic algorithm can be applied to circuits with multiple delays. These delays may affect each other at the output or one delay may compensate for

another one and vanish through several stages. Sometimes two or more delays may combine and affect a logical failure in MVL5.

A further example is given in Figure 11, where (a) shows the transistor level of a two-input NOR gate and (b) shows the arithmetic model of the circuit. Using the algorithm, the output of the circuit “Y” can be calculated with equation (2.18) as follows:

$$Y = P2\ (G, P1\ (G, S)) \nabla\ (N1\ (G, S) \nabla\ N2\ (G, S)) \qquad (2.18)$$

Equation (2.19) represents equation (2.18) after replacing actual signals for gate “G” and source “S”.

$$Y = P2\ (B, P1\ (A, 1)) \nabla\ (N1\ (A, 0) \nabla\ N2\ (B, 0)) \qquad (2.19)$$

The delayed signal “X” is caused by a resistive short fault considered as input signal “A” in the above equation. To propagate this signal to the “Y” output, input “B” must have the value of “0”. The result is shown in equation (2.20).

$$Y = P2\ (0, P1\ (X, 1)) \nabla\ (N1\ (X, 0) \nabla\ N2\ (0, 0)) \qquad (2.20)$$

Values of function “P” and “N” can be derived from lookup Table 4. The value of connection node “∇“ must be taken from lookup Table 5 and the final value for “Y” will be equal to “L”.

All steps are shown as follows:

P1 (H, 1) = W

N1 (H, 0) ∇ N2 (0, 0) = L ∇ Z = L

Y = P2 (0, W) ∇ L = W ∇ L = W

When the logic value for signal A in MVL9 changes from “1” to “X” as a result of the resistive fault in the circuit, the logic value for output “Y” will change from “0” to “W”. This change at output represents a delay of 0.44 μs. At the same time, in MVL5 model signal “A” and “Y” remains in the same logic value “1” and “0” regardless of presenting a delay fault in the circuit.

The objective of the above chapter was to propagate the resistive delay fault with developing a very accurate model that can convert the timing disturbance to logic defect in five value logic model using a nine value logic model. This model carries the error from a timing domain to a voltage domain. The resistive delay faults can make circuit less immune against external radiations. The effect of radiations on a circuit with resistive bridging defect will be discussed in next chapter.

CHAPTER 4
DELAY FAULT MANIFESTS SOFT-ERRORS

The traditional test model of VLSI design being questioned by increasing delay fault manifestations has become even further challenged as a result of unpredictable soft-errors. Consequent probabilistic fault manifestations shift the focus to fault resilience mechanisms and tradeoffs of false alarms vs. escapes. This chapter shows how delay faults can increase the Soft-Error Rate in the circuit.

4.1. The Importance of Delay Faults and Soft-Errors

The continuous shrinking of VLSI devices and the fast increase in chip clock rates raise the challenging problem of ensuring that designs are meeting performance and reliability specifications. It has been widely observed that chips are increasingly susceptible to delay defects and soft-errors, both more difficult to deal with using a manufacturing test compared to traditional stuck-at faults. These potential reliability problems are becoming increasingly critical due to the aggressive technology scaling and design style. Additional test resources mitigate

these problems only to a limited extent, while incurring a much larger testing cost, a quite unpalatable trade-off from a test economics point of view. The solution to these challenges necessitates the incorporation of reliability-oriented design techniques and economic models that guide their applications in product development.

Although extensive work has been conducted on fault tolerance circuitry design, little research has been proposed to address the issue of concurrent handling of delay defects and soft-errors. The challenge of this problem stems from the need for integrating diversified fault-tolerance schemes in a low cost and coherent manner. For example, these faults may either generate noise at the input of a flip-flop or directly corrupt its internal node signal. Hence an appropriate fault-tolerance scheme needs to simultaneously address distinct fault behaviours. Moreover, such a scheme must differ from traditional schemes in the sense that it should not incur any timing pressure which essentially tampers with the delay fault tolerance capability.

4.2. Soft Delay Phenomenon

When a highly energetic particle strikes at a sensitive node of a semiconductor device, the electron-hole pairs are created in the track of the particle. The electron-hole pairs drift when they are generated in the depletion region. This movement of electrons and holes toward an opposite electric field in the depletion region causes charges to be collected at the opposite sides of the depletion region. A current pulse with duration of a few hundred picoseconds is generated due to the movement of charges. This pulse can have positive or negative magnitude depending on whether the particle hit an off NMOS-transistor or off PMOS-transistor. This current pulse is

represented by an equivalent current source between the drain and the substrate of the transistor [49][50]. In the following context, only the positive current pulses are considered.

The active transition scenario of the soft delay phenomenon is as follows. In CMOS circuits during the rising phase of a transition, the NMOS transistor goes to OFF state and the PMOS goes to ON state. As soon as the NMOS transistor turns off and the p-n junction builds up between the drain and the substrate of the NMOS transistor, it becomes sensitive to a particle hit. If the particle hit on this sensitive node occurs during the signal transition, the current generated due to the hit can pull down the signal to the opposite logic level causing longer transition time. This longer transition of the signal at the node can have a delay effect at the output of the succeeding gate(s). This effect is called soft delay [51] due to its nature.

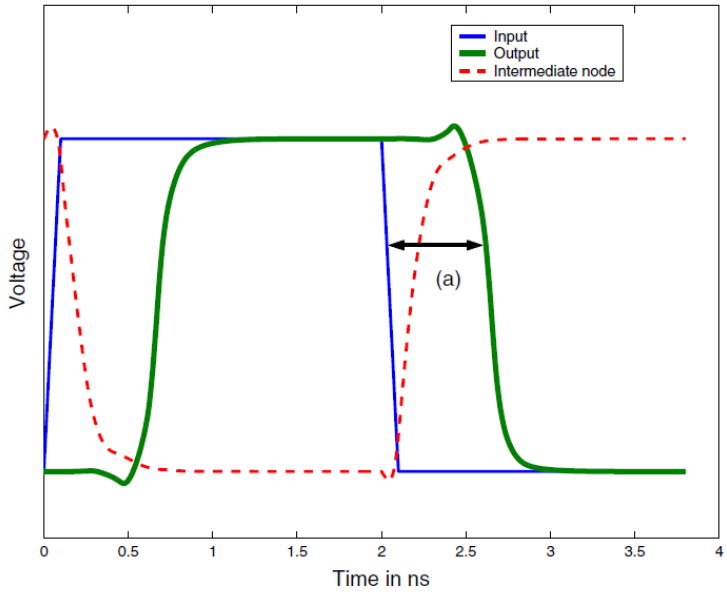


Figure 12 - Propagation delay (a) of a buffer in the normal operation.

Figure 12 shows the normal operation of a buffer between its input (blue line) and output (green line) signal. The intermediate node (red line) is the sensitive node within the buffer. The

arrow marked (a) in Figure 12 shows the propagation delay of 595 ps. between input and output of the buffer.

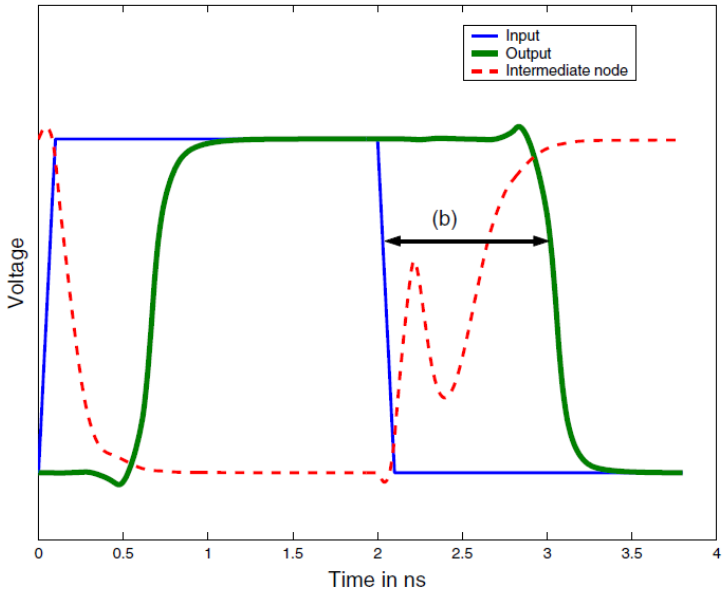


Figure 13 - Propagation delay (b) of a buffer due to the particle strike during the transition.

Figure 13 shows the effect of the particle hit during the transition of the sensitive node which introduces extra propagation delay. This will results in a new delay of 1003 ps. as shown in the arrow marked (b).

To illustrate the active transition scenario in CMOS logic circuits, the circuit in Figure 14 is used as a reference circuit. The gate delays are shown inside the gates. The transistor level diagram of inverter I1 shows an SEU sensitive node. Node, $\bar{A}$, is sensitive to a particle hit as soon as NMOS-transistor of inverter I1 goes off and the reverse junction builds up between the output ($\bar{A}$) and the substrate of NMOS-transistor. The current source, I_{SEU}, represents the particle hit as a positive current pulse at the node $\bar{A}$.

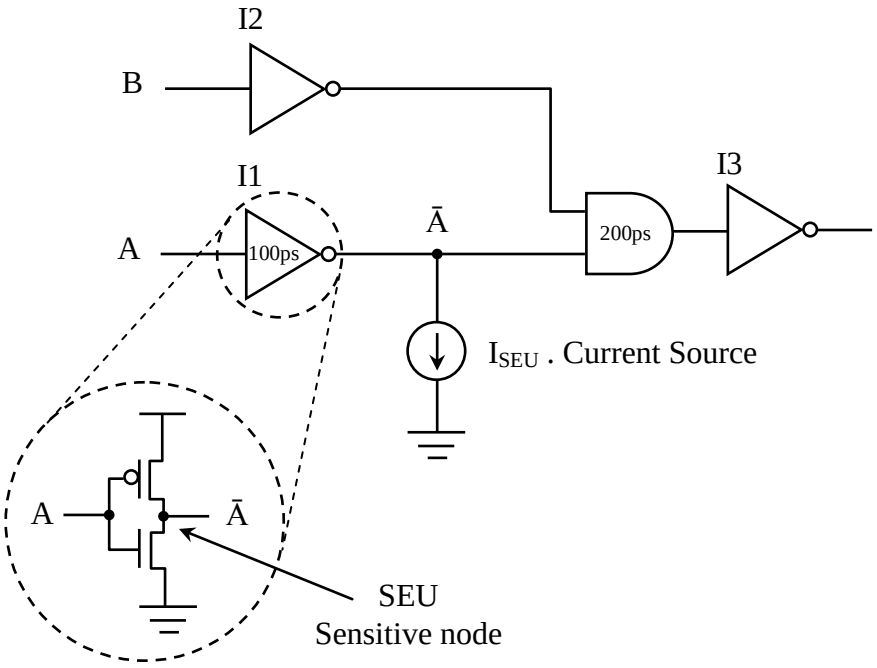


Figure 14 - CMOS circuit to show the soft delay phenomenon.

4.3. Fault Modeling

As mentioned earlier, in CMOS circuits, SEUs are modeled by injecting a current pulse at the sensitive node. This pulse has rapid rise time and gradual fall time. The shape of the pulse can be approximated by the following equation [52].

$$I_{SEU}(t) \propto \frac{Q}{T} . \sqrt{\frac{t}{T}} . exp^{\left(\frac{-t}{T}\right)} \qquad (3.1)$$

Where Q is the charge collected due to the particle strike and T is the time constant for the charge collection process. T depends on the CMOS process technology used for the device and it decreases as technology scales down. The following subsections describe fault models for two soft delay scenarios.

4.4. Active Transition Delay Model

In Figure 14, the inverters I1, I2, and I3 are unit size inverters, and the AND gate was constructed using minimum size transistor with equal rise and fall time delays. The inverter I3 provides capacitive load for the output of AND gate. A current source is used to inject error at node $\bar{A}$ with I_{SEU} of equation (3.1). The pulse is injected during the output rise time of I1, specifically when $A < V_{THN}$ and $\bar{A} > V_{DD}/2$, where V_{THN} is the threshold of NMOS transistor and V_{DD} is the power supply voltage. At this time, the output of I1, $\bar{A}$, is sensitive to particle hit.

4.5. Inactive Transition Delay Model

The current pulse produced by a particle strike results in a logic pulse at the output node of the affected gate. To model the logic fault, the circuit of Figure 14 is used, but this time the fault is injected when gates of the circuit are at steady state. Both of the inputs were assumed to be low so that a current source can be used at the output of the I1 to temporarily flip the node $\bar{A}$. The width of the logic pulse produced at the output of the AND gate is measured at $V_{DD}/2$.

How the delay faults can affect Soft-Error Rate was discussed in detail in this chapter by explaining a fault model and terms of active and inactive transition delay models. Next chapter is introducing a new invention in soft-error detection for transient pulses less than threshold voltage.

CHAPTER 5
STRENGTH VIOLATION EFFECT ON SOFT-ERROR

The SET in the circuit might lead to a current depletion in the circuit that can affect the strength of the signal and eventually origin a Soft-Error in the circuit. The strength violation as the result of SET were not been investigated in previous studies. In this chapter the concept of strength violation will be discussed in detail. A strength-based Soft-Error detection model has been developed which uses several rules and function.

5.1 Strength Violation

The occurrence of radiation strikes are generally distributed uniformly in space and time. To better explain this phenomenon, consider a 2-input NOR gate (Figure 14) driving a parasitic capacitance C_P (interconnect and fan-out) at its output. When all connected transistors to a node are off, the node retains its previous value. This is caused by the charge stored on the stray

capacitance C_T associated with the node (after a long decay time, C_T will be discharged, but usually circuits are operated sufficiently fast so that decay times can be considered infinite) [8]. The total capacitance at output of the gate is given in equation (4.1).

$$C_T = C_U (W/L) + C_P \qquad (4.1)$$

In this equation W/L is the size of a single NMOS transistor in the NOR gate. C_U is the unit output capacitance including NMOS and PMOS obtained by dividing the output capacitance of the NOR gate by the size of the NMOS transistor in the NOR [36][53].

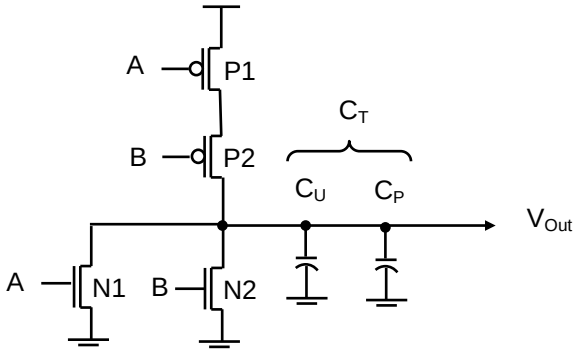


Figure 15 - Stray capacitance illustration in NOR gate.

The magnitude and duration of output voltage VOUT will determine how an SEU propagates through gates in the transitive fan-out of the NOR gate to the primary outputs latches. For an accurate calculation of output voltage, resistive analysis of NMOS and PMOS ON-resistance values are calculated using [27] and are shown in equations (4.2) and (4.3).

$$R_N = V_{DD} / (K_N W (V_{DD} - V_{THN})^2 / 2L) \qquad (4.2)$$

$$R_N = V_{DD} / (K_P W (V_{DD} - V_{THP})^2 / 2L) \qquad (4.3)$$

Where V_{DD} is operating voltage, K_N and K_P are transconductance parameter of a MOSFET, and V_{THN} and V_{THP} are threshold voltages for NMOS and PMOS accordingly. When all inputs for 2-input NOR gate in 0 are high (A = 1 and B = 1), NMOS transistors are on and PMOS are off. The NMOS side of the gate consists of parallel connections of NMOS transistors. In this condition, the high-to-low propagation time will be calculated from equation (4.4) as per [27].

$$1 / R_N = (0.7 C_T / t_{PHL}) * 1/n \quad (4.4)$$

In equation (4.4), t_{PHL} is high-to-low propagation time and n is the number of gate input. Substituting the value of R_N in 4.4 with equation (4.2) and then re-arranging it is shown in equation (4.5).

$$V_{OUT} - 2V_{THN} + \frac{V_{THN}^2}{V_{OUT}} = nX_N \qquad (4.5)$$

Where X_N is:

$$X_N = \frac{0.7C_{LOAD} \times 2L}{t_{PHL} \times K_N' \times W} \qquad (4.6)$$

The voltage V_{OUT} (t) following a particle strike is given in the following differential equations for a serial NMOS combination.

$$V_{OUT} + V^2_{THN} / V_{OUT} = n X_N + 2V_{THN} \qquad (4.7)$$

Similarly, for a parallel NMOS connection,

$$V_{OUT} + V^2_{THN} / V_{OUT} = X_N / n + 2V_{THN} \qquad (4.8)$$

Equation (4.7) must be rearranged to fit the quadratic relation format shown in (4.9).

$$V^2_{OUT} - (nX_N + 2V_{THN})\, V_{OUT} + V^2_{THN} = 0 \qquad (4.9)$$

Reorganizing equation (4.8) will result in equation (4.10).

$$V_{2OUT} - (X_N / n + 2V_{THN})\, V_{OUT} + V_{2THN} = 0 \qquad (4.10)$$

Replacing the actual values of X_N, n and V_{THN} into the equations (4.9) and (4.10), the voltage level at the output V_{OUT} of a NOR gate is obtained. In contrast, the propagation delay value for all low input cases will be completely influenced by the PMOS transistors, unlike the all high input case. For the all low input case (for example, I_1I_2 = 00), PMOS transistors are on and NMOS are off. The I_{DS} [54] for PMOS transistors are calculated using equation (4.11).

$$I_{DS} = (W_{\rho n}\, C_{o\chi})\, [(V_{GS} - V_{THN})\, V_{DS} - V^2_{DS}/2] \qquad (4.11)$$

On the other hand, NMOS transistors are more susceptible to strength violation than PMOS. The drain of a PMOS transistor collects holes. This results in a rise in drain voltage if the transistor is in the OFF-state. However, the drain collection current and, consequently, the collected charge Q_{coll} are in general much smaller than for an NMOS transistor. This is due to the fact that holes have a lower mobility than electrons and also due to charge sharing between the drain junction and the junction formed by the n-well and the p-substrate [55].

The magnitude and duration of a pulse will determine how a SEU propagates through gates in the transitive fan-out of the gate to the primary outputs/latches/flip-flop. The charge deposition due to a particle strike at N (where P and N networks connect) is modeled by a double exponential current pulse I_{in} at the site of the particle strike [36][56].

$$I_{in} = \frac{Q}{(\tau_\alpha - \tau_\beta)}\left\{e^{-\frac{t}{\tau_\alpha}} - e^{-\frac{t}{\tau_\beta}}\right\} \qquad (4.12)$$

In equation (4.12), Q is the charge (positive or negative) deposited as a result of the particle strike, τ_α is the collection time constant of the junction, and τ_β is the ion-track establishment time constant. τ_α and τ_β are constants that depend on several process-related factors. SEU effects are shown in Figure 16. This figure shows that one strike with a constant charge and duration has higher impact when the transistor dimension shrinks.

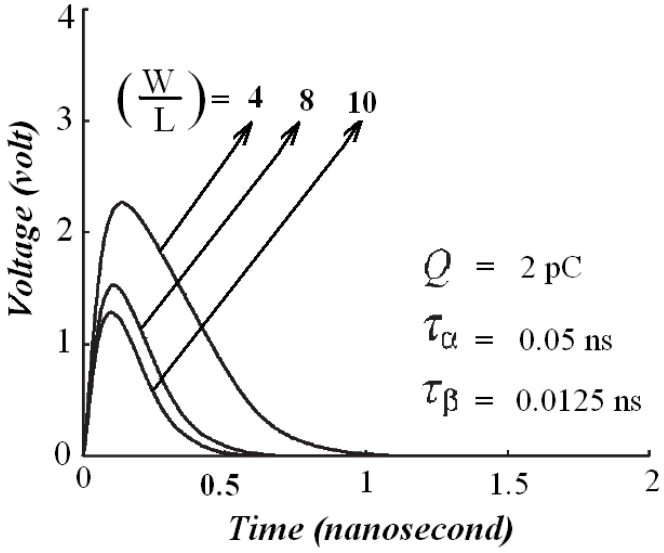


Figure 16 - SEU effects (Q, (W/L), (τ_α, τ_β)).

The following example shows the data corruption as a result of strength violation as previously discussed. Figure 17 illustrates the simulation of a single-event upset using a current source at the drain of P2 transistor which violates the driving strength of the drain for duration of 5 ns to simulate the strength violation. This current course is able to generate different single even upsets in circuit to simulate a strike for soft-errors. In this simulation, both values A and B are equal to one (A = 1 and B = 1), the amplitude of voltage pulse generated due to a particle will induce an electric charge at the point of injection. As a result, the value of unknown X will be stored on flip-flop instead of original value 1. The simulation result of the circuit in Figure 17 is shown in Figure 18 using H-Spice tool.

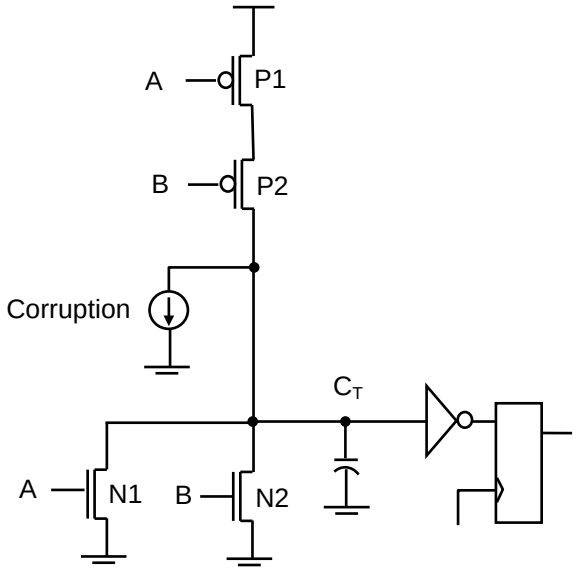


Figure 17 - Strength violation simulation in NOR gate.

The top wave in Figure 18 represents a “Strike Pulse” and the middle wave of the graph represents the driving current stimulation at the drain of PMOS transistor, which is a strength violation. Strength violation is referred to a term that strength of a signal changes without

changing of its logic. This signal shows that the complex analog behaviour of the signal can change the strength within the “High 1” range. The bottom signal in the graph shows an unknown value for the output of NOR gate which is connected to a memory element in the circuit.

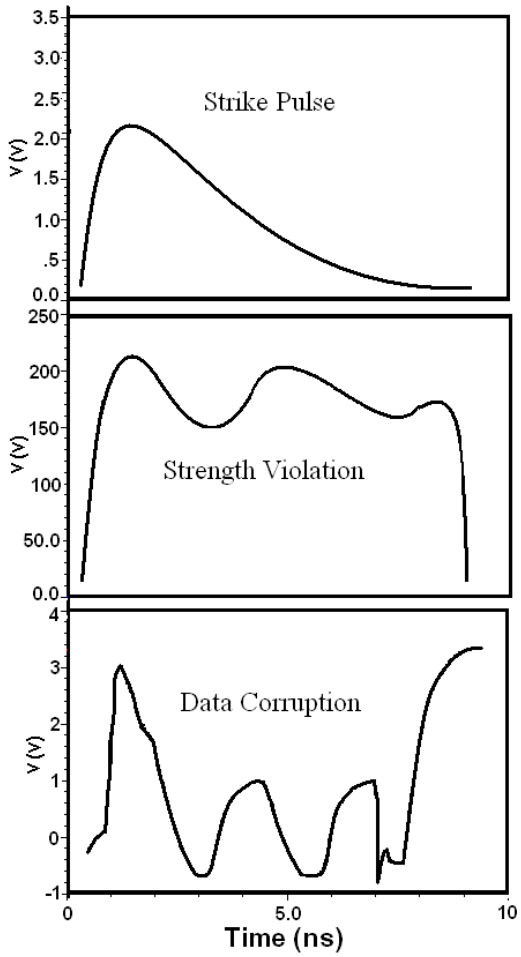


Figure 18 - H-Spice simulation result for two-input NOR gate.

5.2 Soft-Error Detection Coding Model

The switch-level models used in this thesis are based on Verilog switch models and take into consideration seven driving strength levels [35] and four logic values (0,1,U,Z). This multiple-

level, logic-strength modeling resolves combinations of signals into known or unknown values and thus represents the behaviour of hardware with maximum precision. In switch-level models, an NMOS (PMOS) may be in three different states: It is ON when its gate terminal takes the value 1 (0 for PMOS). It is OFF when its gate terminal takes the value 0 (1 for PMOS), and it is in the unknown state when its gate terminal takes the value Z or U [36].

Table 6 - Logic and strength coding for CMOS.

Logic Name	Strength Name	Logic Code	Strength Code	Node N
0	Supply	00	111	00111
0	Strong	00	110	00110
0	Pull	00	101	00101
0	Large	00	100	00100
0	Weak	00	011	00011
0	Medium	00	010	00010
0	Small	00	001	00001
1	Supply	01	111	01111
1	Strong	01	110	01110
1	Pull	01	101	01101
1	Large	01	100	01100
1	Weak	01	011	01011
1	Medium	01	010	01010
1	Small	01	001	01001
U	Supply	11	111	11111
U	Strong	11	110	11110
U	Pull	11	101	11101
U	Large	11	100	11100
U	Weak	11	011	11011
U	Medium	11	010	11010
U	Small	11	001	11001
X	Don’t care	10	000	10000
Z	High Z	xx	000	xx000

As mentioned earlier, voltage strength refers to the amplitude of voltage pulse generated due to particle induced electric charge at the point of injection. Voltage levels are mapped to different

signal strength levels. The highest strength level is called “supply” and is denoted by “111”. When the circuit is working normally (in the absence of any particle strike), all signals are assumed to have strength level “111”. A strike at any point can inject voltage pulse amplitude varying from “strong” to “high impedance”. High impedance is represented by state code “xx000”. The logic states and level of strength based on a 5-bit coding system are shown in Table 6.

In this table, “Logic Names” are “0”, “1”,”U” unknown, “X” don’t care and “Z” for high impedance. Each “Logic” has a different strength from “Supply” to “Small”. “Logic Code” and “Strength Code” are the 2-bit and 3-bit coding representation of “Logic” and “Strength” accordingly. Finally, “Node N” is a 5-bit coding system which includes both the logic and strength value of the node.

Each transistor in a circuit is connected to three nodes. Figure 19 (a) in which “D” denotes the node or net connecting the “Drain” and “G” and “S” are used for “Gate” and “Source” accordingly.

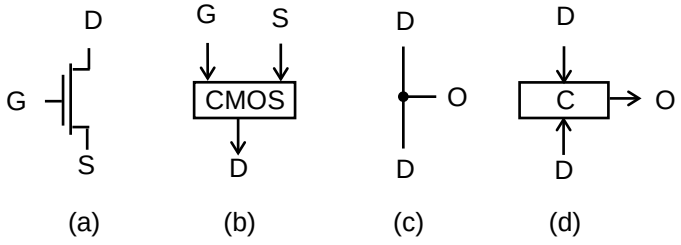


Figure 19 - (a) CMOS transistor (b) CMOS function (c) Node (d) C function.

To calculate the drain output signal based on the gate and source input signals, NMOS and PMOS functions have been developed for N-channel and P-channel transistors. Figure 19 (b) is a symbolic representation of the function.

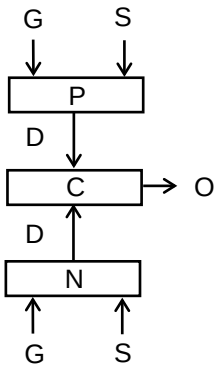


Figure 20 - Functional representation of a NOT gate.

Connection nodes in the circuit are where signals meet each other and are shown in Figure 19 (c). Furthermore, Figure 19 (d) is a symbolic representation of a connection node and Function C, which will be explained in detail in this section. Figure 20 shows a functional representation of a NOT gate. The initial idea is taken from previous publication [57].

5.3 Verilog Strength Rules and Functions

The Verilog HDL provides for accurate modeling of signal contention, bidirectional pass gates, resistive MOS devices, dynamic MOS, charge sharing, and other technology dependent network configurations by allowing scalar net signal values to have a full range of unknown values and different levels of strength or combinations of levels of strength. This multiple level logic strength modeling resolves combinations of signals into known or unknown values to represent the behavior of hardware with maximum accuracy. The switch-level models are based

on the 5-bit coding model shown in Table 6. In these switch-level models, an NMOS or PMOS may be in three different states: It ON when its gate terminal takes the value 1 (0 for PMOS), OFF when its gate terminal takes the value 0 (1 for PMOS), and in the unknown state when its gate terminal takes the value Z or U. The highest strength level is called “supply” and is denoted by “111”. In the absence of any transient pulse, all signals are assumed to have strength level “111”. A strike at any point can inject voltage pulse amplitude varying from “strong” to “Z”. Based on the 5-bit coding system, the functionality of NMOS and PMOS switches can be summarized in the following rules:

Rule 1: When a switch is OFF, it passes state code “10000” (don’t care) to the drain, which indicates that the transistor does not take part in resolving the state of the output at the connection node.

Rule 2: When a switch is ON, it passes the state code of the source to the drain.

Rule 3: When the gate has unknown logic or high impedance state, the drain gets unknown state.

Based on these three rules, switch-level functions named PMOS Function and NMOS Function are developed to represent PMOS and NMOS switches respectively.

To resolve the signals of different strength levels and logic levels at the connection nodes (meeting points of two or more switches), another function called C Function, based on Verilog strength resolution rules, is developed. When all switches connected to a node are OFF, the node retains its previous value caused by the charge stored on the stray capacitance associated with the node; otherwise, the signal with larger strength level passes through. If the signals at the node are of the same strength level but different logic level, the output state of the node is unknown.

5.3.1 PMOS Function

Figure 21 shows the flow diagram for PMOS Function representing PMOS switch. “Drain Logic” represents the logic level of signal at drain while “Drain Strength” represents the strength level for signal at drain of PMOS switch. Similarly, “Gate Logic” and “Gate Strength” signify the logic level and strength level for signal at gate and “Source Logic” and “Source Strength” represent logic level and strength level of signal at source of PMOS switch.

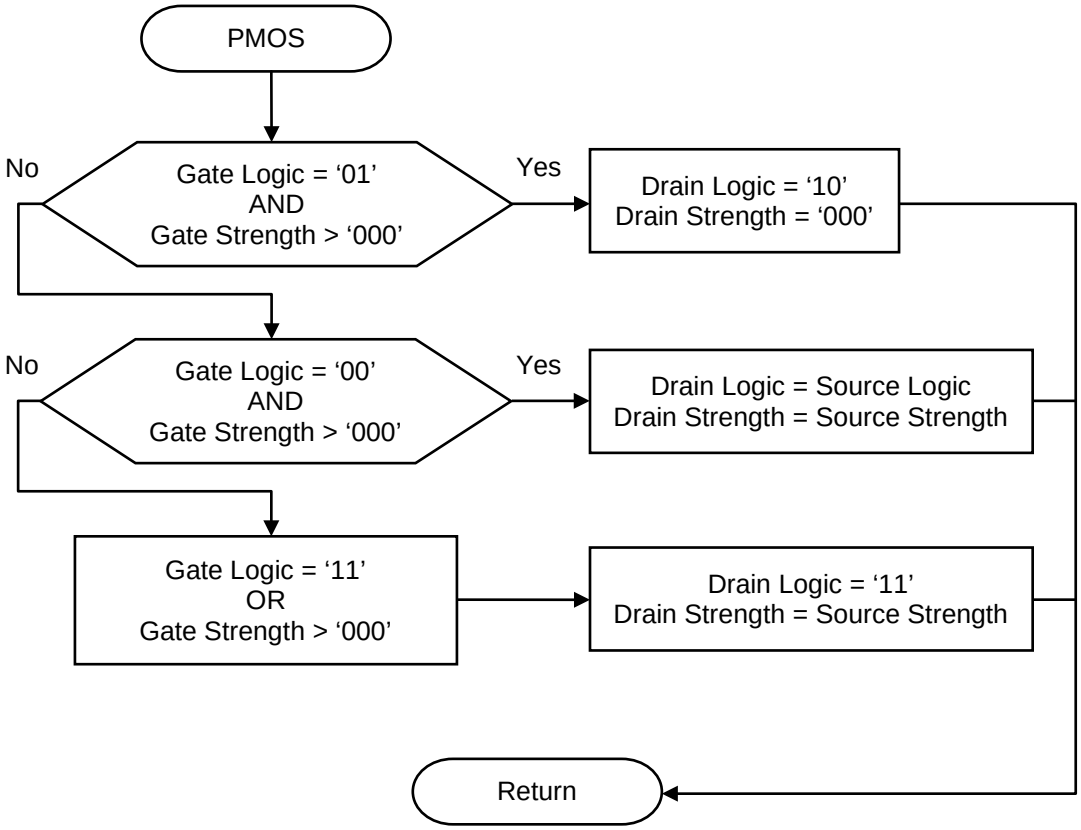


Figure 21 - PMOS Function flowchart.

When the logic level of signal at gate is “01” and strength level is greater than “000” (i.e. the signal is at logic 1 and strength level higher than high impedance), PMOS switch is turned off. Therefore, drain gets state code “10000” which signifies “don’t care” state indicating that PMOS

switch does not let any signal pass through. When logic level of signal at gate of PMOS switch is “00” with strength level higher than “000”, PMOS switch is turned ON. Signal from source passes on to the drain, which means “Drain Strength” and “Drain Logic” become equal to “Source Strength” and “Source Logic” respectively. Furthermore, if gate of PMOS is connected to a signal of strength level “000” (high impedance) or a signal of logic “11” (unknown), then “Drain Logic” is “11” and “Drain Strength” becomes equal to “Source Strength”.

Pseudo-code for PMOS function

If gate_logic is “1” and gate_strength is greater than “Z”

Drain_logic = don’t care

Drain_strength = “Z”

Else if gate_logic is “0” and gate_strength is greater than “Z”

Drain_logic = source_logic

Drain_strength = source_strength

Else if gate_logic is “Unknown” or gate_strength is “Z”

Drain_logic = Unknown

Drain_strength = source_strength

5.3.2 NMOS Function

Figure 22 shows the flow diagram for Function N based on the above mentioned Verilog rules. “Drain Logic” represents logic level of signal at drain while “Drain Strength” represents strength level for signal at drain of NMOS switch. Similarly, “Gate Logic” and “Gate Strength”

signify logic level and strength level for signal at gate and “Source Logic” and “Source Strength” represent logic level and strength level of signal at source of NMOS switch.

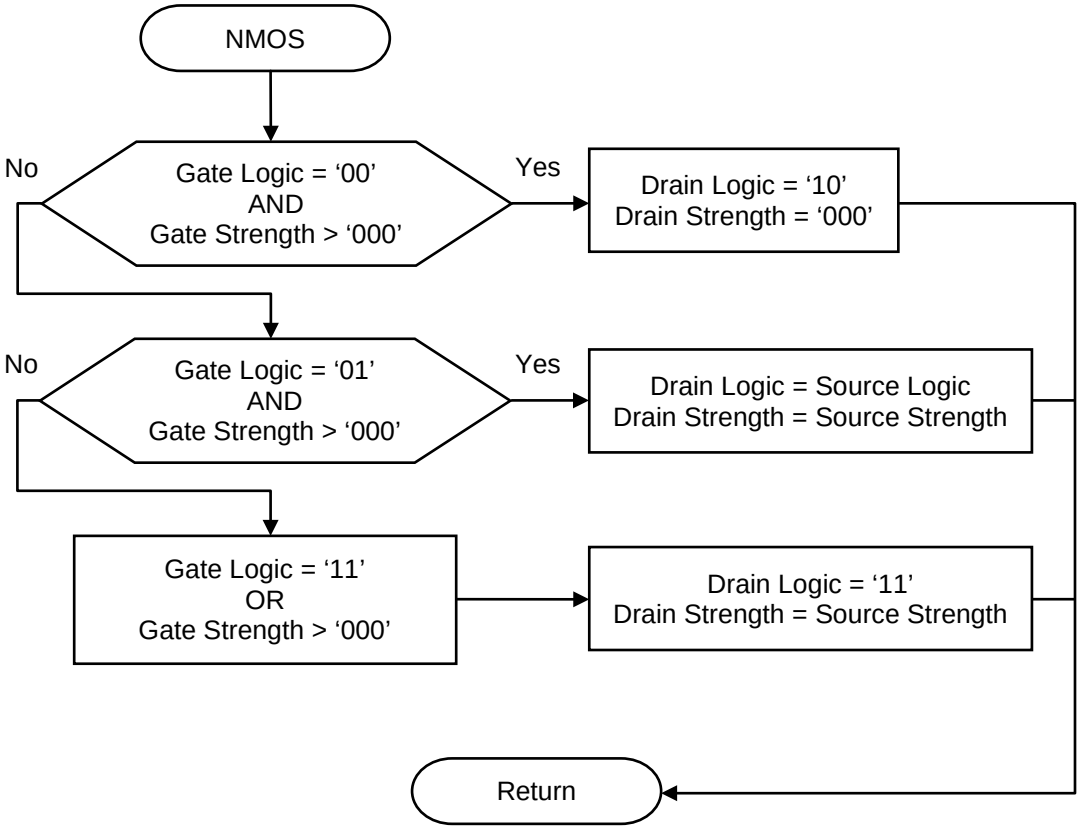


Figure 22 - NMOS Function flowchart.

When the logic value at the gate is “00” and strength level is greater than “000” (the signal is at logic level “0” and strength level higher than high impedance), the NMOS switch is turned off. Therefore, the drain gets state code “10000”, which signifies “don’t care” state indicating that NMOS switch does not let any signal pass through. When logic level is “01” with strength level higher than “000”, NMOS switch is turned on and thus signal from the source passes on to the drain (“Drain Strength” and “Drain Logic” become equal to “Source Strength” and “Source

Logic" respectively). If gate of NMOS is connected to a signal of strength level "000" or a signal of logic level "11", the "Drain logic" is "11" and "Drain Strength" becomes equal to "Source Strength".

Pseudo-code for NMOS function

If gate_logic is "0" and gate_strength is greater than "Z"

Drain_logic = don't care

Drain_strength = "Z"

Else if gate_logic is "1" and gate_strength is greater than "Z"

Drain_logic = source_logic

Drain_strength = source_strength

Else if gate_logic is "Unknown" or gate_strength is "Z"

Drain_logic = Unknown

Drain_strength = source_strength

5.3.3 C Function

Figure 23 shows the flow diagram of C Function which represents a connection node with signals "S1" and "S2" incident on it. The strength level for "S1" is indicated by "Strength S1" and logic level is indicated by "Logic S1". Similarly, strength level and logic level of "S2" are indicated by "Strength S2" and "Logic S2" respectively. If one of the signals is "10000" (i.e. the switch corresponding to that signal is off), then the other signal passes through the node. If both the signals are "10000", this indicates that both the switches attached to the node are turned off. In this case, the resolved signal is "10000". If the two signals have the same logic level but

different strength levels, then the signal with higher strength level passes. On the other hand, if the signals have the same strength level but different logic levels, then the resolved signal has “unknown logic” and strength level is the same as the strength level of signals incident on the node.

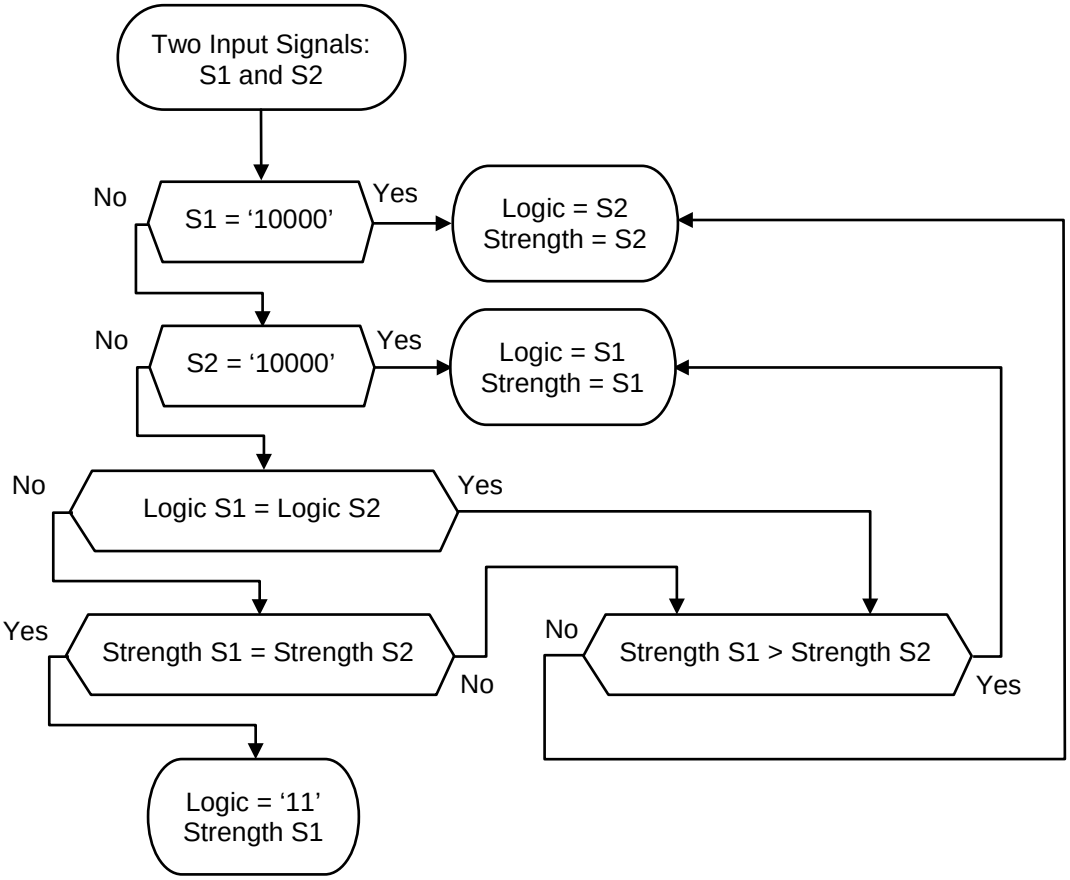


Figure 23 - C Function flowchart.

In nanometer technology circuits, node capacitances are small, so capacitor charge strength is taken as “small” and its logic value is taken from the resolved signal at the node. This idea is based on the fact that the values stored in the capacitance unit are weak values as they result from small charges [35].

Pseudo-code for C Function (connection node)

If one of the incoming signals is “don’t care”

Second signal passes through

Else if both incoming signals have same logic values

If strength is different

Signal with larger strength passes through

Else if strength is same

Result is same as incoming signals

Else if both signals have different logic values

If strength is same

Result is unknown logic and incoming strength

Else if strength is different

Signal with larger strength passes through

In this chapter the offered approach used a novel coding system to be applied in certain functions that are sensitive to strength variations. It was able to detect even the slight changes in signal strength caused by both cosmic rays and alpha particle from package contamination. Most soft-error detection techniques sense the logic changes in the circuit while this method proves that a wide range of soft-errors are the result of strength violation in switch-level. Next chapter is introducing Soft-Error injection techniques in three main locations in circuits.

CHAPTER 6
SOFT-ERROR INJECTION TECHNIQUE

The location of Soft-Error injection is playing a very important role in Soft-Error detection. This chapter will develop and analyse the Soft-Error techniques for the three locations of the Gate input, the Gate of a transistor and the Drain of a transistor.

6.1. Soft-Error Injection

To evaluate the error resilience of the schemes, soft-error injection was conducted during the execution-driven simulation. The choice of error injection location plays an important role in evaluating the effectiveness and accuracy of any error detection mechanism. All switch-level simulation-based soft-error detection methods devised so far inject current pulses at the internal nodes of the circuit assuming that internal nodes represent the drains of the transistors connected to the nodes [58]. FPGA-based soft-error emulation systems described in existing literature [59] focus on injection of transients at the inputs of gates. For soft-error detection techniques based on injection of bit-flip error models, transistor gate or input of a gate can be used as point of

injection without any loss of accuracy. However, the work presented in this thesis is not based on normal bit-flip based soft-error model. Rather, it uses a detailed soft-error model based on 23 different types of soft-errors resulting from voltage pulses of different logic and strength levels. The switch-level models used in this research may lead to miscalculated error coverage if the location of transient injection is not selected appropriately. In this section, theoretical analysis of the switch-level models is carried out to explain the effect of transient injection location on error coverage. The three possible alternatives for transient injection are injection at the input of a gate, injection at the gate of a switch, and injection at the drain of a switch. These three choices are analyzed by taking switch-level 2-input NAND gate as an example. The analysis shows that drain is the most appropriate and accurate choice for the injection of strength-based transient errors.

6.1.1. Pulse Injection at Gate Input

The injection position and size of the gate are the major control parameters for a simulated injection-mould. Once the injection parameters (gate size and gate position) are given, the product performance (deformation) can be accurately predicted by the abductive network developed. In most of the error injection techniques used so far, errors are injected at the inputs of gate-level circuit. This can lead to inaccurate results as injection at the input of one gate corresponds to injection of the same error at more than one location at switch-level. For example, if an error is injected at the input of an inverter, it gets injected at gates of two transistors which is equivalent to multiple bit-flips. This may lead to inaccurate error coverage calculations for single bit-flip based soft-error detection as each error may get counted twice.

6.1.2. Pulse Injection at the Gate of Transistor

One of the alternatives to avoid the above miscalculations is to use the gate of a switch as the location for error injection. For the presented strength-based switch-level soft-error detection method, transient injection at the gate of the switch poses another problem. The propagation of a transient pulse injected at the gate of the switch-level model of any gate is a function of input vector "V", logic level of the signals incident at the node to which the switch is connected "L_{in}" and the state of other switches connected to that node "T_{state}". This can be represented as:

$$P_g\,(SET_{prop}) = f\,(V, L_{in}, T_{state})$$

$P_g\,(SET_{prop})$ is probability of Single Event Transient (SET) induced pulse injected at gate to be transmitted to output. L_{in} is logic level of signals incident on the resolving node and T_{state} is state of the next MOS switch ON or OFF.

6.1.3. Pulse Injection at the Drain of Transistor

Previous research shows that when radiation strikes a node inside the circuit, it generates voltage pulse at the drain of the struck transistor [60]. Therefore, considering the drain of a switch as a transient injection location is quite a relevant option. This technique is closer to transistor level circuits as in this case the injected pulse participates in resolving the final output of the node. Moreover, in Nanometre technology CMOS devices, the most sensitive areas are the depletion regions at the transistor drain [61]. The propagation of an SET-induced pulse injected at the drain

takes into account the effect of input vector “V”, logic level and strength level of the signals incident at the node to which the switch is connected (L_{in}, S_{in},) and the state of other switches connected to that node “T_{state}” and the storage strength of node capacitance “S_{cap}” as indicated by the following equation:

$$P_d\ (SET_{prop}) = f\ (V, L_{in}, S_{in}, S_{cap}, T_{state})$$

$P_d\ (SET_{prop})$ is the probability of SET-induced pulse injected at the drain to be transmitted to output. S_{cap} is the storage strength of node capacitance and S_{in} is the strength of signals incident on the resolving node.

6.2. Soft-Error Injection Examples

In the following, some examples are brought to further clarify the above theoretical analysis. The propagation of transient pulses will be discussed step-by-step with a 2-input NAND gate for different transient injection locations.

6.2.1. Injection at the Input of 2-Input NAND Gate

For example, when transient pulse “00001” is applied at one of the inputs of 2-input NAND gate at switch-level, it affects two switches at the same time as shown in Figure 24. Let the values of the two node capacitances C1 and C2 be “01001” and both the inputs of NAND gate be “High=01111” at the time of transient injection. The injection of transient at input “A” switches on the P1 transistor and turns off the N1 transistor. As N1 is turned off, it does not participate in

signal resolution. As a result, the signal “01111” coming from the P1 reaches the primary output Y. For a NAND gate, when both inputs are high, the output should be “Low” but introduction of signal strength variations at two different switches at the same time results in logic change at the output. This logic-flip may result in a soft-error if it gets latched. The soft-error emulation technique presented in this thesis is based on a single bit-flip model, but injection at input of a CMOS gate is equivalent to multiple bit-flips. The C node in the figure represents the C Function.

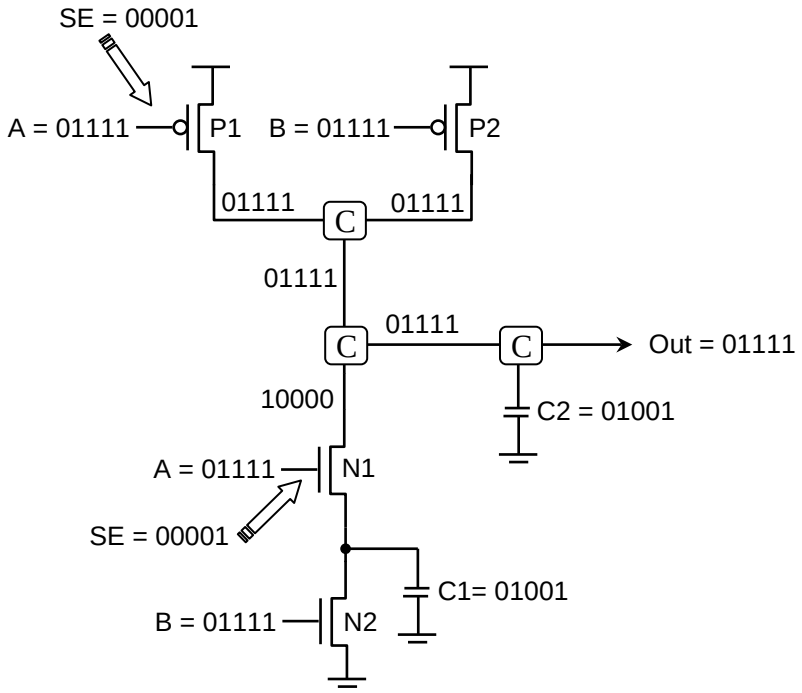


Figure 24 - SEU “00001” injected at the input of 2-input NAND gate.

In another example, the value of “01010” will be injected at the input of switch-level NAND gate. Transistor P1 is off and N1 is switched on as shown in Figure 25. Since both NMOS N1 and N2 are switched on, signal “00111” reaches the node. Since both the PMOS are switched

off, the primary output depends completely on the signal coming from NMOS network and the output capacitance value. The two values will collide at node C and the outcome value will be calculated using C Function. The signal stored in capacitance C2 is “01001”, which has a lower strength level than “00111” coming from NMOS network. The value of primary output changes to “00111”. Thus, the logic value of the output flips as the actual output is supposed to be “01111” for a NAND gate.

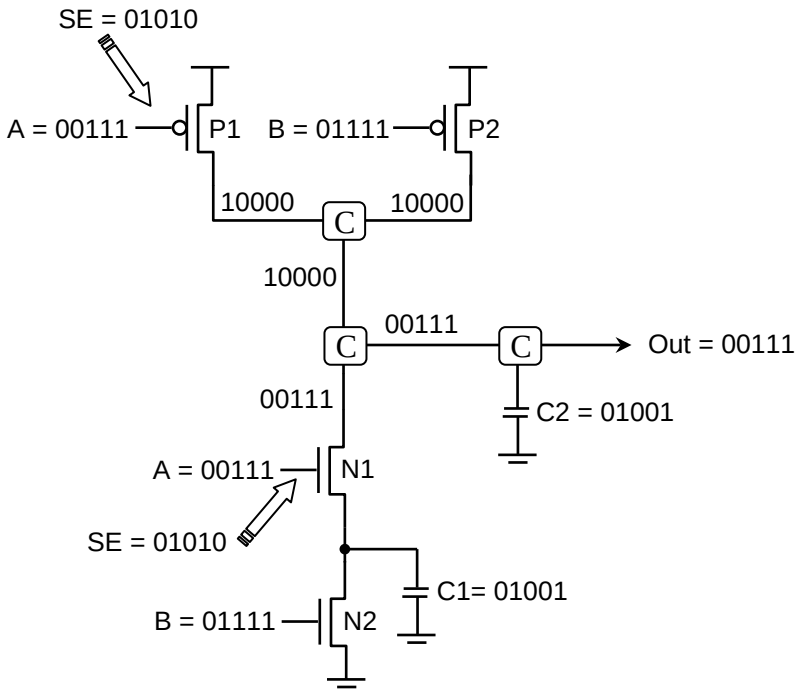


Figure 25 - SEU “01010” injected at the input of 2-input NAND gate.

6.2.2. Injection at Gate

In this example, again the value of “00001” will be injected at the gate of one of the switches, for example, P1 shown in Figure 26. Transient “00001” injected at gate G1 of transistor P1

results in switching on the P1. If both the inputs of the gate are “High” at the time of transient injection, then both NMOS are on. Therefore, at node C, two signals of the same strength but opposite logic values meet (signal “00111” coming from NMOS network and signal “01111” coming from P1 due to injected transient). As per the switch-level functions, this results in a signal of unknown logic at the primary output. Therefore, the output of NAND gate turns out to be “11111” which results in a soft-error.

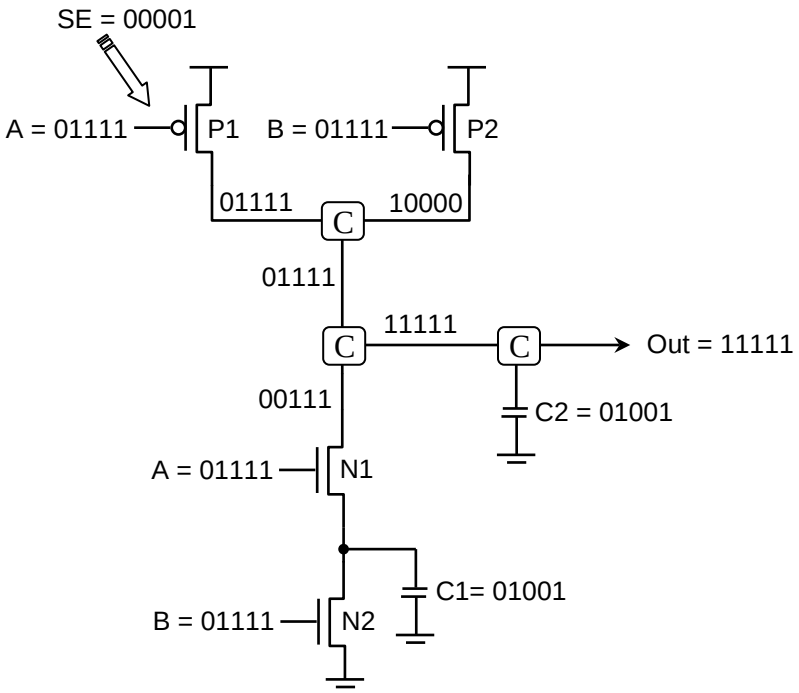


Figure 26 - SEU “00001” injected at the Gate of 2-input NAND gate.

In the second example as shown in Figure 27, transient pulse “01010” injected at the gate of P1 transistor does not affect the primary output as the gate treats it as “00111”. This turns off PMOS P1. Since NMOS N1 is in off state, the NMOS network does not participate in signal

resolution at the node. Hence, due to the effect of the injected transient, the signal stored in node capacitance C2 passes to the primary output. As the node capacitance is assumed to be “01001”, it does not result in any error as the logic value is the same as the non-faulty output. Thus, in this case, the injected transient does not cause an error.

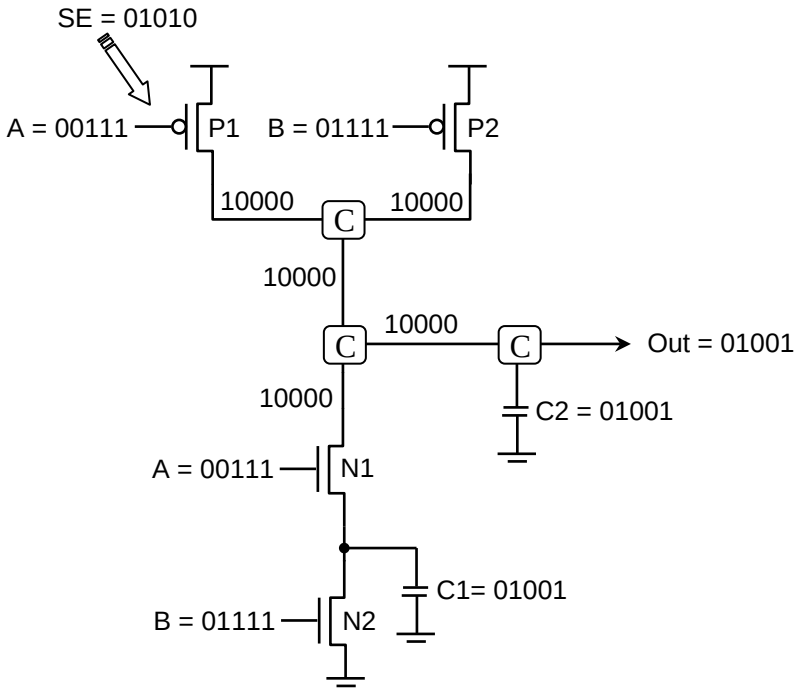


Figure 27 - SEU “01010” injected at the Gate of 2-input NAND gate.

6.2.3. Injection at Drain

In the following example, transient “00001” will be injected at the drain of P1 transistor as shown in Figure 28. In this case, the injected transient pulse gets masked by a higher strength signal “00111” coming from the NMOS network. Therefore, no logic change occurs at the

primary output of the gate. This shows that injection at the drain takes into account the effect of logical masking in combinational logic.

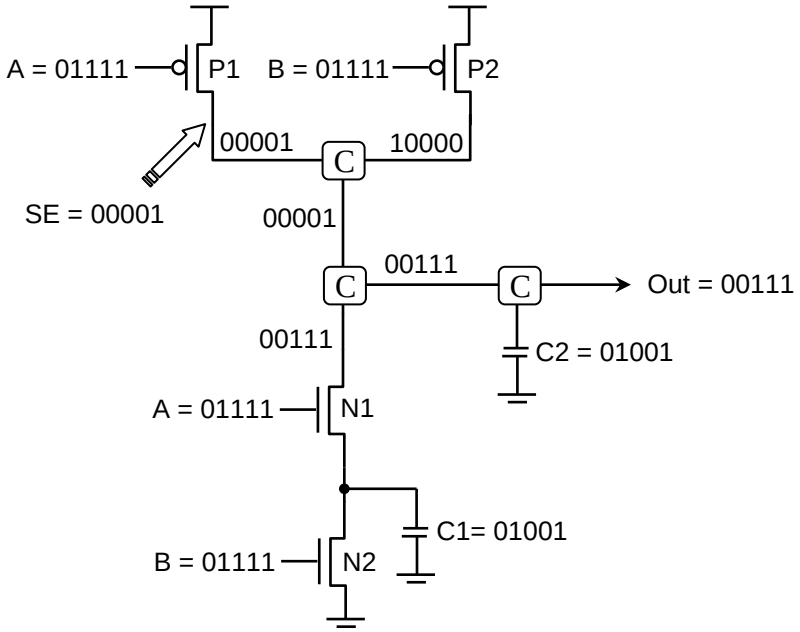


Figure 28 - SEU “00001” injected at the Drain of 2-input NAND gate

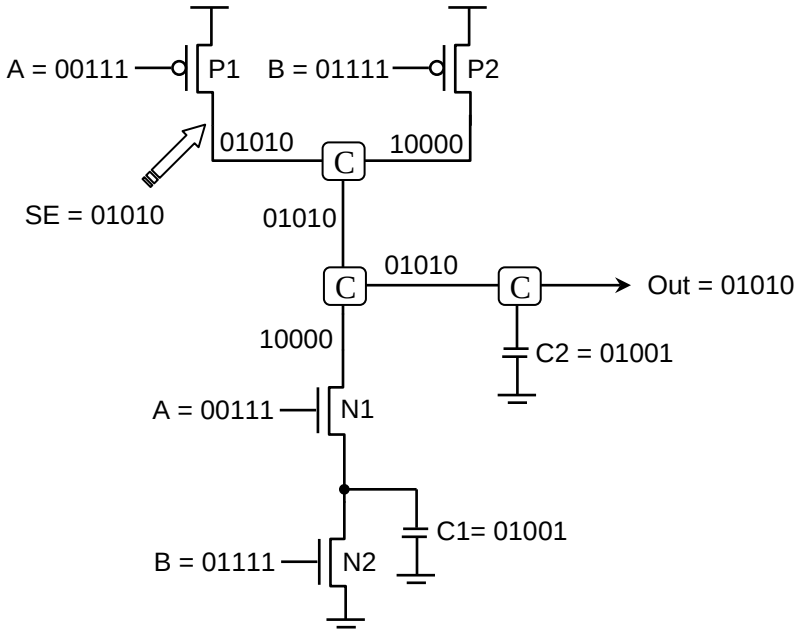


Figure 29 - SEU “01010” injected at the Drain of 2-input NAND gate.

When transient pulse “01010” is injected at the drain of P1 transistor, it passes on to the primary output without getting masked. All details are shown in Figure 29.

In order to apply and test the efficiency of new invented model for strength-based soft-error detection that was explained in detail in chapters 5 and 6, soft-error simulation software was developed in MathLab environment. The soft-error simulation software and its simulation result will be discussed in following chapter.

CHAPTER 7
SIMULATION RESULTS

The simulation environment was generated using MATLAB in order to first convert the ISCAS'85 Verilog gate-level net-lists into switch-level circuit based on the advanced Switch-level model explained earlier. A simulation system was designed to allow injection of soft-errors at gate inputs, drain and gate in switch-level. The simulator functions are divided into different levels that will be explained in detail in this chapter.

7.1 Data Acquisition

Data acquisition is the first level of simulation involving loading the net-list gate-level in a Verilog format. This file will be loaded as a text file. Simulation test vectors can be produced either by a random test vector generation such as a Linear Feedback Shift Register (LFSR), or Deterministic Test Pattern Generator (DTPG). In both case, users can determine the number of

applied test vectors. This thesis uses test set compaction algorithms for combinational circuits based on [62].

7.2 Data Processing

At this stage, the text files and the specified type of test vectors achieved from the previous level are loaded into the program. At the beginning, the scripts will code the gate types and organize the net-list gate-level into different categories, such as inputs, outputs, the gate types and their connections (in /out). The gate-level net-list at this stage is entirely coded in numeric form for data processing purposes. The coded gate-level net-list is processed for modifications in the links between the gates so that the links' addresses correspond to increasing numbers without any gaps. For example, Table 7 shows C17 gate-level net-list fragmentation.

As evident through the links that connect the gates, there are some missing link numbers in C17 before modification. These numbers are reorganized in Table 7 for C17 after modification. The maximum link indexes in C17 before modification is 23 and after modification 11.

The elimination of these gaps is useful for programming and for reducing the arrays size as they are linked to an array that stores the logic/strength value of the link at different stages of the simulation. Eventually, the gate-level net-list data will be converted to a switch-level net-list. Every gate is converted to switch-level. The functions used to replace the gate-level are the PMOS function, NMOS function and Function C explained earlier.

Table 7 - C17 gate net-list before and after modification.

C17 before modification	C17 After modification
// Verilog	// Verilog
// c17	// c17
// Ninputs 5	// Ninputs 5
// Noutputs 2	// Noutputs 2
// NtotalGates 6	// NtotalGates 6
// NAND2 6	// NAND2 6
module C17	module C17
(N1,N2,N3,N6,N7,N22,N23);	(N1,N2,N3,N4,N5,N10,N11);
input N1,N2,N3,N6,N7;	input N1,N2,N3,N4,N5;
output N22,N23;	output N10,N11;
wire N10,N11,N16,N19;	wire N6,N7,N8,N9;
nand NAND2_1 (N10, N1, N3);	nand NAND2_1 (N6, N1, N3);
nand NAND2_2 (N11, N3, N6);	nand NAND2_2 (N7, N3, N4);
nand NAND2_3 (N16, N2, N11);	nand NAND2_3 (N8, N2, N7);
nand NAND2_4 (N19, N11, N7);	nand NAND2_4 (N9, N7, N5);
nand NAND2_5 (N22, N10, N16);	nand NAND2_5 (N10, N6, N8);
nand NAND2_6 (N23, N16, N19);	nand NAND2_6 (N11, N8, N9);
endmodule	endmodule

7.3 Main Simulation Program

The main program in the simulation process can determine the number of sites for soft-error injection or simply decide the location of the injected faults. A mapping array allows the user to inject the fault at a specific location as well as the desired fault type. Table 8 shows all 23 types of fault for C17 benchmark. When it comes to inject the fault type “00111”, the simulator takes into consideration the calculated value at the injected site before injecting the fault. If the calculated value of the fault free circuit at this specific location is equal to “00111”, then the program will simply flip the bit of the logic fault type and the fault becomes “01111”.

Table 8 - C17 fault types and relevant SER.

c17	Logic	Strength	Faults	SER
1	01	111	77	1
2	00	000	129	0.958333
3	00	001	144	1
4	00	010	253	0.75
5	00	011	253	0.75
6	00	100	253	0.75
7	00	101	253	0.75
8	00	110	253	0.75
9	01	000	129	0.958333
10	01	001	211	0.916667
11	01	010	290	0.708333
12	01	011	290	0.708333
13	01	100	290	0.708333
14	01	101	290	0.708333
15	01	110	290	0.708333
16	11	000	129	0.958333
17	11	001	110	1
18	11	010	110	1
19	11	011	110	1
20	11	100	110	1
21	11	101	110	1
22	11	110	110	1
23	11	111	57	1
Total			4251	0.873188

In Table 8, fault types refer to all logic and strength level combinations. For each fault type, a number of faults have been injected as shown in this table. There are 77 fault locations for the first fault type and in total 4251 faults are injected for C17 in switch-level model.

This method incorporates a feature in the simulator that breaks the loop cycle of the test vectors when the fault is detected. This feature allows the number of applied test vectors and the

simulation time to be reduced. Eventually, SER is calculated for each fault type and the average of SER = 0.8731 is achieved for C17.

7.4 Simulation Result for all ISCAS85 Benchmark

The simulation result for C432 (27-channel interrupt controller) is shown in Table 9. The result for all ISCAS85 benchmarks can be found in Appendix section at the end of this thesis.

Table 9 - C432 fault types and relevant SER.

c432	Logic	Strength	Faults	SER
1	01	111	5846	1
2	00	000	11501	0.920759
3	00	001	11663	0.970424
4	00	010	22229	0.722098
5	00	011	22229	0.722098
6	00	100	22229	0.722098
7	00	101	22229	0.722098
8	00	110	22229	0.722098
9	01	000	11501	0.920759
10	01	001	16980	0.864397
11	01	010	25457	0.657366
12	01	011	25457	0.657366
13	01	100	25457	0.657366
14	01	101	25457	0.657366
15	01	110	25457	0.657366
16	11	000	11501	0.920759
17	11	001	6623	0.979353
18	11	010	6623	0.979353
19	11	011	6623	0.979353
20	11	100	6623	0.979353
21	11	101	6623	0.979353
22	11	110	6623	0.979353
23	11	111	2573	1
Total			349733	0.842197

7.5 Simulation Summary

The study of simulation results for all circuits shows that they all follow the same pattern for the soft-error rate for different fault types. This fact is presented in Figure 30.

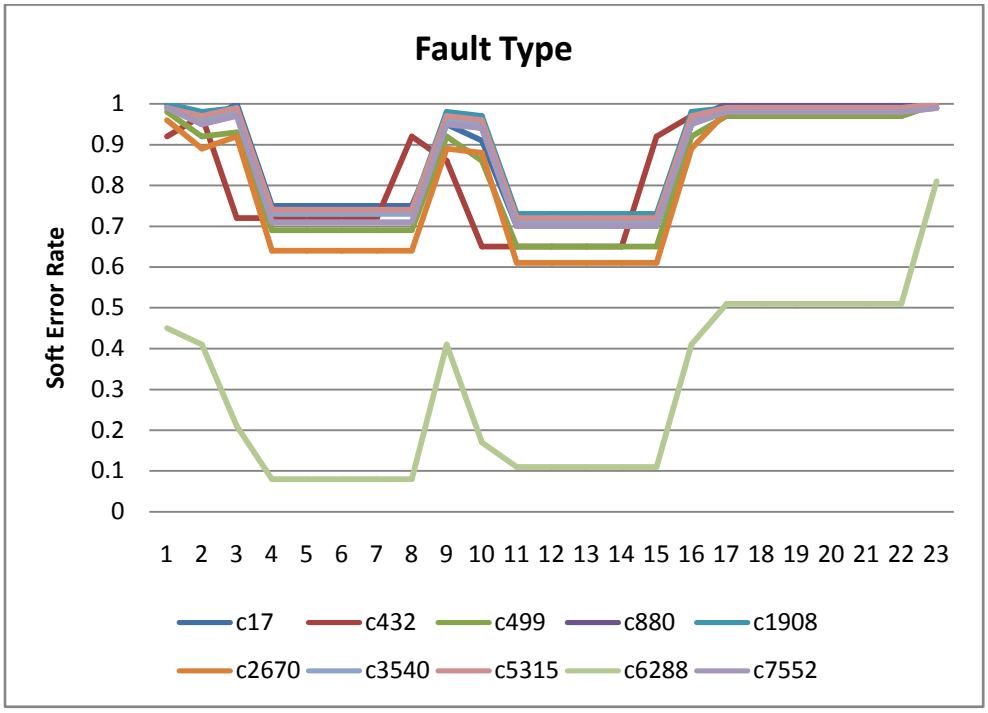


Figure 30 - SER versus Fault type.

As seen in this chart, the rate is higher for fault type “1” and “2” then drops until fault type “4”. It stays the same until the logic change in fault type “9” and so on. The most important point in Figure 30 is the rate coverage for benchmark C6288 was lower than others as expected. For a more detailed investigation of SER behaviour based on different logic, error types are divided into three major groups for logic “0”, logic “1” and logic “Unknown”. As illustrated in Figure 31, fault types 2 - 8 are considered in logic “0” and soft-error rates are shown for all benchmark circuits.

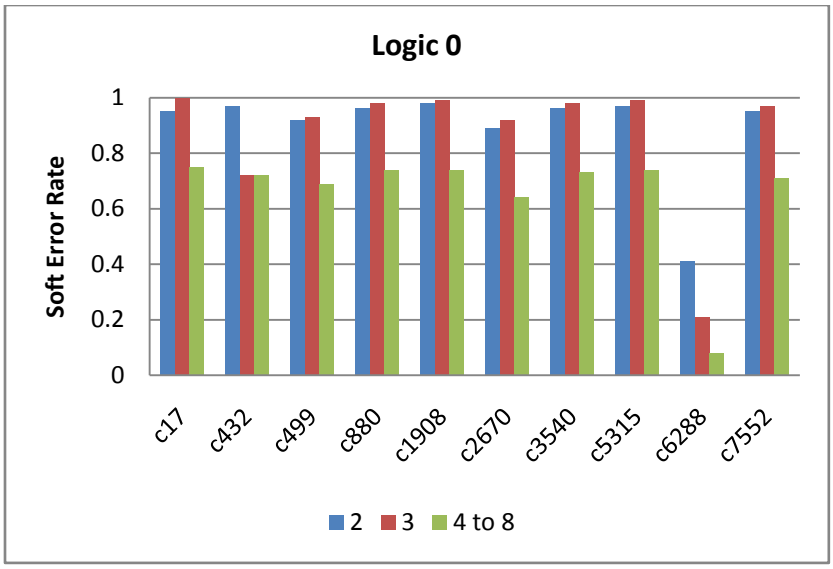


Figure 31 - SER changes based on fault type for Logic 0.

Similarly, fault types 9 - 15 as well as fault type 1 are considered in logic “1” as presented in Figure 32. Comparing the results in Figure 31 and Figure 32 shows that both logics follow the same pattern in terms of coverage for different circuits.

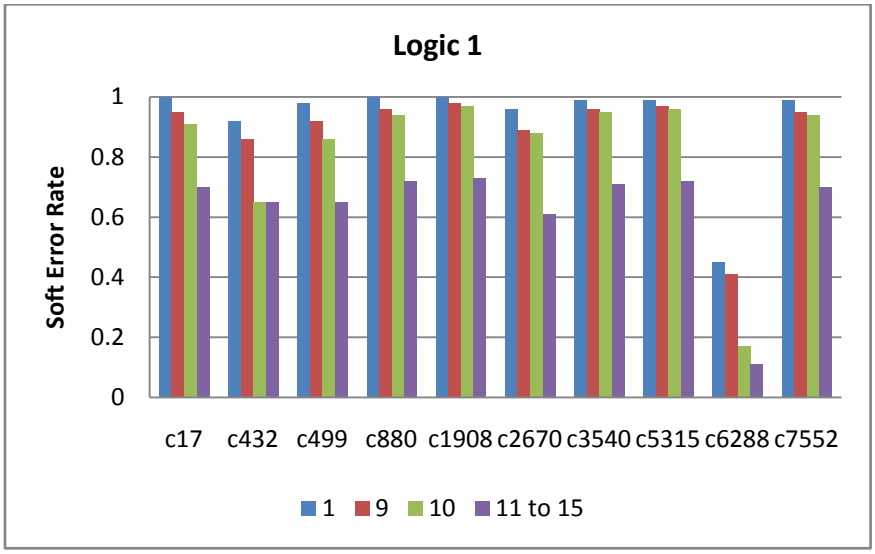


Figure 32 - SER changes based on fault type for Logic 1.

Finally, the third group of fault types is Logic = “11” which is considered as “Logic Unknown”. These faults range from 16 to 23. The soft-error rates for all benchmarks for “Logic Unknown” are shown in Figure 33.

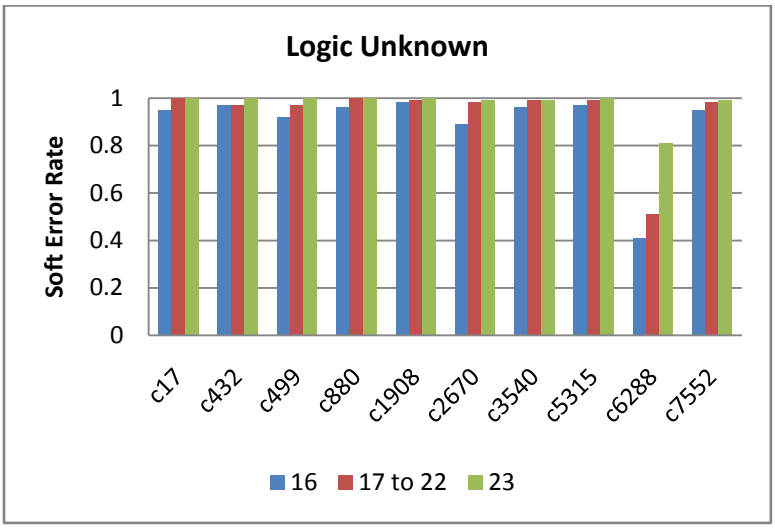


Figure 33 - SER changes based on fault type for Logic Unknown.

The overall average SER achieved from simulation for all ISCAS’85 benchmarks including the total number of faults injected as well as number of sites and test vectors are shown in Table 18.

Table 18 - Simulation result for ISCAS’85 benchmarks.

Circuits	Total injected Faults	Number of Sites	Number of Test Vectors	SER
C17	4021	48	12	0.87
C432	334687	1792	27	0.85
C499	1386383	4360	52	0.83
C880	391794	3604	16	0.87
C1908	3281154	6892	106	0.88
C2670	3040179	11336	44	0.81
C3540	6113154	15008	84	0.87
C5315	4458895	22524	37	0.87
C6288	5777985	24368	12	0.3
C7552	10569314	30800	73	0.86

The graph in Figure 34 displays the simulation results for soft-error rate in Table 18. Soft-error detection rates based on strength violation in switch-level for most ISCAS'85 benchmarks are above 0.8.

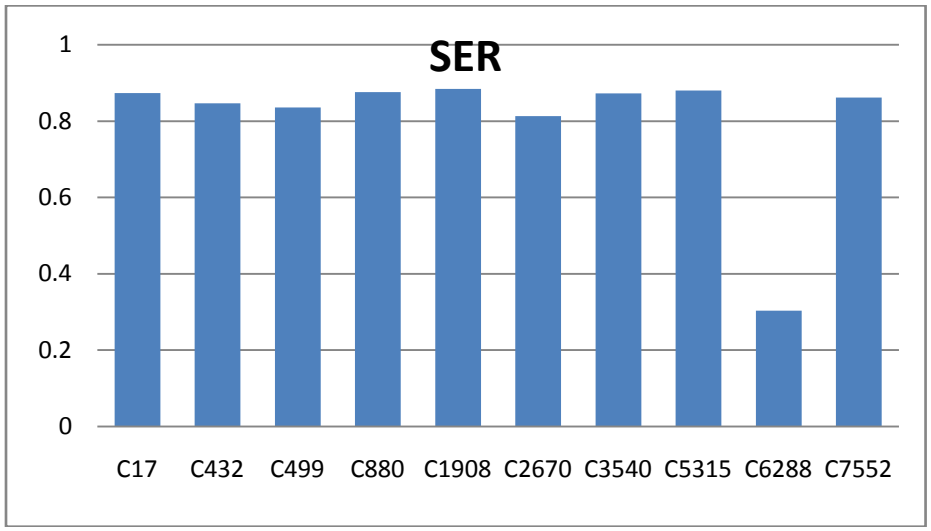


Figure 34 - SER graph for ISCAS'85 benchmarks.

The simulation result that has been achieved in this chapter is proving that a wide range of soft-errors are due to strength violation in circuit. However the simulation time for very large circuits are too long, so the entire idea was implemented in hardware base emulation which will be discuss in next chapter.

CHAPTER 8

FPGA –BASE EMULATION FOR SOFT-ERROR

FPGA-Base Emulation of the model discussed in the previous sections will be implemented in this chapter. The gate-level component has been used to implement the switch-level behaviour of the circuit. This emulation is an advanced extension of [67]. These switch-level components will be described as follows:

8.1 Implementation of Switch-Level Functions

- Component N: is functionally equivalent to the NMOS switch
- Component P: is functionally equivalent to the PMOS switch
- Component N_{VSS}: represents the NMOS switch with its source connected to V_{SS}
- Component P_{VDD}: represents the PMOS switch with its source connected to V_{DD}
- Cap_unit: represents the stray capacitance associated with each node in a switch-level circuit
- Res_unit: resolves the signals based on Function C described in previous section

8.1.1 Implementation of NMOS Function

Figure 35 shows the implementation of NMOS function using "N_{VSS}". For N_{VSS}, the source is permanently connected to "0" (GND); therefore, for this component, source was not included on primary ports. Signal "Sel" is a two bit signal that consists of Sel1 and Sel2. Whenever the "Sel1" or "Sel2" signal goes high, the error will be injected to the Gate or Drain accordingly.

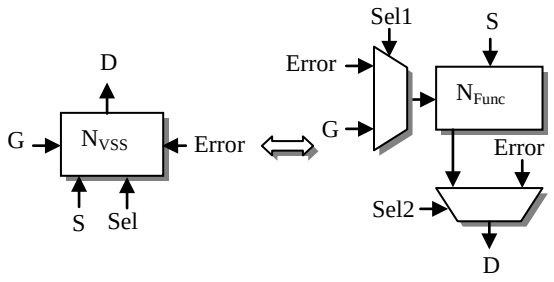


Figure 35 - Implementation of NMOS.

8.1.2 Implementation of PMOS Function

Component P was implemented in the same manner for PMOS function using P_{VDD}. Figure 36 shows the internal details of component P. It contains gate-level implementation of PMOS function along with two multiplexer units to inject the error into the gate or drain. For $P_{VDD,}$ the source is permanently connected to "1" (V_{DD}); therefore, for this component, source was not included on primary ports.

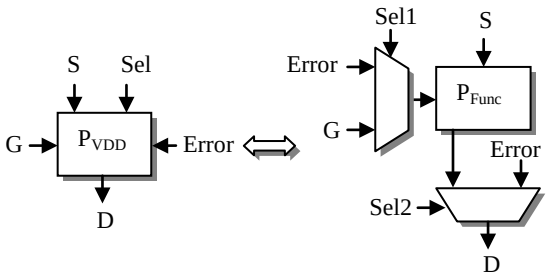


Figure 36 - Implementation of PMOS.

8.1.3 Implementation of C Function

Each node in the circuit is represented either by Res_unit or by a combination of Res_unit and a Cap_unit. The Res_unit is used to resolve two incoming signals and generate the dominant signal. It was implemented as a 5-bit comparator which generates the output signal based on the C Function described in Chapter 4. The Cap_unit is equivalent to the stray capacitance associated with the node. The output of Res_unit is fed to the Cap_unit after reducing its strength to “001.” This idea is based on the stray capacitance behaviour that the values stored in the Cap_unit are “weak” values as they result from small charges [35]. The output of the Res_unit will be “10000,” if all the switches connected to a node are OFF. In this case, the Cap_unit retains its previous value. The Cap_unit contains a memory element to store the previous value. Due to the presence of the memory element, the overall implementation of any combinational logic circuit becomes sequential. For a better understanding of the C function, the implementation of some of the Gates will be described below.

8.2 Switch-Level Implementation of Gates

Traditional FPGA-based fault emulation approaches are done at gate-level by means of hardware description language (HDL) modifications. Faults modeled at gate-level are not as realistic as switch-level faults because in real hardware, circuit outputs are computed by the switching of transistors. FPGA-based switch-level fault emulation has already been proposed in [63] to improve the accuracy of the fault model and to improve the emulation speed compared to software fault simulators. In this thesis, a similar approach is presented to implement the novel idea of detecting soft-errors caused by the strength violation on FPGA in switch-level discussed in previous chapters. To achieve this resolution, components N, P, P_{VDD}, N_{VSS}, Res_unit and Cap_unit are combined to form a switch-level structure for different logic gates. These switch-level structures are further combined to form switch-level implementations ISCAS'85 benchmark circuits. The process of combining the gate components to form larger benchmarks was automated by writing a script in MATLAB. In this section, switch-level implementations of basic logic gates used in this research are presented.

Switch-level implementations of all gates were constructed using the same approach. These gates were further combined to form switch-level implementations ISCAS'85 benchmark circuits. The process of combining the gate components to form larger benchmarks was automated by writing a script in MATLAB.

8.2.1 NAND3 Implementation in FPGA

Figure 37 shows how components N, P_{VDD}, N_{VSS}, Res_unit and Cap_unit are combined to form a switch-level structure for 3- input NAND gate.

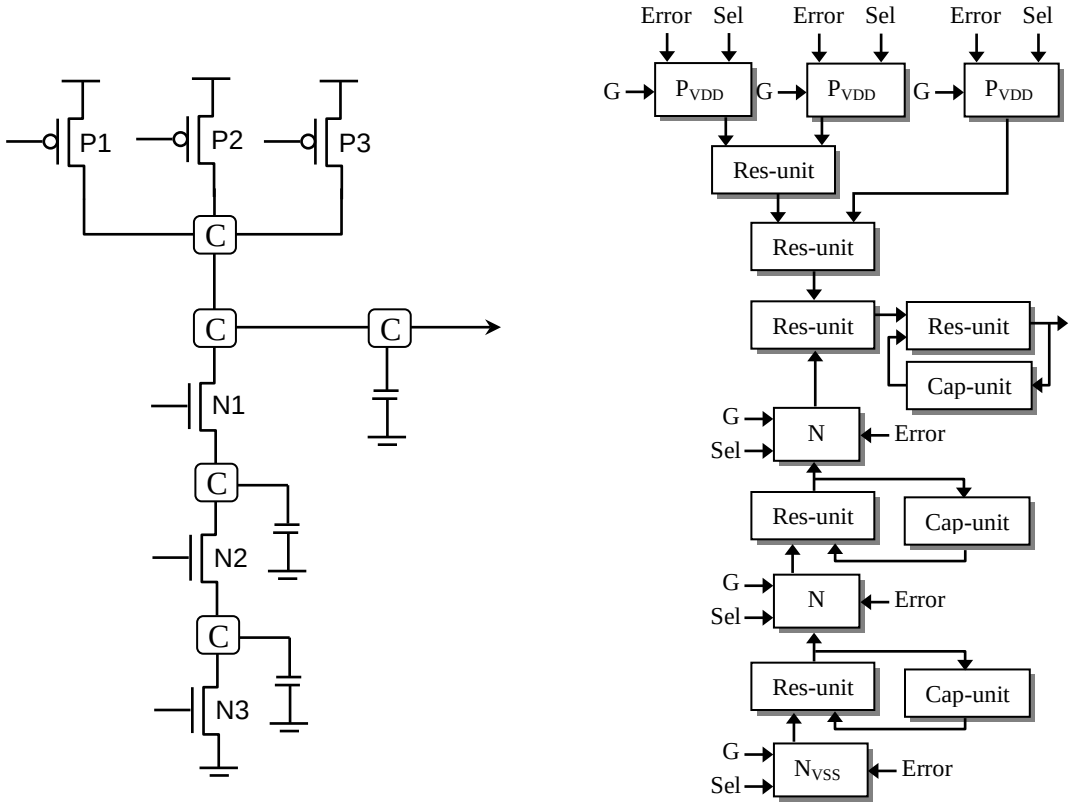


Figure 37 - Implementation of switch-level 3-input NAND gate.

8.2.1 AND3 Implementation in FPGA

The Figure 38 shows the implementation of a three input AND gate in FPGA.

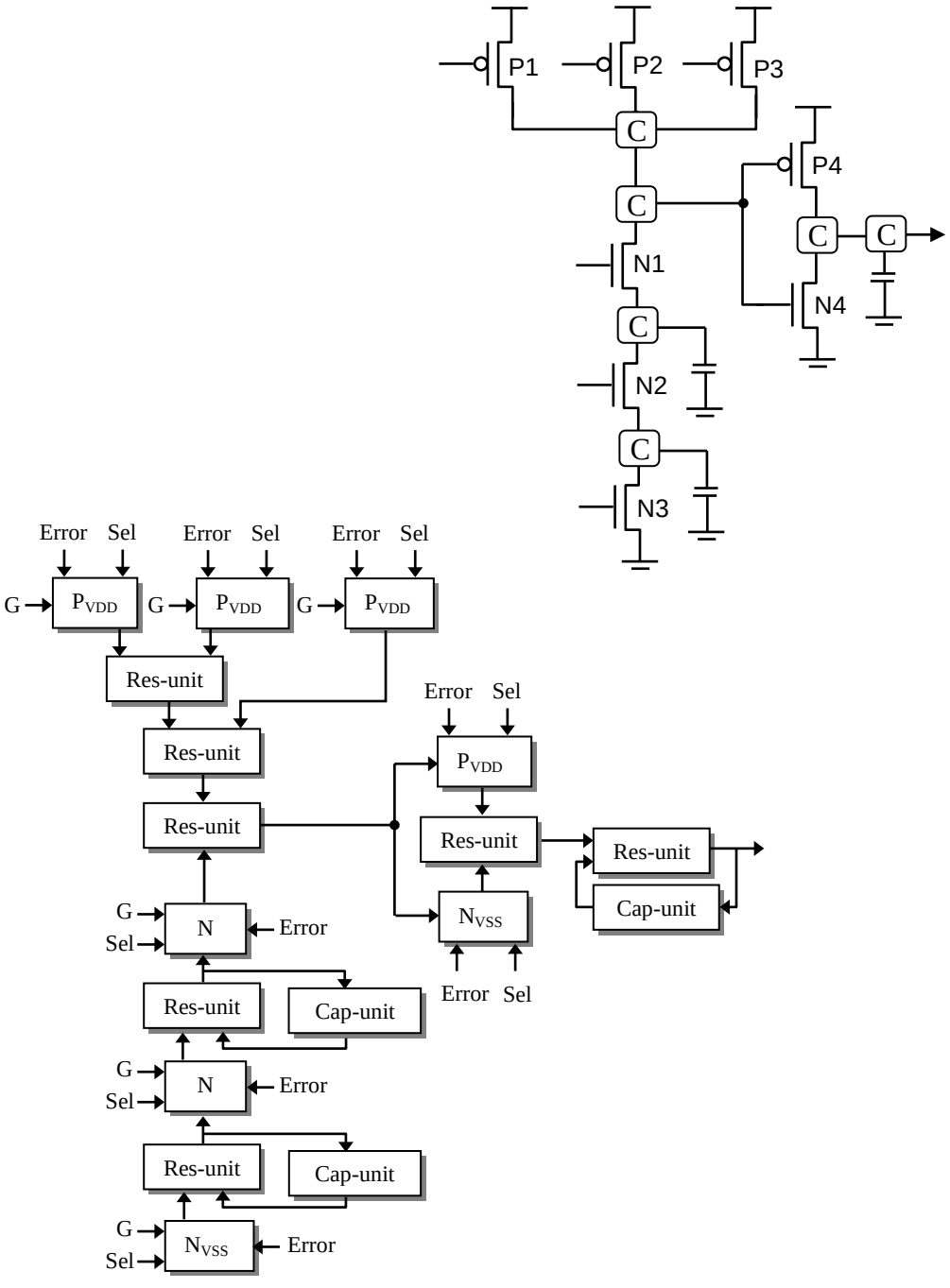


Figure 38 - Implementation of switch-level 3-input AND gate.

8.2.2 NOR3 Implementation in FPGA

The Figure 39 shows the implementation of a three input NOR gate in FPGA.

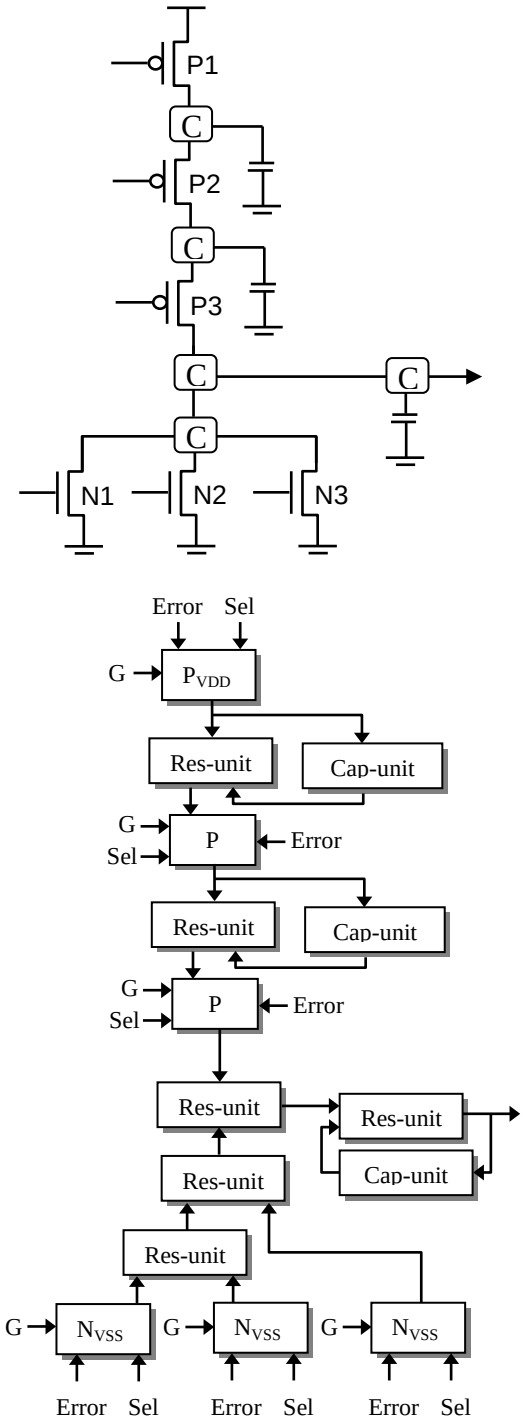


Figure 39 - Implementation of switch-level 3-input NOR gate.

8.2.2 OR3 Implementation in FPGA

The Figure 40 shows the implementation of a three input OR gate in FPGA.

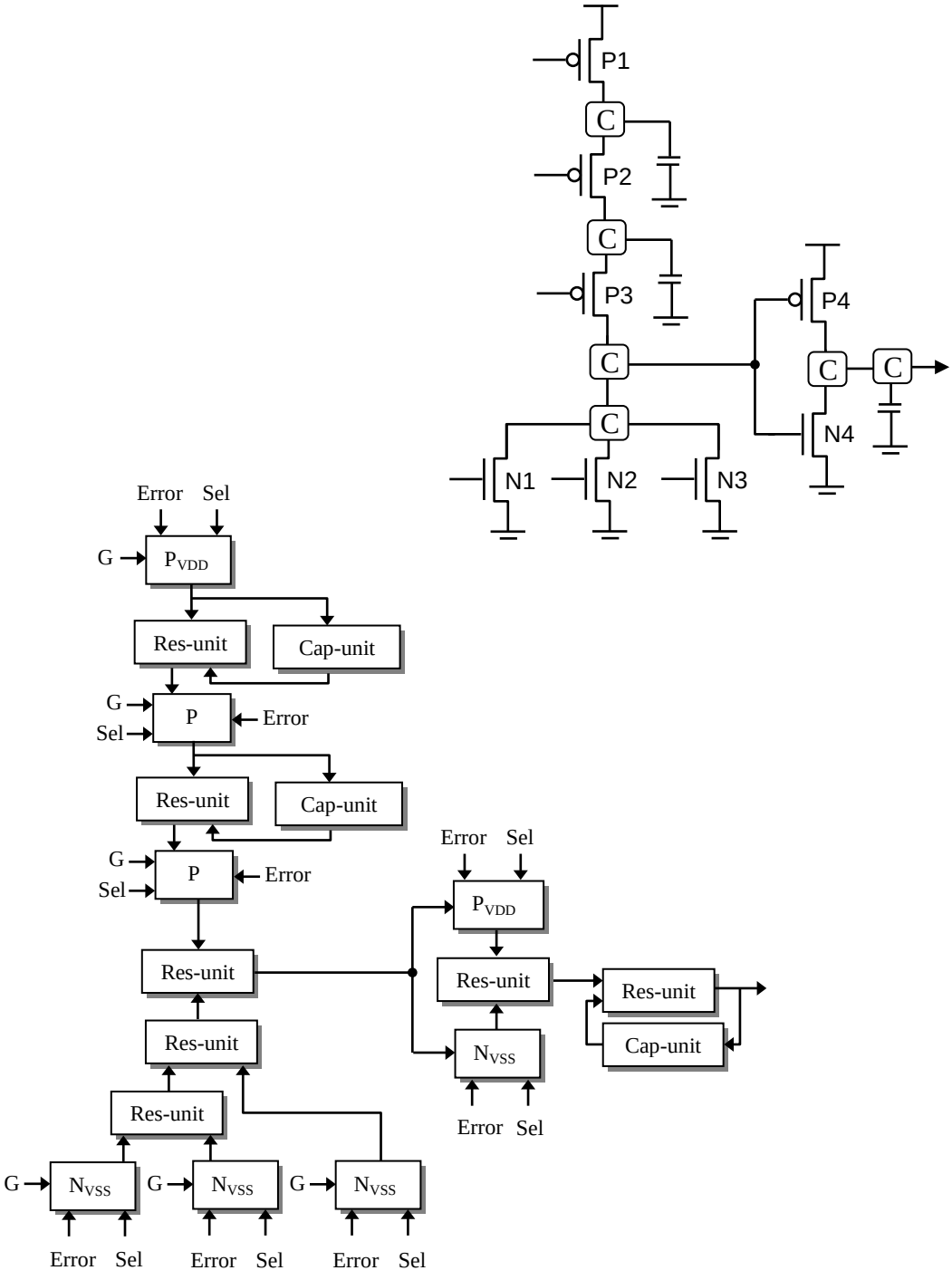


Figure 40 - Implementation of switch-level 3-input OR gate.

8.2.3 BUFFER Implementation in FPGA

The Figure 41 shows the implementation of a BUFFER gate in FPGA.

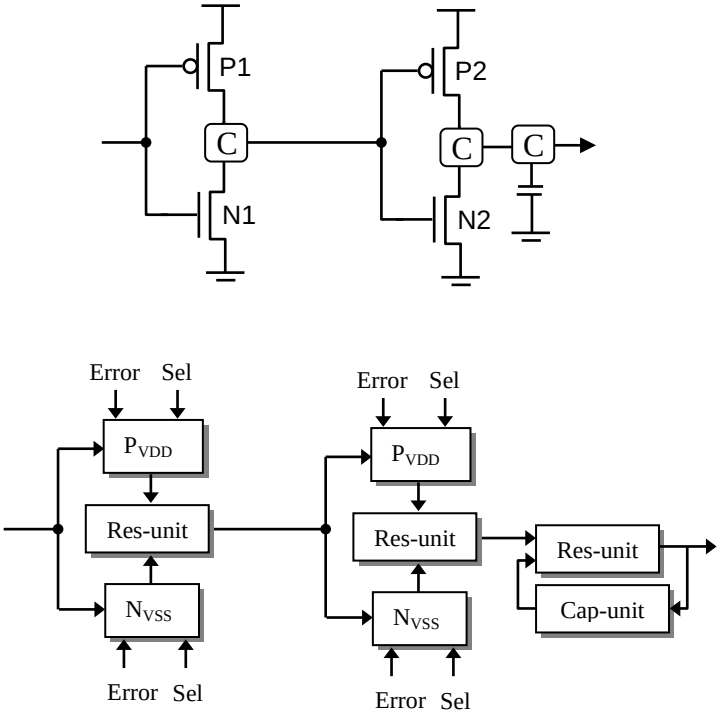


Figure 41 - Implementation of switch-level Buffer gate.

8.2.4 Inverter Implementation in FPGA

The Figure 42 shows the implementation of an Inverter gate in FPGA.

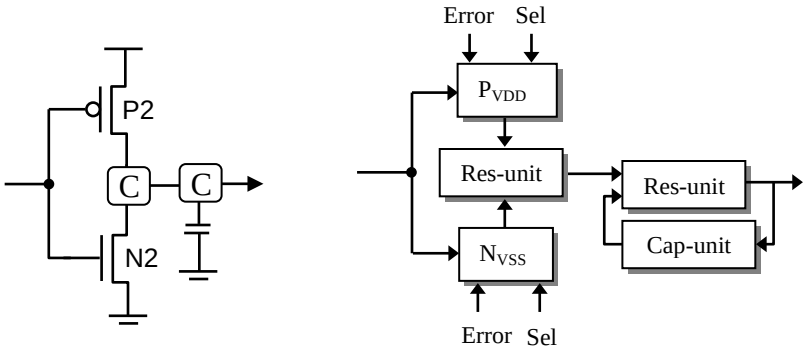


Figure 42 - Implementation of switch-level Inverter gate.

8.3 Emulator Architecture

The block diagram of the proposed emulation system is shown in Figure 43. This block diagram consists of core unit (CU), soft-error injection unit (SIU), observation unit (OU) and the soft-error coverage calculation unit (SCCU).

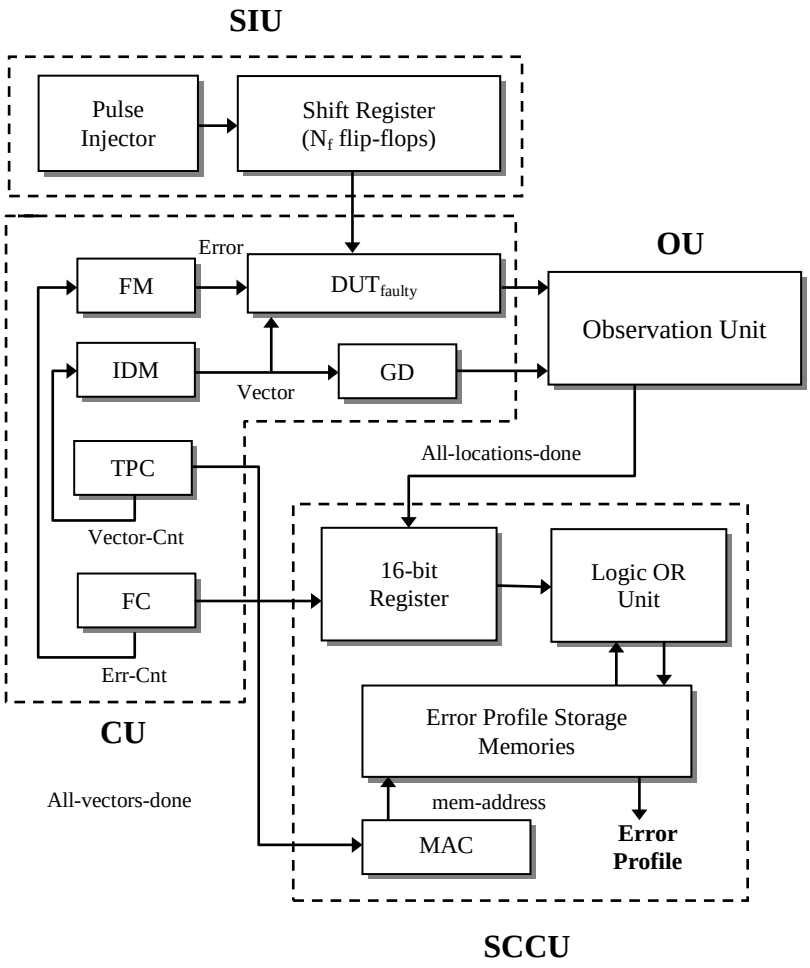


Figure 43 - Emulation Architecture.

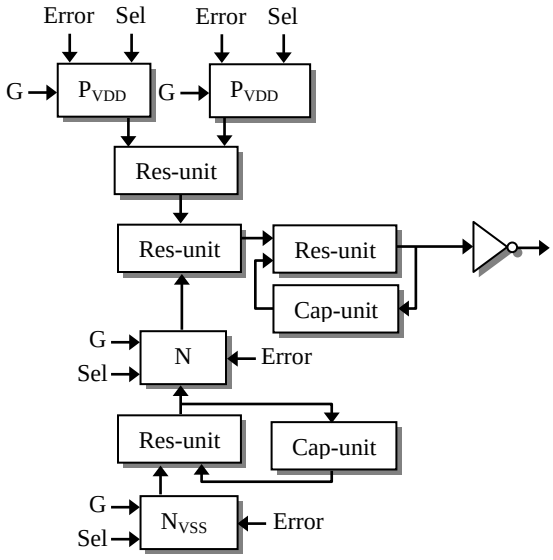
For a given Device Under Test (DUT) with “n” primary inputs and “m” primary outputs, N_f will be the number of selected soft-error locations. The descriptions of all the modules in the architecture block diagram are as follows:

SIU: Soft-Error Injection Unit, **FM**: Fault Memory, **IDM**: Input Data Memory, **TPC**: Test Pattern Counter, **GD**: Golden Device (fault-free gate-level circuit), **MAC**: Memory Address Counter, **SCCU**: Soft-Error Coverage Calculation Unit, **FC**: Fault Counter, **CU**: Core Unit

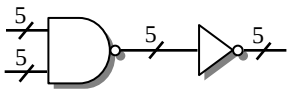
8.3.1 Core Unit (CU)

Core unit is where the device under test will be implemented with the proposed methodology. The fault-free gate-level implementation of the DUT is considered as golden device (GD). The block which is indicated by DUT_{faulty} is a hybrid version of DUT implemented by partitioning the circuit into gate and switch-level. It is obvious that purely switch-level implementation of any circuit using any switch-level models consumes much more resources than the gate-level implementation. The circuit under test is partitioned using an unbalanced partitioning structure so that faults are injected only in a small sub-circuit. Therefore, the size of the switch-level partition was decided on the basis of the available FPGA resources. The main idea was to minimize the number of switch-level partitions and hence reduce the number of reconfiguration files needed to emulate the complete circuit. This resulted in an overall reduction in run-time. Benchmarks C17, C432, C499 and C880 were implemented entirely at switch-level whereas the remaining benchmarks were too big to fit into the FPGA. The size of the switch-level partition for these benchmarks was fixed to cover approximately 2180 soft-error locations (N_f) and the circuits were implemented using hybrid representation consisting of switch-level

and gate-level partitions. To convert the number of gates corresponding to these soft-error locations into switch-level and generate the DUT_{faulty} component code, a script was written in MATLAB.



(a) Hybrid representation



(b) Gate-level Representation

Figure 44 - Propagation of transient pulse from switch-level part to gate-level part.

The switch-level partition is based on the switch-level models on [68]. Figure 44 (a) shows hybrid representation of circuit which is shown in Figure 44 (b). As the signals are 5-bit signals, the inputs and outputs of gate-level partition were 5-bit consisting of 2-bit logic and 3-bit strength part. To resolve the 2-bit logic part, dual-railed logic with three-valued logic signals

were used based on [40]. “00” was used to represent “0”, “01” for “1” and “11” for “Unknown” logic. The presented work is based on the idea of signal strength variations; therefore, the strength part of the incoming signal to the gate partition was preserved and transmitted further.

One of the objectives of the proposed approach is to estimate the soft-error coverage achieved by the soft-error detection method based on the concept of strength scaling for voltage pulses. The selection of primary input values plays an important role in the final results as randomly picked can result in lower soft-error coverage. To resolve this issue, Input Data Memory (IDM) has been considered to store the test patterns for the DUT. To maximize the error coverage for this approach, the primary input values generated by Compaction Algorithm [62] were selected for all ISCAS‘85 benchmarks. These inputs are commonly used for detection of permanent faults, but in this work they are being applied for the soft-error detection. The inputs are stored in memory before starting the experimentation. Each bit of an input data is converted into its equivalent 5-bit code before applying it to the switch-level circuit. Bit “0” is replaced by “00111” which represents logic “0” with highest strength “111” and bit “1” is replaced by “01111.” For the golden device circuit (gate-level), the input values are applied without conversion. Fault Memory (FM) stores the 5-bit codes of transient error types obtained after applying the concept of transient equivalence.

8.3.2 Soft-Error Injection Unit (SIU)

Traditional fault emulation approaches require dedicated fault injection circuitry and control pins to activate and deactivate faults in the circuit [69][70][71]. This not only adds hardware cost

but also design complexity. Field Programmable Gate Array (FPGA) device has been utilized as an efficient platform for hardware fault emulation because of its re-configurability and the close representation of the real runtime environment [72]. FPGA-based fault injection can be achieved by reconfiguring the FPGA with faulty configuration [73][74][75]. In these types of approaches, faulty configuration bit-streams are generated either from Hardware Description Language (HDL) description or from direct bit-stream modification. For fault injection in HDL, bit-streams generated from HDL require long synthesis and routing time. As for direct bit-stream modification methods, there is no need for re-synthesis and routing time because faults are injected directly into the bit-streams. In general, FPGA-based error injection techniques can be classified into two categories: Instrumentation-based techniques and Reconfiguration-based techniques [67]. In Instrumentation-based techniques, an error-injection unit is synthesized into an FPGA along with the circuit to be tested. The main advantage of these techniques is that no reconfiguration is required during the experiments. In Reconfiguration-based techniques, no error injection hardware is used. Instead, the FPGA is reconfigured every time a new error is injected. The main advantage of these techniques is that there is no hardware overhead. On the other hand, there is a significant amount of time-overhead due to reconfiguration.

In [63] an injection technique using the concept of mask-chain is used to inject SEU-faults. Similarly, in [64] a scan-chain based technique is used for fault injection. Both these methods are unsuitable for error injection into combinational logic since the scan-chain method requires extra hardware in the form of flip-flops for this purpose. In this work, a shift-register based error injection technique is used to inject transient errors into combinational logic. A detailed fault model is created to represent transient pulses of different magnitudes. All transient pulses are

represented as a 5-bit state code as explained in Section 3. This fault model can be used to analyze the effect of voltage pulses of different magnitudes which usually pass undetected when a bit-flip model is used .The SIU consists of a pulse generator circuit which generates a pulse of N_{clk} clock cycles' duration to activate error injection at the first selected location.

The output of the pulse generator is applied to a shift-register consisting of N_f flip-flops where N_f is the number of soft-error locations. The pulse is shifted every N_{clk} clock cycles until all soft-error locations are covered. Test Pattern Counter (TPC) is incremented by one whenever transient injection at all locations is completed. It starts from 0 and goes to (V-1) where V is the number of test patterns for any given benchmark. In Nanometre technologies, the capacitance associated with a node is very small; therefore, a small amount of charge deposited by a radiation strike can result in relatively large voltage disturbances. As mentioned in [65], the voltage swing associated with SETs in 5 volt CMOS technologies is about 14% greater than the normal voltage swing of the node, and thus its impact is very limited in terms of both magnitude and duration. On the other hand, for 3.3 volt technology, the voltage disturbance becomes 21% larger than normal swing. Thus the restoration of charge at the struck node takes a longer time. This shows that the duration of the injected voltage pulse grows with technology scaling and may become comparable with the propagation delay of the gates in modern technologies. This may become a critical issue as the voltage pulse may spread throughout the circuit easily. Therefore, in this thesis, the injected transient pulse is assumed to be wide enough to propagate through a logic circuit. The fault counter increment by one after every test run, which consists of application of V test patterns, is to be a given DUT.

Fault counter counts from 0 to 7 as the number of applied error types is 8 (obtained after applying transient equivalence). In this method, single error injections are used, i.e. only one error is present at any given time. This assumption is relevant because in real circuits the time gap between any two soft-errors is relatively large. In most of the error injection techniques used so far, errors are injected at the inputs of a gate-level circuit [59]. This can lead to wrong results as injection at the input of one gate corresponds to injection of the same error at more than one location at switch-level. For example, if an error is injected at the input of an inverter, it gets injected at the gate of two transistors which is equivalent to multiple bit-flips. This may lead to wrong soft-error coverage calculations as each error may get counted twice. One of the methods to avoid this situation is to use the gate of a switch as the location for error injection.

8.3.3 Observation Unit (OU)

Fast fault detection and recovery can be realized by clock synchronized duplicated systems, which have fast fault detection and recovery features with optimal time diversity. A duplication technique is commonly used to observe the effect of injected faults in processors [66]. This technique can be used in FPGA-based error detection as well [67]. In this technique, a faulty circuit and a fault-free circuit are both implemented on the FPGA, and the outputs of both are compared every N_{clk} clock cycles to observe the effect of the injected fault. The major drawback of duplication technique is high degree of hardware overhead. On the other hand, scan-chain based observation techniques are quite slow due to a trade-off between speed and observability in these techniques. For example, to observe the internal register values at each clock cycle,

requires shifting out all the internal values at every clock cycle which is apparently very time-consuming [64]. In this thesis, a technique similar to the duplication method is used, but hardware overhead is reduced by a significant amount. Since the fault-free (gate-level) circuit is much smaller as compared to a faulty (hybrid) circuit, overhead is less than 100 % (unlike the duplication method). The observation unit receives the outputs of the golden device and DUT_{faulty} and compares their 2-bit logic parts. The output of the observation unit goes high if the results do not match indicating the detection of an error.

8.3.4 Soft-Error Coverage Calculation Unit (SCCU)

SCCU consists of a 16-bit register, error profile storage memories, and Memory Address Counter (MAC) and an "OR" unit. There are 8 error profile storage memories each corresponding to one transient error type. The depth of each memory changes according to the number of the soft-error locations and the width is fixed at 16. All memories are initialized to 0 before starting the experiment. Each stored bit in a particular memory indicates the detection status of a particular error type at a particular location. Each bit corresponds to one soft-error location (LSB corresponds to detection status of transient injected at the first location). After every (N_{clk}*16) clock cycles, the data from register are fed to the "OR" unit. The other input to the "OR" unit consists of the data read from the soft-error coverage storage memory. The result of the "OR" operation is written back to the memory. This process is carried out for each error type. MAC unit generates the address of memory where data are to be written or read from. At the end of the experiment, each error profile storage memory contains the detailed profile of one

transient error type. The stored profile indicates the detection status (“0” for “not detected” and “1” for “detected”) of the corresponding error injected at all the soft-error locations. This detailed profile is used to calculate the soft-error coverage for individual error types as well as the overall soft-error coverage for a given DUT. The collected data can be used to find out the locations inside a given circuit which are most susceptible to soft-errors. There is no runtime communication with any external host involved in this method as the data can be retrieved at the end of experiments. This also contributes to making the emulation faster.

As the FPGA are designed to implement the circuit at gate-level, it is not possible to implement the circuit in switch-level. The creative idea in this chapter made it possible to implement the circuit in switch-level on FPGA by creating some component and define the whole architecture for emulation. The experimental result for both emulation and simulation and the speed up of emulation are all brought in next chapter.

CHAPTER 9
EXPERIMENTAL RESULTS

This chapter will discuss the experimental result achieved for resistive delay faults followed by the simulation and emulation results for soft-errors detections. This chapter also explain the architecture of Mathlab simulation more in detail.

9.1 Resistive Delay Fault Simulation

Simulation results were obtained in two phases. In the first phase, the dynamic effect of resistive faults and the calculation of output voltage was obtained using Cadence Spectra based on the method presented in Chapter 2 for all primary gates; for example, all the resistive values for 3-input NOR gate mentioned in Table 19. The experiment was completed for all CMOS gates at transistor level. In the second phase, Delay Fault Simulation (DFS) software was developed and run on several ISCAS'85 and ISCAS'89 benchmark circuits. This application has three main subroutines that were explained in Chapter 2. The first subroutine converts the circuit to a net-list

to generate the circuit graph. The second part will convert the graph to arithmetic functions. This software is able to convert a whole circuit to a graph and then convert the graph to an arithmetic model. Eventually, the last subroutine will generate and inject random faults by replacing faulty functions in the arithmetic equation. Although this arithmetic model is able to simulate various faults, testing on delay faults was achieved in 65nm technology for the purposes of this thesis. The experimental result for further technologies will be obtained based on the availability of technology.

Table 19 - 65nm - V_{OUT} v/s t_{PLH}, $I_1I_2I_3$ = 000, t_{PLH} in µs, V_{OUT} in volt.

	D_3 to V_{DD}		D_2 to V_{DD}		D_1 to V_{DD}	
R_S (Ω)	t_{PLH}	V_{OUT}	t_{PLH}	V_{OUT}	t_{PLH}	V_{OUT}
15K	0.25	0.83	0.27	0.82	0.25	0.82
10K	0.22	0.83	0.23	0.83	0.24	0.82
5K	0.19	0.85	0.22	0.83	0.23	0.81
1K	0.15	0.98	0.17	0.88	0.23	0.83

As explained earlier in Chapter 2, while injecting a resistive short or open fault in the circuit within a gate, the RC model of the circuit is needed to calculate the voltage charge at the gate output. In this experiment, a range of resistive faults was considered, including 1k, 5k, 10k and 15k. Figure 45 illustrates three types of faults injected into a 3-input NOR gate subjected to an all zero input vector. All three faults have different orientations for their data labels in order to differentiate them from the other fault lines.

The resistive short between PMOS drain, D_3 and power has lower delay value for 5KΩ defect resistance compared to the same configuration for D_2 to power. Similarly, the delay caused due

to 5KΩ defect resistance for the short fault D_1 to V_{DD} is the highest among all three short fault D_i to V_{DD} faults shown in Table 19.

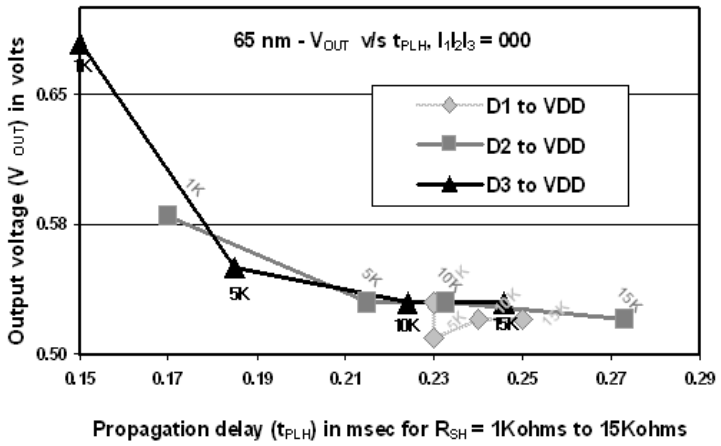


Figure 45 - 65nm , Output voltage v/s Propagation delay; Resistive shorts for $I_1I_2I_3$ = 000.

The closer the fault location to the power source, the higher will be the delay induced into the system due to the resistive effect of the fault. On the other hand, the delay sizes for the same spot with different resistance sizes are almost the same. Table 19 shows all delays and related output voltage for a 3-input NOR gate with resistive faults between drain D1 through D3 and V_{DD} in 65nm technology. Table 19 illustrates that the delay size t_{PLH} and the voltage output V_{OUT} for different resistor sizes RS are almost the same.

To apply MVL9 in simulation software, the voltage range between V_{DD} and Ground is divided into intervals to represent one of the logic levels at MVL9. Eventually, the voltage values generated based on the injected fault and input values (as explained in Table 19) will be assigned a logic value in that interval. For instance, the logic value for fault D_3 to V_{DD} in Table 19 is considered as X (forcing unknown) in MVL9.

Finally, to complete this simulation pseudo-random input test sets are obtained from gate-level test generation tools. As expected, results indicate that gate-level test vector sets detect fewer switch-level faults. In this case, switch-level fault coverage was less than gate-level fault coverage. This result confirms the fact that switching-fault simulation can be a better design verification tool inasmuch as a larger test set would be required to achieve switch-level fault coverage similar to the gate-level fault coverage. Any possible switch-level faults that are undetected for the gate-level test set could be detected by the larger switch-level test set. Fault Coverage (FC) simulation results for resistive short delay faults are shown in Table 20, where several ISCAS'85 and ISCAS'89 benchmark circuits have been used. The number of switches and injected files, and, eventually, the delay fault coverage were calculated using the proposed method.

Table 20 - Fault coverage simulation results for resistive short delay fault.

Circuit	# of Switches	# of Faults	Delay Propagation (%FC)
C17	24	30	100
C2670	6212	1247	62
C7552	18802	2667	58
S27	66	70	98
S298	582	463	63
S1238	2662	1054	51
S13207	30984	4705	43
S38417	85912	9816	37

Test simulation results can be achieved for different technologies and a larger variety of resistors based on the technology.

CPU time and memory requirements depend on both the number of nodes and the number and order of the Boolean variables in the circuit function. Therefore, the analysis of very large

circuits requires embedding variable ordering and partitioning strategies into the analyzer. The simulation ran on a Pentium 4 system (3.0 GHz, RAM =1 GB, OS = Windows XP). Table 21 shows the CPU time based on this system. The unit of measurement is second.

Table 21 - Benchmarks Synthesis CPU Time.

Circuit	# of Switches	CPU time(s)
C17	24	<1
C2670	6212	753
C7552	18802	1235
S27	66	<1
S298	582	585
S1238	2662	938
S13207	30984	2620
S38417	85912	5390

Although currently there is no similar research publication to compare in deep submicron, the method presented here has a noticeable fault coverage and CPU time.

9.2 Experimental Result for Soft-Error Detection

The emulation result for strength violation effect on soft-error detection is presented in this section. The experimental result is divided into main parts. First part of experiment is the implementation of proposed methodology for soft-errors in customized MATLAB simulation. The second part of the experiment was carried on XC5VFX130T-2FF1738 Xilinx FPGA using ML510 embedded system development board. The transient errors based on the previously explained strength scaling technique were injected into ISCAS‘85 benchmark series for two part

of experiment. To compare the amount of speed-up achieved by the presented emulation method with simulation techniques, two sets of experiments were carried out.

9.2.1 Customized MATLAB Simulation Environment

Instead of using state-of-the-art simulators like Modelsim and Xilinx ISIM, a customized simulation environment was created using MATLAB which was discussed fully in Chapter 6. MATLAB based simulation is relatively faster than other simulators as it is optimized for high speed applications. For example, transient injection into C499 with 52 primary input values achieved a speed-up of 6.0 for MATLAB based simulation as compared to simulation using ISIM. Both simulations were run on the same computer and timings were calculated based on CPU time. In this technique, switch-level circuits were created using C Function, PMOS Function and NMOS Function. Simulations were run for different specifications by varying the number of soft-error locations, number of primary input values, and number of injected transient errors. Simulation results were processed to calculate the soft-error coverage, and the total elapsed time was recorded for each benchmark circuit. MATLAB simulation structure is shown in Figure 46 which consists of several classes. The definition of each class is described as follows:

9.2.1.1 Parser Class

Parser class in Figure 46 takes the gate-level net-list of the circuit under test as input as well as the set of test vectors in a text format. These text files are parsed into a data structure.

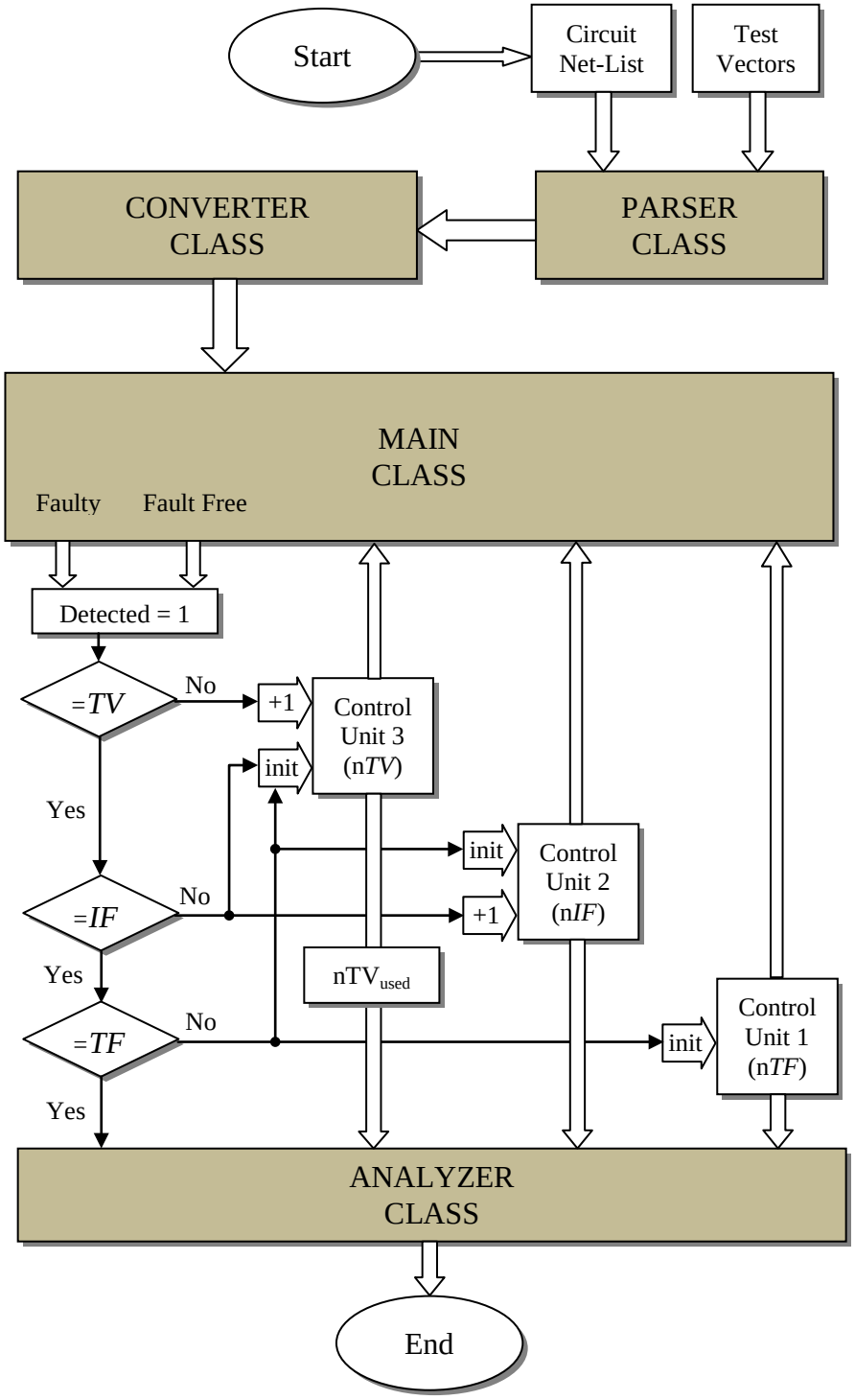


Figure 46 - Simulation Flow Diagram.

9.2.1.2 Converter Class

In Figure 46 converter class will code and convert the input data structures, which are coming from parser into the switch-level model of the circuit under test.

9.2.1.3 Main Class

Main class is the main program where the modeled circuit is loaded to be processed for simulation. The parameters of the simulation are controlled mainly by the following blocks:

- *Control Unit 1*: Control unit 1 controls and injects the fault type, where "*nTF*" is the total number of faults.
- *Control Unit 2*: Control unit 2 selects the injection location, where "*nIF*" is the total number of locations.
- *Control Unit 3*: Control unit 3 controls the test vectors, where "*nTV*" is the total number of test vectors.

These blocks are incremented or initialized by the conditional status of the output of main class. The main program ejects the test vectors when a fault is detected in order to minimize the number of iterations for fault detections. nTV_{used} represents the number of used test vectors to detect the specific fault type at the specific location

9.2.1.4 Analyzer Class

Analyzer class processes the output of the simulation and prepares the results for analysis. The arrays generated by this module allow the user to locate the detected faults as well as their corresponding test patterns. These are provided as statistical results as they can indicate the weak location of the circuit and the corresponding test vectors when a fault is detected.

9.2.2 Switch-level Emulation

In this set of experiments, the presented emulation architecture was evaluated by injecting transient pulses into ISCAS‘85 benchmark suite. Different test runs were performed by varying the input parameters. To observe the effect of transient equivalence on emulation time, the emulation time for all 23 types of transient errors was injected into the switch-level of all benchmarks for one input value and then the number of transients was reduced to 8. The emulation times for both test runs were compared to evaluate the speed-up achieved by applying the concept of transient equivalence. An average speed-up of 2.875 was achieved in emulation time by using transient equivalence. This also resulted in lower memory utilization as only 2192 bytes of memory were used to store error profiles instead of 6302 bytes. This amounted to almost 65% reduction of size of error profile storage memory.

Table 22 shows the comparison between simulation-time for MATLAB based simulations and FPGA-based technique. Emulation time includes the time required for error injection as well

as the processing time needed to create the profiles of each error type. The proposed emulation technique showed an average speed-up of the order of 10^6 as compared to simulation. The timings given in the table are based on the injection of transient errors after applying the concept of equivalence in simulation as well as emulation using one input value for comparison. As the size of the circuit increased, the amount of speed-up achieved also increased. For example, for C17 the magnitude of speed-up achieved was of the order of 10^4 while for the biggest benchmark c7552, it was 10^6.

Table 22 - Emulation vs. Simulation.

Circuits	Total injected Faults	Simulation Time(s)	Emulation Time(s)	Speed-Up
C17	1639	0.3945	6E-06	65750
C432	132479	481.19	0.0007	687412
C499	533853	4360.01	0.0014	1406435
C880	157849	2844.35	0.0018	1580198
C1908	1112537	7075.6	0.0033	2144122
C2670	1121814	19208.96	0.0054	3557216
C3540	2018866	34002.08	0.0072	4722511
C5315	1555919	76739.59	0.0108	7105518
C6288	2517877	125404.19	0.0138	9087260
C7552	3456606	143917.16	0.0148	9724133

Figure 47 shows that the growth of speed-up achieved by emulation with the circuit size was almost linear. The complete switch-level circuits of some of the benchmarks were too large to fit into the FPGA. The emulation time for these circuits is calculated using equation (8.1).

$$T_{EM} = 8 \times N_{clk} \times N_{loc} \times N_{test} \quad (8.1)$$

Where N_{clk} is the number of clock cycles needed to process one location for one input value for any given benchmark, N_{loc} is the number of soft-error locations. N_{test} is the number of test patterns. The multiplication factor 8 indicates the total number of error types. As equation (8.1) shows, emulation time is independent of the number of inputs and outputs of the circuit as all data are processed in parallel.

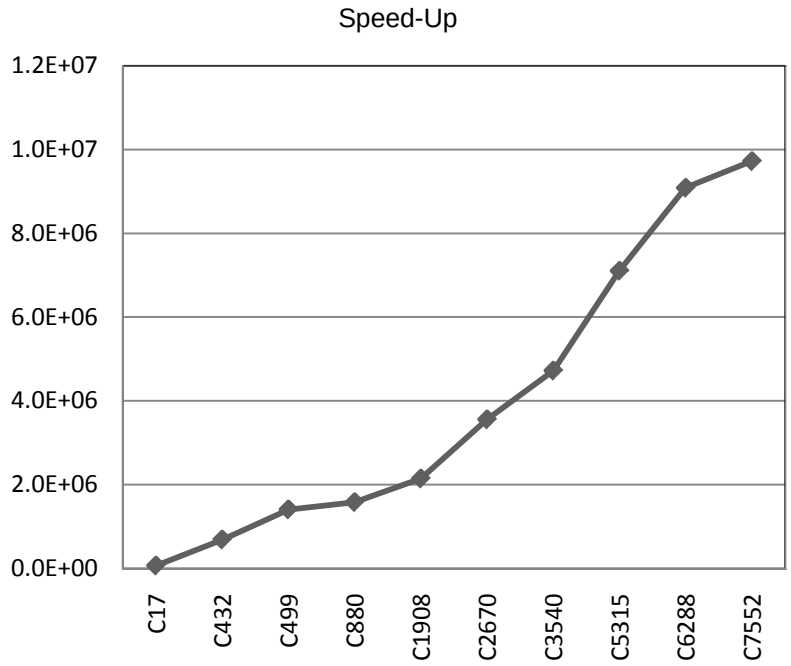


Figure 47 - Emulation speed-up for ISCAS‘85 benchmark.

This makes the presented emulation technique suitable for even complex circuits. Emulation time (T_{EM}) is purely based on the time required to inject transient pulses, process and extract soft-error coverage data from a given configuration of the circuit. It does not include the synthesis time and reconfiguration time for different reconfiguration files needed to cover all the transient error locations in larger circuits.

Figure 48 shows the soft-error fault detection coverage obtained by presenting a strength-scaling based soft-error detection method using gate and drain as transient injection location. An average coverage of about 0.7 - 0.88 was achieved for injection at gate and drain which further solidifies the presented idea about soft-errors resulting due to transient pulses of strength lesser than logic threshold.

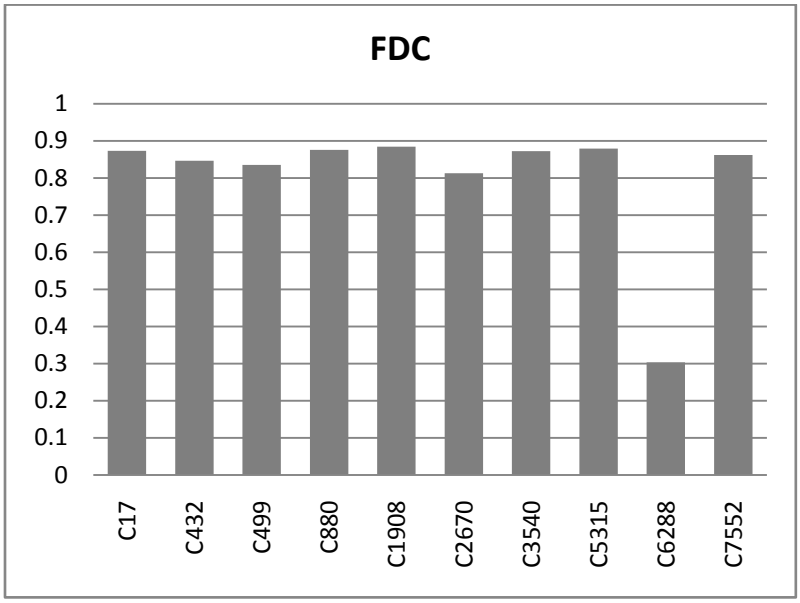


Figure 48 - FPGA-Base Soft-Error Fault Detection Coverage for Strength Violation.

As expected, benchmark C6288 showed lesser soft-error coverage than other benchmarks. The major reason for this deviation was that C6288 is a 16x16 multiplier, and most of the transient errors get logically masked by the other no-faulty inputs. The number of soft-error locations is twice of the number transistors in the circuit as transient errors were injected at each gate and drain of the switch-level circuit.

CHAPTER 10
CONCLUSION AND FUTURE WORK

This thesis presents the analysis, design and implementation of fault detection in Nanometre Technology. An algorithm was proposed for circuit fault diagnosis in deep sub-micron technology to propagate the delay faults that lead to logic faults at primary outputs. This approach propagates the fault from the fault site by mapping a nine-valued voltage model on top of a five-valued voltage model. As the soft-errors manifest the effect of delay faults, an algorithm was proposed to detect soft-errors at the switch-level caused by current spikes which can affect the driving strength. It also presented the analysis, design and implementation of strength violation effect on soft-errors for the first time. The offered approach uses a novel coding system to be applied in certain functions that are sensitive to strength variations. Most soft-error detection techniques sense the logic changes in the circuit whereas this approach proved that a wide range of soft-errors are actually the result of strength violations at switch-level.

10.1 Resistive Delay Faults

This research showed that some of the delay faults are the result of resistive open and short defects in the circuit. These defects are inevitable during the fabrication stage. It was proven that some of the resistive short and open defects are not causing permanent stock-at-fault problems; however, they may delay the signal at the switch-level and eventually generate a logic fault at the output of the circuit that latches on a memory element. A switch-level algorithm was presented for delay faults, and its applicability to robust delay fault testing was demonstrated. This algorithm can be implemented for varied applications, i.e. switch-level min-max mode grading of robust/non-robust delay test vectors and analysis of dynamic hazards for delay fault diagnosis for general switch-level circuits. These applications will be explored in future research.

The proposed algorithm for resistive delay faults was implemented in two phases for diagnosing and propagating gate-delay faults in deep sub-micron technology.

In phase one the circuit converted to a resistive model, and the accurate time and voltage disturbance at gate output were calculated.

In phase two, the gate-delay fault is propagated to the primary output using a powerful arithmetic model of the circuit in transistor level. Even when the delay size is relatively small, the algorithm maps a MVL9 on top of a MVL5 in order to propagate those delays that are not causing logical failure.

To implement these two phases, Cadence tools were used for accurate testing of time and voltage disturbances in the circuit. Delay Fault Simulation (DSF) software was also developed to inject faults into the circuit and measure the fault coverage.

10.2 Delay Faults and Soft-Errors

The continuous shrinking of VLSI devices and the fast increase in chip clock rates raise the challenging problem of ensuring that designs are meeting performance and reliability specifications. It has been widely observed that chips are increasingly susceptible to delay defects and soft-errors, both more difficult to deal with using a manufacturing test compared to traditional stuck-at faults. These potential reliability problems are becoming increasingly critical due to the aggressive technology scaling and design style. Additional test resources mitigate these problems only to a limited extent, while incurring a much larger testing cost, a quite unpalatable trade-off from a test economics point of view. The solution to these challenges necessitates the incorporation of reliability-oriented design techniques and economic models that guide their applications in product development.

Although extensive work has been conducted on fault tolerance circuitry design, little research has been proposed to address the issue of concurrent handling of delay defects and soft-errors. The challenge of this problem stems from the need for integrating diversified fault-tolerance schemes in a low cost and coherent manner. For example, these faults may either generate noise at the input of a flip-flop or directly corrupt its internal node signal. Hence, an appropriate fault-tolerance scheme needs to simultaneously address distinct fault behaviours.

Moreover, such a scheme must differ from traditional schemes in the sense that it should not incur any timing pressure which essentially tampers with the delay fault-tolerance capability.

10.3 Strength Variation Effect on Soft-Error Detection

A novel concept of soft-errors resulting from voltage pulses of different magnitudes for transient injection on soft-error and accuracy of simulation/emulation was presented. As expected, a high rate was achieved from switch-level analysis of soft-errors in the circuit. This research proves that a wide range of soft-errors are the result of strength violation in switch-level. Simulation results show an average rate between ‘0.7’ and ‘0.88’. The presented work has allowed the authors to think beyond the normal logic change based on soft-error detection techniques currently used and has opened a door towards the design of more precise and efficient soft-error detection algorithms.

The proposed emulation technique opens a new research area in the field of soft-error detection. It uses a new fault model based on soft-errors produced by induced voltage pulses of variable magnitudes. Some of the other contributions of this paper can be described as follows:

This research presents a novel soft-error model based on signal strength variations at switch-level. The switch-level emulation system uses a novel soft-error detection which can detect soft-errors resulting due to transient pulses of variable strengths. An innovative soft-error coverage

calculation method is presented which does not require communication with any external device, and hence results in the faster processing of collected data.

The data collected from different sets of experiments conducted on ISCAS‘85 benchmarks circuits’ show speed-up of 2.875 in emulation and memory saving of about 65%. FPGA-based emulation using the proposed method is about 10^6 times faster than simulation-based soft-error detection.

10.4 Future Work

As this thesis is devoted to a pioneering work in the field of resistive delay faults and strength-based soft-error simulation and emulation, it has a lot of potential for future research. The resistive delay fault detection technique can be further extended by applying Verilog strength for MVL5 and MVL9 for higher accuracy. On the other hand, strength-based soft-error detection has been discussed for the first time in this research and can be used as an accuracy technique for all the existing soft-error detection or soft-error rate calculation methods. It can also be applied for sequential circuits like ISCAS’89 benchmark as an extension to this research.

Furthermore the speed of emulation can be improved by exploring sensitivity of internal nodes to all 23 types of transient errors presented in this research. This is an interesting area of research and may result in reducing the number of transient injection locations by eliminating redundant nodes and lowering the area overhead. Although this research is based on the single bit-flip model, the technique can be extended to multi bit-flip soft-error models.

PUBLICATIONS

Referred Journals:

1. **Reza Javaheri,** Reza Sedaghat, Prabhleen K. Kalkat, “Switch-level soft error emulation for SET-induced pulses of variable strengths”, Microelectronics Journal Elsevier, (Accepted 14 June 2010. Available online 9 July 2010

2. **Reza Javaheri,** Reza Sedaghat, “Strength Violation Effect on Soft-Error Detection in Sub-micron Technology” International Journal of Microelectronics Reliability, Science Direct, Elsevier, Volume 50, Issue 7, pp. 971-977, 2010

3. **Reza Javaheri,** R. Sedaghat, "Multi-valued Logic Mapping of Resistive Short and Open Delay Fault Testing in Deep Sub-Micron”, International Journal of Microelectronics Reliability Elsevier, Volume 49, Issue 2, pp. 178–185, February 2009

4. **Reza Javaheri,** R. Sedaghat, Leo Kant, Jason Zalev, “Verification and Fault Synthesis Algorithm at Switch-Level”, Journal of Microprocessors and Microsystems, Science Direct, Elsevier, Volume 30, Issue 4, pp. 199-208, June 2006

5. **Reza Javaheri,** R. Sedaghat, “Bi-directional Switch-Level Verification and Multiple Fault Synthesis Algorithm”, WSEAS Transactions on Circuit and Systems, pp. 88-101, June 2006

6. R. Sedaghat, M. Kunchwar, R. Abedi, **Reza Javaheri,** “Transistor-level to Gate-level Comprehensive Fault Synthesis for n Input Primitive Gates”. International Journal of Microelectronics Reliability, Science Direct, Elsevier, Volume 46, Issue 12, pp. 2149-2158, December 2006

Submitted Referred Journals:

7. **Reza Javaheri,** Reza Sedaghat, “Switch-Level Emulation of Strength Violation Effect on Soft-Error Detection in Nanotechnology”, *IEEE Transactions on Computers (TC)*. (Submitted 2010)

8. **Reza Javaheri,** Reza Sedaghat, Prabhleen K. Kalkat, J. M. Chikhe, “Soft-error Detection using Reconfigurable Hardware at Switch-Level for SET-Induced Pulses of Variable Strengths’ International Journal of Microelectronics Reliability, Science Direct, Elsevier, (Second revision, 2010)

9. **Reza Javaheri,** Reza Sedaghat, Prabhleen K. Kalkat, 'Strength Violation Effect on Soft-Error Detection in Switch-Level', *ACM Transactions on Design Automation of Electronic Systems*, (Submitted, 2010)

Referred Conferences:

10. **Reza Javaheri**, R. Sedaghat, 'Dynamic Strength Scaling for Delay Fault Propagation in Nanometer Technologies' *IEEE 14th International CSI Computer Conference (CSICC)*, pp. 95-99, December 2009

11. Prabhleen K. Kalkat, Reza Sedaghat, Jalal Mohammad Chikhe, **Reza Javaheri**, 'Soft-Error Injection using Advanced Switch-Level Models for Combinational Logic in Nanometer Technologies' *IEEE 21st International Conference on Microelectronics (ICM), Published in the 21st Conference Proceedings*, pp. 326-329, December 2009

12. **Reza Javaheri**, R. Sedaghat, "A Novel Delay Fault Testing Methodology for Resistive Faults in Deep Sub-micron Technologies ", *Springer Verlag, International CSI Computer Conference (CSICC),* pp. 177-182, November 2008

REFERENCES

[1] Almukhaizim, S., Feng Shi, Love, E., Makris, Y., “Soft-Error Tolerance and Mitigation in Asynchronous Burst-Mode Circuits” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Volume: 17 , Issue: 7, pp.869 – 882, 2009

[2] Munteanu, D., Autran, J.-L., “Modeling and Simulation of Single-Event Effects in Digital Devices and ICs “, IEEE Transactions on Nuclear Science, Volume: 55 , Issue: 4, pp.1854 – 1878, 2008

[3] Kim, S., Iyer, R. K., “Impact of Device Level Faults in a Digital Avionic Processor”, Proc. AIAA/IEEE 8th Digital Avionics Systems Conference (DASC), pp.428-436, Oct 17-20, 1988

[4] Sanyal, A., Ganeshpure, K., Kundu, S., “An Improved Soft-Error Rate Measurement Technique “, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems Volume: 28 , Issue: 4, pp.596 – 600, 2009

[5] Gilson I. Wirth, Michele G. Vieira, Egas H. Neto , Fernanda Lima Kastensmid,”Modeling the sensitivity of CMOS circuits to radiation induced single event transients”, Microelectronics Reliability 48, pp.29–36, 2008.

[6] Balkaran S. Gill, Chris Papachristou, Francis G. Wolff, “Soft Delay Error Effects in CMOS Combinational Circuits”, Proceedings of the 22nd IEEE VLSI Test Symposium, 2004

[7] N. Weste and K. Eshraghian, Principles of CMOS VLSIDesign, Addison-Wesley, 1993

[8] Abramovici M., Breuer, M.A., and Friedman, A.D., “Digital System Testing and Testable Design”, Revised edition IEEE Press, 1995

[9] Favalli, M., Metra, C., “Testing Resistive Opens and Bridging Faults Through Pulse Propagation” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Volume: 28 , Issue: 6, pp.915 – 925, 2009

[10] Garcia-Gervacio, J.L., Champac, V., Detectability analysis of small delays due to resistive opens considering process variations”, 15th IEEE International On-Line Testing Symposium, pp.195 – 197, 2009

[11] Heijmen, T., Nieuwland, A, “Soft-Error Rate Testing of Deep-Submicron Integrated Circuits “. Eleventh IEEE European Digital Object Identifier, pp. 247 – 252, 2006

[12] C. M. Hsieh, P. C. Murley, and R. R. O’Brien, “Dynamics of charge collection from alpha-particle tracks in integrated circuits,” in Proc. IRPS, pp.38-42, 1981

[13] 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 Proc. Int’l conf. Dependable Systems and Networks, pp. 389-398, 2002

[14] H. T. Nguyen, and Y. Yagil, “A systematic approach to SER estimation and solutions,” in Proc. 41st Annual Int’l Reliability Physics Symposium, pp. 60-70, 2003

[15] I. Polian, J. P. Hayes, S. Kundu, and B. Becker, “Transient fault characterization in dynamic noisy environments,” in Proc. Int’l Test Conf., pp. 40.1-40.10 , 2005

[16] Sanyal, Alodeep, Ganeshpure, Kunal, Kundu, Sandip, “On Accelerating Soft-Error Detection by Targeted Pattern Generation”, ISQED '07. 8th International Symposium, pp. 723 – 728, 26-28 March 2007

[17] H. Kobayashi, H. Usuki, K. shiraishi, H. Hiroo Tsuchiya, N. Kawamoto, G. Merchant, and J. Kase, “Comparison between neutron-induced system-SER and accelerated-SER in SRAMs,” in Proc. Int’l Reliability Phys. Symp., pp. 288-293, 2004

[18] N. Seifert, X. Zhu, and L. W. Massengill, “Impact of scaling on soft-error rates in commercial microprocessors,” IEEE Trans. On Nuclear Science, Vol. 49, No. 6, pp. 3100-3106, 2002

[19] Ziegler, J. F. and W. A. Lanford, "Effect of Cosmic Rays on Computer Memories", Science, 206, 776, 1979

[20] J.F. Ziegler, “Terrestrial cosmic rays”, IBM Journal of Research and Development, Vol. 40, no. 1, pp. 19-40, Jan 1996

[21] R. Baumann, T. Hossain, S. Murata, H. Kitagawa, “Boron compounds as a dominant source of alpha particles in semiconductor devices”, IRPS Proceedings, pp. 297-302, 1995

[22] I. Polian, P. Engelke, B. Becker., “Efficient bridging fault simulation of sequential circuits based on multi-valued logics”, International Symposium on Multi-Valued Logic, pp. 216–222, 2002

[23] Prithviraj Banerjee and Jacob A. Abraham., “A multivalued algebra for modeling physical failures in MOS VLSI circuits”, IEEE Trans. on CAD, pp. 312–321, 1985

[24] Thomas M. Storey and Wojciech Ma., “CMOS Bridging Fault Detection”, Int’l Test Conference, pp. 842 – 851, 1990

[25] James Chien-Mo Li, E.J. McCluskey, “Diagnosis of resistive-open and stuck-open defects in digital CMOS ICs”, Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on Volume 24, Issue 11, pp. 1748 – 1759, Nov. 2005

[26] Renovell, M., Azais, F., Bertrand, Y. “Improving defect detection in static-voltage testing”, Design & Test of Computers, IEEE Volume 19, Issue 6, pp. 83 – 89, Nov.-Dec. 2002

[27] R. Sedaghat, M. Kunchwar, R. Abedi, R. Javaheri, “Transistor-level to Gate-level Comprehensive Fault Synthesis for n Input Primitive Gates”, Journal of Microelectronics Reliability, Volume 46, Issue 12, pp. 2149-2158, December 2006

[28] M. Kunchwar, R. Sedaghat, “Dynamic Behavior of Resistive Faults in Nanometer Technology” Microelectronics Reliability, Volume 47, Issue 12, pp. 2141-2146, December 2007

[29] Marcello Dalpasso, Michele Favalli, Piero Olivio, Bruno Riccò. “Parametric Bridging Fault Characterization for the Fault Simulation of Library-Based IC’s”, ITC, pp. 486-495, 1992

[30] R. Jacob Baker. “CMOS Circuit Design, Layout, and Simulation”, Second Edition, IEEE Press 2005

[31] Polian, I., Engelke, P., Becker, B., Kundu, S., Galliere, J.-M. and Renovell, M., Resistive “Bridge fault model evolution from conventional to ultra deep submicron”, VTS, Proceedings. 23rd IEEE, pp. 343 – 348, May 2005

[32] M. Reza Javaheri, R. Sedaghat, Leo Kant, Jason Zalev, “Verification and Fault Synthesis Algorithm at Switch-Level”, Journal of Microprocessors and Microsystems, Science Direct, Elsevier, Volume 30, Issue 4, pp. 199- 208, 6 June 2006

[33] Kohyama, S., and Sate, T., “CMOS technologies for VLSI circuits”, IEEE Conf. on VLSI, pp. 24-25, 1981

[34] Krishnaswamy, V., Casas, J., Tetzlaff, T., “A Switch Level Fault Simulation Environment”, Design Automation Conference Proceedings. 37th (2000), pp. 780-785, 2000

[35] Verilog Hardware Descriptor Language Reference Manual (LRM) DRAFT, IEEE 1364, April 1995

[36] Dahlgren, P., Liden, P., “Efficient modeling of switch-level networks containing undetermined logic node states”, Proceedings IEEE/ACM International Conference on CAD. pp. 746–752, 1993

[37] Bryant, R.E., “A switch-level model and simulator for MOS digital systems”, IEEE Transactions Computers C-33 2, pp. 160–177, 1984

[38] Al-Khalili, D., N-Rozon, C., B. Show, D., “Fault security analysis of CMOS VLSI circuits using defect-injectable VHDL models”, Elsevier Integration, the VLSI journal 32, pp. 77–97, 2002

[39] Leveugle, R., “A low-cost hardware approach to dependability validation of IPs”, IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, pp. 242–249, 2001

[40] Cheng, K.T., Huang S.Y., and Dai, W.J., “Fault emulation: a new methodology for fault grading”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 18 – 10, pp. 1487–1495, 1999

[41] Benso, A., Rebaudengo, M., Impagliazzo, L., Marmo, P., “Fault-list collapsing for fault injection experiments, RAMS'98”, Annual Reliability and Maintainability Symposium, pp. 383–388, 1998

[42] Johnson, B.W., “Design and analysis of fault-tolerant digital systems”, Addison –Wesley, Chapter 2, 1989

[43] Antoni, L., Leveugle, R., Feher, B., “Using run-time reconfiguration for fault injection in hardware prototypes”, IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, pp. 405–413, 2000

[44] Leveugle, R., “Fault Injection in VHDL descriptions and emulation”, IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, pp. 414–419, 2000

[45] J. G. Dastidar, and N. A. Touba., “A Systematic Approach for Diagnosing Multiple Delay Faults”, in Proc. of the IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, pp. 211-216, 1998

[46] Z Wang, K. -H. Tsai, M. Marek-Sadowska and J. Rajski., “An Efficient and Effective Methodology on the Multiple Fault Diagnosis”, In proc. of the International Test Conference, pp. 329-338, 2003

[47] Z. Wang, M. Marek-Sadowska, K. -H. Tsai, and J. Rajski., “Multiple fault diagnosis using n-detection tests”, In Proc. Of the 21st International Conference on Computer Design, pp. 198 – 201, October 2003

[48] Alt, J., Mahlstedt, U., “Simulation of non-classical faults on the gate level fault modeling”, Institute fur Theoretische Elektrotechnik Universitat Hannover, Germany, 11th VLSI Test Symposium, pp. 351 – 354, April 1993

[49] L. Anghel and M. Nicolaidis., “Cost reduction and evaluation of a temporary faults detecting technique”, Design Automation and Test in Europe Conference and Exhibition, pp. 591–598, 2000

[50] P. Hazucha, K. Johansson, and C. Svensson., “Neutron induced soft errors in cmos memories under reduced bias”, IEEE Transactions on Nuclear Science, 45(6) pp. 2921–2928, 1998

[51] Gill, B.S., Papachristou, C., Wolff, F.G., “Soft Delay Error Effects in CMOS Combinational Circuits”, VLSI Test Symposium Proceedings. 22nd IEEE Digital Object Identifier, pp. 325 – 330, 2004

[52] P. Shivkumar, M. Kistler, S. W. Keckler, D. Burger, and L. Alvisi. “Modeling the effects of technology trends on the soft error rate of combinational logic”, Dependable Systems and Networks, Proceedings International, pp. 389–398, 2002

[53] A. Dharchoudhury, S. M. Kang, H. Cha, J. H. Patel, “Fast timing simulation of transient faults in digital circuits”, in Proc. Int. Conf. Computer-Aided Design, San Jose, CA, pp. 719–726, 1994

[54] Neil Weste, David Harris : “CMOS VLSI Design”, A Circuits and Systems Perspective, Addison Wesley, ISBN: 0-321-14901-7

[55] Tino Heijmen and Bram Kruseman, “Alpha-particle-induced SER of embedded SRAMs affected by variations in process parameters and by the use of process options”, Science Direct, Solid State Electronics, No. 49, pp. 1783-1790, 2005

[56] Quming Zhou, Mohanram, K., “Gate sizing to radiation harden combinational logic” Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on Volume 25, Issue 1, pp. 155 - 166 Jan. 2006

[57] Reza Javaheri, Reza Sedaghat, “Dynamic Strength Scaling for Delay Fault Propagation in Nanometer Technologies” 14th International CSI Computer Conference (CSICC), pp. 95 – 99, 2009

[58] Dahlgren P, Liden P, "A switch-level algorithm for simulation of transients in combinational logic," 25th International Symposium on Fault-Tolerant Computing, FTCS-25. Digest of Papers, vol., no., pp. 207-216, June 1995

[59] S. Kundu, M.D.T. Lewis, I. Polian, B. Becker.” A soft error emulation system for logic circuits”, Conference on Design of Circuits and Integrated Systems, Page: 137, 2005

[60] Dodd P.E, Sexton F.W, Hash G.L, Shaneyfelt M.R, Draper B.L, Farino A.J, Flores R.S, "Impact of technology trends on SEU in CMOS SRAMs," IEEE Transactions on Nuclear Science , vol.43, no.6, pp. 2797-2804, December 1996

[61] K. J. Hass, J. W. Gambles, “Single event transients in deep submicron CMOS”, IEEE 42nd Midwest Symposium on Circuits and Systems, vol. 1, pp. 122-125, 1999

[62] Hamzaoglu, I. Patel, J.H. “Test set compaction algorithms for combinational circuits” Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on Volume 19, Issue 8, pp. 957 – 963 Aug. 2000

[63] Ejlali A, Miremadi SG, “FPGA-based fault injection into switch-level models”, Journal of Microprocessors and Microsystems, Elsevier Science, 28(5–6): pp. 317–27, April 2004

[64] P. Civera, L. Macchiarulo, M. Rebaudengo, M. Sonza Reorda, M. Violante, “Exploiting circuit emulation for fast hardness valuation”, IEEE Transactions Nuclear Science 48. pp. 2210–2216, 2001

[65] K. J. Hass, J. W. Gambles, “Single event transients in deep submicron CMOS”, IEEE 42nd Midwest Symposium on Circuits and Systems, pp. 122-125, 1999

[66] Carreira J, Madeira H, Silva J, Xception, “Software fault injection and monitoring in processor functional units”. In Conference on dependable computing for critical applications (DCCA-5), pp. 135–149, September 1995

[67] Prabhleen Kalkat, Reza Sedaghat “Switch-Level Soft-error emulator of single event transient-induced pulses of variable strength”, Master of science thesis report, Ryerson University, 2010

[68] Peter Ming-Han Lee, Reza Sedaghat, “FPGA-based switch-level fault emulation using module-based dynamic partial reconfiguration” Microelectronics Reliability Volume 48, Issue 10, Pages 1724-1733, October 2008

[69] Hwang SA, Hong JH. “Sequential circuit fault simulation using logic emulation”, IEEE Trans Computer-Aided Design Integrated Circ Syst. 17(8), pp. 724–36, 1998

[70] Madeira H, Rela M, Silva JG. “A general purpose pin-level fault injector”, In Proceedings of first European dependable computing conference, pp. 199–216, 1994

[71] Chakraborty TJ, Chiang CH. “A novel fault injection method for system verification based on FPGA boundary scan architecture”, In Proceedings of international test conference, pp. 923–929, 2002

[72] Burgun L, Reblewski F, Fenelon G, Barbier J, Lepape O. “Serial fault emulation”, In Proceedings of the 33rd IEEE design automation conference, pp. 801–806, 1996

[73] Antoni L, Leveugle R, Feher B. “Using run-time reconfiguration for fault injection applications”, IEEE Trans Instrum Measurement, 52(5) pp. 1468–73, 2003

[74] Leveugle R. “Towards modeling for dependability of complex integrated circuits”, IEEE international on-line testing workshop, pp. 194-198, 1999

[75] Antoni L, Leveugle R, Feher B. “Using run-time reconfiguration for fault injection in hardware prototypes” In: Proceedings of the 17th IEEE international symposium on defect and fault tolerance in VLSI systems, pp. 245-253, 2003

APPENDIX

The simulation results for all ISCAS'85 benchmarks are shown in the following tables.

Simulation result for C17 circuit:

Table 8 - C17 fault types and relevant SER.

c17	Logic	Strength	Faults	SER
1	01	111	77	1
2	00	000	129	0.958333
3	00	001	144	1
4	00	010	253	0.75
5	00	011	253	0.75
6	00	100	253	0.75
7	00	101	253	0.75
8	00	110	253	0.75
9	01	000	129	0.958333
10	01	001	211	0.916667
11	01	010	290	0.708333
12	01	011	290	0.708333
13	01	100	290	0.708333
14	01	101	290	0.708333
15	01	110	290	0.708333
16	11	000	129	0.958333
17	11	001	110	1
18	11	010	110	1
19	11	011	110	1
20	11	100	110	1
21	11	101	110	1
22	11	110	110	1
23	11	111	57	1
Total			4251	0.873188

Simulation result for C432, 27-channel interrupt controller circuit:

Table 9 - C432 fault types and relevant SER.

c432	Logic	Strength	Faults	SER
1	01	111	5846	1
2	00	000	11501	0.920759
3	00	001	11663	0.970424
4	00	010	22229	0.722098
5	00	011	22229	0.722098
6	00	100	22229	0.722098
7	00	101	22229	0.722098
8	00	110	22229	0.722098
9	01	000	11501	0.920759
10	01	001	16980	0.864397
11	01	010	25457	0.657366
12	01	011	25457	0.657366
13	01	100	25457	0.657366
14	01	101	25457	0.657366
15	01	110	25457	0.657366
16	11	000	11501	0.920759
17	11	001	6623	0.979353
18	11	010	6623	0.979353
19	11	011	6623	0.979353
20	11	100	6623	0.979353
21	11	101	6623	0.979353
22	11	110	6623	0.979353
23	11	111	2573	1
Total			349733	0.842197

Simulation result for C499, 32-Bit Single-Error-Correcting Circuit:

Table 10 - C499 fault types and relevant SER.

c499	Logic	Strength	Faults	SER
1	01	111	21901	0.987385
2	00	000	43189	0.923853
3	00	001	50366	0.933945
4	00	010	96178	0.692661
5	00	011	96178	0.692661
6	00	100	96178	0.692661
7	00	101	96178	0.692661
8	00	110	96178	0.692661
9	01	000	43189	0.923853
10	01	001	66245	0.863532
11	01	010	106227	0.655275
12	01	011	106227	0.655275
13	01	100	106227	0.655275
14	01	101	106227	0.655275
15	01	110	106227	0.655275
16	11	000	43189	0.923853
17	11	001	27295	0.970872
18	11	010	27295	0.970872
19	11	011	27295	0.970872
20	11	100	27295	0.970872
21	11	101	27295	0.970872
22	11	110	27295	0.970872
23	11	111	8779	1
Total			1452653	0.831362

Simulation result for C880, 8-Bit ALU circuit:

Table 11 - C880 fault types and relevant SER.

c880	Logic	Strength	Faults	SER
1	01	111	7560	1
2	00	000	13251	0.963097
3	00	001	16115	0.987791
4	00	010	26165	0.743896
5	00	011	26165	0.743896
6	00	100	26165	0.743896
7	00	101	26165	0.743896
8	00	110	26165	0.743896
9	01	000	13251	0.963097
10	01	001	17693	0.949778
11	01	010	27351	0.720866
12	01	011	27351	0.720866
13	01	100	27351	0.720866
14	01	101	27351	0.720866
15	01	110	27351	0.720866
16	11	000	13251	0.963097
17	11	001	9364	1
18	11	010	9364	1
19	11	011	9364	1
20	11	100	9364	1
21	11	101	9364	1
22	11	110	9364	1
23	11	111	4484	1
Total			409369	0.876116

Simulation result for C1908, 16-bit SEC/DED circuit:

Table 12 - C1908 fault types and relevant SER.

c1908	Logic	Strength	Faults	SER
1	01	111	37142	1
2	00	000	77397	0.98462
3	00	001	83892	0.999565
4	00	010	250241	0.748549
5	00	011	250241	0.748549
6	00	100	250241	0.748549
7	00	101	250241	0.748549
8	00	110	250241	0.748549
9	01	000	77397	0.98462
10	01	001	118821	0.972867
11	01	010	271284	0.734185
12	01	011	271284	0.734185
13	01	100	271284	0.734185
14	01	101	271284	0.734185
15	01	110	271284	0.734185
16	11	000	77397	0.98462
17	11	001	50464	0.99971
18	11	010	50464	0.99971
19	11	011	50464	0.99971
20	11	100	50464	0.99971
21	11	101	50464	0.99971
22	11	110	50464	0.99971
23	11	111	18038	1
Total			3400493	0.88427

Simulation result for C2670, 12-bit ALU and controller circuit:

Table 13 - C2670 fault types and relevant SER.

c2670	Logic	Strength	Faults	SER
1	01	111	49834	0.965949
2	00	000	98043	0.893613
3	00	001	99000	0.926694
4	00	010	220673	0.642025
5	00	011	220673	0.642025
6	00	100	220673	0.642025
7	00	101	220673	0.642025
8	00	110	220673	0.642025
9	01	000	98043	0.893613
10	01	001	119505	0.886909
11	01	010	235582	0.621736
12	01	011	235582	0.621736
13	01	100	235582	0.621736
14	01	101	235582	0.621736
15	01	110	235582	0.621736
16	11	000	98043	0.893613
17	11	001	43349	0.987032
18	11	010	43349	0.987032
19	11	011	43349	0.987032
20	11	100	43349	0.987032
21	11	101	43349	0.987032
22	11	110	43349	0.987032
23	11	111	16393	0.998412
Total			3120230	0.813035

Simulation result for C3540, 8-bit ALU circuit:

Table 14 - C3540 fault types and relevant SER.

c3540	Logic	Strength	Faults	SER
1	01	111	71106	0.997401
2	00	000	159815	0.967217
3	00	001	153856	0.988606
4	00	010	462045	0.733076
5	00	011	462045	0.733076
6	00	100	462045	0.733076
7	00	101	462045	0.733076
8	00	110	462045	0.733076
9	01	000	159815	0.967217
10	01	001	202785	0.954091
11	01	010	495211	0.715219
12	01	011	495211	0.715219
13	01	100	495211	0.715219
14	01	101	495211	0.715219
15	01	110	495211	0.715219
16	11	000	159815	0.967217
17	11	001	66316	0.996269
18	11	010	66316	0.996269
19	11	011	66316	0.996269
20	11	100	66316	0.996269
21	11	101	66316	0.996269
22	11	110	66316	0.996269
23	11	111	21786	0.999933
Total			6113154	0.872207

Simulation result for C5315, 9-bit ALU circuit:

Table 15 - C5315 fault types and relevant SER.

c5315	Logic	Strength	Faults	SER
1	01	111	62361	0.999911
2	00	000	122924	0.973983
3	00	001	131314	0.996404
4	00	010	316691	0.745383
5	00	011	316691	0.745383
6	00	100	316691	0.745383
7	00	101	316691	0.745383
8	00	110	316691	0.745383
9	01	000	122924	0.973983
10	01	001	165391	0.96022
11	01	010	337962	0.726292
12	01	011	337962	0.726292
13	01	100	337962	0.726292
14	01	101	337962	0.726292
15	01	110	337962	0.726292
16	11	000	122924	0.973983
17	11	001	71091	0.999512
18	11	010	71091	0.999512
19	11	011	71091	0.999512
20	11	100	71091	0.999512
21	11	101	71091	0.999512
22	11	110	71091	0.999512
23	11	111	31246	1
Total			4458895	0.879736

Simulation result for C6288, 16x16 multiplier circuit:

Table 16 - C6288 fault types and relevant SER.

c6288	Logic	Strength	Faults	SER
1	01	111	206543	0.452499
2	00	000	216597	0.4114
3	00	001	280814	0.217254
4	00	010	320058	0.083728
5	00	011	320058	0.083728
6	00	100	320058	0.083728
7	00	101	320058	0.083728
8	00	110	320058	0.083728
9	01	000	216597	0.4114
10	01	001	293572	0.176225
11	01	010	311360	0.111811
12	01	011	311360	0.111811
13	01	100	311360	0.111811
14	01	101	311360	0.111811
15	01	110	311360	0.111811
16	11	000	216597	0.4114
17	11	001	183609	0.519279
18	11	010	183609	0.519279
19	11	011	183609	0.519279
20	11	100	183609	0.519279
21	11	101	183609	0.519279
22	11	110	183609	0.519279
23	11	111	88521	0.813474
Total			5777985	0.303783

Simulation result for C7552, 32-bit adder/comparator circuit:

Table 17 - C7552 fault types and relevant SER.

c7552	Logic	Strength	Faults	SER
1	01	111	115280	0.992078
2	00	000	258998	0.959578
3	00	001	261368	0.972662
4	00	010	800793	0.716071
5	00	011	800793	0.716071
6	00	100	800793	0.716071
7	00	101	800793	0.716071
8	00	110	800793	0.716071
9	01	000	258998	0.959578
10	01	001	342041	0.945747
11	01	010	845400	0.703961
12	01	011	845400	0.703961
13	01	100	845400	0.703961
14	01	101	845400	0.703961
15	01	110	845400	0.703961
16	11	000	258998	0.959578
17	11	001	131984	0.989253
18	11	010	131984	0.989253
19	11	011	131984	0.989253
20	11	100	131984	0.989253
21	11	101	131984	0.989253
22	11	110	131984	0.989253
23	11	111	50762	0.998019
Total			10569314	0.861866