IMPACT OF LOGIC SYNTHESIS ON THE SOFT ERROR RATE OF DIGITAL INTEGRATED CIRCUITS

By

Daniel Limbrick

Dissertation
Submitted to the Faculty of the
Graduate School of Vanderbilt University
in partial fulfillment of the requirements
for the degree of

DOCTOR OR PHILOSOPHY

in

Electrical Engineering

December, 2012

Nashville, TN

Acknowledgements

First, I would like to thank my parents, whose love, guidance, and support gave me the strength to explore my academic limits. In both my successes and my failures, they were able to provide examples from their past that mirrored (or dwarfed) my own. This dissertation is as much their accomplishment as it is my own.

I would also like to thank my advisor, Dr. William Robinson, the most composed man I ever met. It was a pleasure working with someone who could provide advice on any matter with no judgement or ego. My improvement as a researcher is a direct result of his tutelage.

My sincerest gratitude to Trey Mack, Stacey Lawrence, Brittany Kamai, and Michell Chresfield for asking tough questions and requiring thorough answers throughout my dissertation project. Also, I would like to thank Nelson Gaspard for letting me pick his brain over many hours of billiards when we could have been conversing over more light-hearted matters.

I want to thank my committee members, Dr. Bharat Bhuva, Dr. Lloyd Massengill, Dr. Gabor Karsai, and Dr. Mark Ellingham for their research guidance. I also want to thank Dr. Schrimpf and Dr. Fleetwood for their support on my project.

I want to thank each member of the Security and Fault Tolerance (SAF-T) and Radiation Effects and Reliability (RER) groups for their support and feedback.

Finally, I would like to thank the National Science Foundation and Vanderbilt University's Provost's Graduate Fellowship for funding my dissertation project.

Contents

Page

LIST OF FIGURES v

LIST OF TABLES xi

LIST OF ACRONYMS xiii

1 INTRODUCTION 1

1.1 Research Contributions . . . 3
1.2 Organization of Dissertation . . . 4

2 RADIATION EFFECTS IN DIGITAL ELECTRONICS 6

2.1 Soft Errors . . . 7
2.2 Masking Effects . . . 8
2.2.1 Logical Masking . . . 9
2.2.2 Electrical Masking . . . 10
2.2.3 Latch-Window Masking . . . 11
2.3 Mitigation Techniques . . . 12

3 LOGIC SYNTHESIS 14

3.1 Circuit Design Representations . . . 14
3.1.1 Register-Transfer Level . . . 14
3.1.2 Cell-level Netlist . . . 15
3.1.3 Graphical Database Format . . . 16
3.2 Cell Libraries . . . 16
3.2.1 Universal Gates . . . 17
3.2.2 Standard Cells . . . 17
3.3 Design Implementation Space . . . 18
3.4 Library Binding . . . 19
3.4.1 Heuristic Algorithms . . . 19
3.4.1.1 Covering Algorithms Based on Structural Matching . . . 21
3.4.1.2 Covering Algorithms Based on Boolean Matching . . . 23
3.4.2 Rule-based Systems . . . 23

4 MODELING AND EVALUATING SOFT ERROR RATE 25

4.1 Probability of Generation . . . 25
4.2 Probability of Propagation . . . 27
4.2.1 Statistical Fault Injection . . . 29
4.2.2 Probabilistic Graphical Models . . . 30
4.2.3 Additional EPP equations . . . 30

4.2.4 Determining Input Probabilities . . . 32
4.3 Probability of Latching . . . 32
4.3.1 Worst-case Pulse Width . . . 33
4.4 Tools for Modeling and Evaluating Soft Error Rate . . . 37
4.4.1 SEUTool . . . 38
4.4.2 SEAT . . . 38
4.4.3 TFIT and SoCFIT . . . 39

5 CELL AND CIRCUIT CHARACTERISTICS THAT IMPACT RELIABILITY 40

5.1 Identifying Trends . . . 41
5.2 Circuit Characteristics . . . 41
5.2.1 Logic Depth . . . 42
5.2.2 Fanout . . . 42
5.3 Cell Characteristics . . . 43
5.3.1 Drain Area . . . 43
5.3.2 Capacitance . . . 44

6 SYNTHESIS CONSTRAINTS THAT IMPACT RELIABILITY 53

6.1 Identifying Trends Between Constraints and Reliability . . . 54
6.2 Area- vs. Timing-Constrained Implementations . . . 55
6.2.1 Cell Frequency . . . 55
6.2.2 Reliability Comparisons by FIT . . . 57
6.2.3 Reliability Comparisons by EPP . . . 57
6.3 Increasing Area and Delay Constraints . . . 66
6.3.1 Reliability Comparisons of Increasing Area by EPP . . . 66
6.3.2 Reliability Comparisons of Increasing Delay by EPP . . . 68
6.4 Reliability Comparisons of Cell Availability by EPP . . . 70
6.5 Summary . . . 70

7 RELIABILITY-AWARE SYNTHESIS 72

7.1 Post-synthesis Mitigation Techniques . . . 72
7.1.1 Rewiring . . . 73
7.1.2 Resynthesis/Rewriting . . . 73
7.2 Covering-based Reliability-Aware Synthesis . . . 75
7.2.1 Replacing Cells for Reduced Pulse Widths and Drain Areas . . . 77
7.2.2 Replacing the XOR Pattern Tree for reduced EPP . . . 80
7.2.3 Replacing Common Subcircuits for Reduced EPP, Drain Area, and Pulse Width . . . 83
7.3 Rule-based Reliability-Aware Synthesis . . . 85
7.3.1 Replacing Cells for Reduced Pulse Width . . . 86
7.3.2 Replacing Subcircuits for Reduced POF . . . 87

8 SUMMARY 93

8.1 Recommendations for Circuit Designers . . . 94
8.2 Future Extensions . . . 95

REFERENCES **96**

List of Figures

Figure Page

1.1 Electronic design automation flow for digital circuit design (adapted from Lavagno et al. (Lavagno et al., 2006)). 2

2.1 Abstraction levels for a digital integrated circuit. Faults become errors when they become observable at the level of abstraction of the user. 7

2.2 Charge generation and collection in a reverse-biased junction: (a) formation of a cylindrical track of electron-hole pairs, (b) funnel shape extending high field depletion region deeper into substrate, (c) diffusion beginning to dominate collection process, and (d) the resultant current pulse caused by the passage of a high-energy ion (Baumann, 2005). 9

2.3 Example circuit showing possible paths for sequential soft fault creation. State 000 path is only possible with an input vector of (A1; B1; C1) = (0; 0; 0). State 100 path is only possible for an input vector of (1, 0, 0). Both combinational node hits and direct latch hits can contribute to SE soft faults. (Massengill et al., 2000). 10

2.4 Example circuit showing electrical masking as a result of circuit delays caused by the switching delay of the transistors (Ramanarayanan et al., 2009). . . 11

2.5 Latch-window Masking (Mukherjee, 2008). 12

3.1 Electronic design automation flow for digital circuit design. 15

3.2 Verilog-based hardware description of a 4-bit adder. 15

3.3 Verilog-based cell-level netlist of a 4-bit adder. 16

3.4 A rendering of a small GDSII standard cell with three metal layers (dielectric has been removed). The sand-colored structures are metal interconnect, with the vertical pillars being contacts, typically plugs of tungsten. The reddish structures are polysilicon gates, and the solid at the bottom is the crystalline silicon bulk. (Carron, 2007). 17

3.5 AND gate constructed from two NAND gates. 17

3.6 CMOS transistor implementation of an AND gate. 18

3.7 Example of design space: four implementations of a logic function (De Micheli, 1994). 19

3.8 Example of a design evaluation space: (a) and (b) are area/delay trade-off points. (De Micheli, 1994). 20

3.9 Pattern graph for full adder with base functions NAND2 and INV. 22

3.10 (a) Simple cell library (b) Pattern trees (INV = white, NAND2 = black, Input = gray). 22

3.11 Example of three transformations rules (De Micheli, 1994): (a) Two cascaded two-input AND gates can be bound to a three-input AND gate. (b) A two-input AND gate with an inverted input and output can be bound to a two-input OR gate with an inverted input. (c) A two-input AND gate on a critical path with a high load can be replaced by a NAND gate followed by two inverters in parallel. The first inverter drives the critical path and the second the remaining gates, which amount to a large load. 24

4.1 MRED simulations performed using 5.5-MeV alpha particles incident from several different incident angles. Only over this small range of angles is a 5.5-MeV alpha particle able to generate enough charge to produce a propagating SET (20 fC) (Gadlage et al., 2008) 26

4.2 Example circuit and associated DAG representation. 30

4.3 Output cases for inputs involving an error (a) or the inverse of an error ($\bar{a}$) that was propagated from a previous gate. The inputs of Cases 1, 2, 4, 5, and 6 can be mirrored to produce equivalent cases (not shown). 32

4.4 Two-input NAND gate with SET current source (Zhou and Mohanram, 2006). 34

5.1 Correlation coefficient between logic depth and EPP. Ranges for the correlation coefficient, r, are separated by shaded regions: $|r| > 0.5$ represents a strong correlation, $0.3 < |r| < 0.5$ represents a medium correlation, $0.1 < |r| < 0.3$ represents a weak correlation, $|r| < 0.1$ represents no correlation. A negative r implies an inverse relationship. 42

5.2 Correlation coefficient between fanout and EPP. Ranges for the correlation coefficient, r, are separated by shaded regions: $|r| > 0.5$ represents a strong correlation, $0.3 < |r| < 0.5$ represents a medium correlation, $0.1 < |r| < 0.3$ represents a weak correlation, $|r| < 0.1$ represents no correlation. A negative r implies an inverse relationship. 43

5.3 Sensitive regions of NAND2 cell (Nicolaidis, 2011). Zone 1 represents sensitive area that causes output to flip from 0 to 1. Zones 2 and 3 represent sensitive area that causes output to flip from 1 to 0. 45

5.4 Layout of NAND2x1 cell in the FreePDK45 cell library. 46

5.5 Layout of NAND2x2 cell in the FreePDK45 cell library. 47

5.6 Layout of AND2x2 cell in the FreePDK45 cell library. 48

5.7 Layout of AND4x1 cell in the FreePDK45 cell library. 49

5.8 Selective node hardening techniques that use standard cells. 50

5.9 Schematic designs for the (a) NAND2x1 and (b) NAND2x2 cells. The extra transistors in parallel for the NAND2x2 cell add to the capacitance, and therefore the critical charge of each node compared to the NAND2x1 cell. Adapted from (Stine et al., 2007) 51

5.10 Schematic designs for the (a) AND2x1 and (b) AND2x2 cells. The larger transistor sizes in the NAND stage and the extra transistors in parallel for the Inverter stage of the AND2x2 cell add to the capacitance, and therefore the critical charge of each node compared to the AND2x1 cell. Adapted from (Stine et al., 2007) 51

5.11 Schematic designs for the (a) AND2x2 and (b) AND4x1 cells. When tying two sets of the inputs of the AND4x1 together, the capacitance, and therefore the critical charge of each transistor is equivalent to that of the AND2x2 cell. The inverter stage of the AND2x1 has larger transistors and a greater critical charge. Adapted from (Stine et al., 2007) 52

6.1 Frequency of cell usage for area-constrained implementations of selected ISCAS85 benchmark circuits. 56

6.2 Frequency of cell usage for delay-constrained implementations of selected ISCAS85 benchmark circuits. 57

6.3 FIT rate for selected ISCAS85 benchmark circuits (range: 200-7000 cells), separated by area- and timing-constrained implementations. 58

6.4 FIT rate for selected LgSynth91 benchmark circuits(range: 90-10000 cells), separated by area- and timing-constrained implementations. 58

6.5 Error propagation probability average for the selected ISCAS85 benchmark circuits (range: 200-10000 cells) separated by area-constrained and timing-constrained implementations. 60

6.6 Error propagation probability average for the selected LgSynth91 benchmark circuits (range: 8000-700 cells) separated by area-constrained and timing-constrained implementations. 60

6.7 Error propagation probability average for the selected LgSynth91 benchmark circuits (range: 80-400 cells) separated by area-constrained and timing-constrained implementations. 61

6.8 Error propagation probability average for the selected LgSynth91 benchmark circuits (range: 80-400 cells) separated by area-constrained and timing-constrained implementations. 61

6.9 Error propagation probability average for the selected LgSynth91 benchmark circuits (range: 300-1700 cells) separated by area-constrained and timing-constrained implementations. 62

6.10 Error propagation probability average for the selected LgSynth91 benchmark circuits (range: 150-2000 cells) separated by area-constrained and timing-constrained implementations. 62

6.11 Number of cells with EPP greater than 0.5 for the ISCAS85 benchmark circuits (range: 200-10000 cells) separated by area-constrained and timing-constrained implementations. 63

6.12 Number of cells with EPP greater than 0.5 for the LgSynth91 benchmark circuits (range: 8000-700 cells) separated by area-constrained and timing-constrained implementations. 63

6.13 Number of cells with EPP greater than 0.5 for the LgSynth91 benchmark circuits (range: 80-400 cells) separated by area-constrained and timing-constrained implementations. 64

6.14 Number of cells with EPP greater than 0.5 for the LgSynth91 benchmark circuits (range: 80-400 cells) separated by area-constrained and timing-constrained implementations. 64

6.15 Number of cells with EPP greater than 0.5 for the LgSynth91 benchmark circuits (range: 300-1700 cells) separated by area-constrained and timing-constrained implementations. 65

6.16 Number of cells with EPP greater than 0.5 for the LgSynth91 benchmark circuits (range: 150-2000 cells) separated by area-constrained and timing-constrained implementations. 65

6.17 Error propagation probability vs. logic depth for each node in the c432 circuit. (a) area-constrained implementation (b) timing constrained implementation. 66

6.18 EPP for the selected ISCAS85 benchmark circuits separated by map effort. 67

6.19 Correlation coefficient between fanout and EPP. Ranges for the correlation coefficient, r, are separated by shaded regions: $|r| > 0.5$ represents a strong correlation, $0.3 < |r| < 0.5$ represents a medium correlation, $0.1 < |r| < 0.3$ represents a weak correlation, $|r| < 0.1$ represents no correlation. A negative r implies an inverse relationship. 67

6.20 Error propagation probability vs. node fanout for the c432 circuit 68

6.21 Average EPP for the ISCAS85 benchmark circuits separated by mininum delay and maximum delay implementations. 69

6.22 Number of cells with EPP greater than 0.5 for the ISCAS85 benchmark circuits separated by mininum delay and maximum delay implementations. 70

6.23 EPP for the selected ISCAS85 benchmark circuits separated by implementations that have one cell restricted during synthesis. There is not a cell that consistently increases or decreases the EPP of a circuit. Additionally, the impact of removing cells is negligible in some circuits (e.g. c5315) and noticeable in other circuits (e.g. c2670). 71

7.1 Qualitative comparison (in terms of electrical properties) between INVX1 and INVX4. An increase in drive strength results in an increase in pulse attenuation. Top (Circled) = Injected waveforms and Sides = Propagated waveforms. (Rao et al., 2009). 74

7.2 Pulse width decrease, drain area decrease, area increase, and delay increase for complex logic cell implementations. 78

7.3 Pulse width and drain area calculations for alternative implementations of individual cells. The percentages of pulse width decrease and area increase are defined with respect to the pulse width and area values of the original cell. The most efficient implementations are labeled. 79

7.4 Pulse width decrease and drain area decrease per area penalty ratio for multiple benchmark circuits. 79

7.5 Error locations for XOR logic function constructed from 4 NAND gates . . 81

7.6 Three implementations of the XOR logic function. 81

7.7 Ratio of the decrease in EPP over the increase in performance penalty for each circuit implementation normalized by the NAND4 implementation. . 82

7.8 Pattern trees for the NAND4 and NAND5 implementations of the XOR logic function (INV = white, NAND2 = black, Input = gray). 83

7.9 Circuit 1. Alternative implementation for a common subcircuit of the 74181 circuit which implements the boolean equation E·D·(C + A·B) + !E 83

7.10 Circuit 2. Alternative implementation for a common subcircuit of the 74181 circuit which implements the boolean equation E·(A + B) + F·(C + D) . . 84

7.11 Pulse width efficiency calculations for the two common subcircuits of the 74181 circuit 84

7.12 Selective node hardening techniques that use standard cells to replace a two-input NAND cell. 87

7.13 Pulse width vs. deposited charge for a two-input NAND cell with multiple drive strengths (“nand2x1”, “nand2x2”, “nand2x4”) and three- and four-input NAND cells functioning as a two-input NAND cell (“nand3x1”, “nand3x2”, “nand4x1”, “nand4x2”). 88

7.14 Pulse width vs. deposited charge for a two-input AND cell with multiple drive strengths (“and2x1”, “and2x2”, “and2x4”), three- and four-input AND cells functioning as a two-input AND cell (“and3x1”, “and3x2”, “and4x1”, “and4x2”), and a two-input NAND cell (“nand2x1”). 89

7.15 Selective node hardening techniques that use standard cells to replace a two-input AND cell. 89

7.16 Pulse width efficiency (area) vs. deposited charge for a two-input AND cell with drive strength four times the original cell (“4x drive strength”), a four-input AND cell functioning as a two-input AND cell (“4-input AND”), and a two-input NAND cell connected to an inverter (“NAND -> INV”). 90

7.17 Pulse width efficiency (delay) vs. deposited charge for a four-input AND cell functioning as a two-input AND cell (“4-input AND”) and a two-input NAND cell connected to an inverter (“NAND -> INV”). 90

7.18 Pulse width efficiency (power) vs. deposited charge for a two-input AND cell with drive strength four times the original cell (“4x drive strength”), a four-input AND cell functioning as a two-input AND cell (“4-input AND”), and a two-input NAND cell connected to an inverter (“NAND -> INV”). . 91

List of Tables

Table Page

3.1 List of commonly constructed standard cells 18

4.1 Overall SER figures (assuming equiprobable states) for a selection of combinational cells. The SET pulse-width threshold is set to 50 ps (Alexandrescu et al., 2011). 28

4.2 Equations for computing probability at the output of a gate in terms of its inputs (Asadi and Tahoori, 2010). P_0 represents the probability that the signal value is 0. P_1 represents the probability that the signal value is 1. P_a represents the probability that the signal value has an error. $P_{\bar{a}}$ represents the probability that the signal value has an error and that error value has been inverted. 31

4.3 Probability equations for additional CMOS cells. P_0 represents the probability that the signal value is 0. P_1 represents the probability that the signal value is 1. P_a represents the probability that the signal value has an error. $P_{\bar{a}}$ represents the probability that the signal value has an error and that error value has been inverted. 33

4.4 Comparison of pulse widths generating from nMOS and pMOS transistors for a 180-nm technology. “PW” represents the pulse width of the cell. When NOR2 has inputs a,b = {1,1}, the result is 0 (as denoted by an asterisk) because the output voltage peaks at 0.4V (less than 1/2 of a VDD of 1.8V) 35

4.5 Logic state. “AND” and “OR” have an additional sensitive state because the inverter stage is vulnerable to strikes (denoted by asterisk). 36

4.6 Transient pulse widths obtained from simulating a 15 MeV-cm^2/mg ion strike on FreePDK45 cells. The pulse width of the cell normalized to an inverter cell is also included. 37

6.1 EPP of c432 for varying circuit delay optimizations. The symbol “Δ” represents delay step size and is calculated as the maximum delay minus the minimum delay divided by 4. 69

7.1 Pulse width and drain area results from (Atkinson et al., 2011) 75

7.2 Pulse width decrease, drain area decrease, area increase, and delay increase for complex logic cell implementations 76

7.3 Pulse width decrease, drain area decrease, area increase, and delay increase for complex logic cell implementations 76

7.4 Pulse width decrease, drain area decrease, area increase, and delay increase for complex logic cell implementations 76

7.5 Pulse width decrease, drain area decrease, area increase, and delay increase for complex logic cell implementations 76

7.6 EPP of implementations with best pulse width and drain area 80

7.7 Typical pulse width and area results for common subcircuit replacement within the 74181 design 85

7.8 Percentage overheads in terms of area, power, and delay and Probability of Failure (POF) reduction of selective replacement of vulnerable nodes in the ISCAS85 benchmark circuit suite. 91

7.9 Low Area ($< 5\%$) 91

7.10 Low Power ($< 1\%$) 92

7.11 Low Delay ($< 1\%$) 92

List of Acronyms

AVF	Architectural Vulnerability Factor
CMOS	Complementary Metal-Oxide-Semiconductor
DAG	Directed Acyclic Graph
EDA	Electronic Design Automation
EDAC	Error Detection and Correction
EPP	Error Propagation Probability
HDL	Hardware Description Language
LET	Linear Energy Transfer
MOSIS	Metal Oxide Semiconductor Implementation Service
MRED	Monte Carlo Radiative Energy Deposition
NMOS	n-type Metal-Oxide-Semiconductor
ODC	Observability-Don't-Care
PMOS	p-type Metal-Oxide-Semiconductor
RHBD	Radiation-Hardened By Design
RTL	Register-Transfer Level
SCMOS	Scalable Complementary Metal-Oxide-Semiconductor
SER	Soft Error Rate
SET	Single Event Transient
SEU	Single Event Upset
SFI	Statistical Fault Injection
SPICE	Simulation Program with Integrated Circuit Emphasis
TMR	Triple Modular Redundancy

Chapter 1

Introduction

Digital integrated circuits are used in electronic equipment (e.g., computers, mobile phones, and other digital home appliances) that provide a service to its user. An interruption from this service is an *error*, and the cause of that error is a *fault* (Avizienis et al., 2004). Errors that appear temporarily (i.e., *soft errors*) are becoming an increasingly serious problem for combinational logic as technology scales (Shivakumar et al., 2002). These radiation-induced soft errors are the result of particle strikes typically caused by: (1) alpha particles from package decay, (2) cosmic rays that produce energetic protons and neutrons, and (3) thermal neutrons (Dodd and Massengill, 2003; Srour and McGarrity, 1988). These particle strikes can be observed as flipped bits at the output of the affected node. If the strike occurs while the node is not in use (or not being latched), then the fault is masked from the output, and normal execution occurs. Because of masking, a circuit with a low frequency of soft errors could potentially be immune to a visible malfunction. However, there have been numerous studies to show that soft errors in microelectronics are a growing trend due to technology scaling (Baumann, 2002; Santarini, 2005; Zhou and Mohanram, 2006).

For modern digital systems, designers can use tools that take a high-level description of a module and transform it into a physical implementation. This method, known as electronic design automation (EDA) (Breuer, 1966; Jess, 2000; MacMillen et al., 2000), provides the ability to leverage functional complexity and multiple design constraints more efficiently and accurately than manual device design. A typical EDA flow can be seen in Figure 1.1. This flow starts with the functional description of a circuit, written in a high-level language (e.g., C). The EDA tool translates this code into a register-transfer level (RTL) description in a step called high-level synthesis. The RTL description can be represented using a hardware description language (e.g., Verilog). Alternatively, it is common to begin the design flow at this step. In the logic synthesis step, the RTL description is mapped to technology-specific logic gates (e.g., NAND, OR). The physical location for the mapped logic gates are specified in the placement step. The final step, routing, adds wires to the physically placed components.

Several design constraints that have garnered interest in the field of EDA impact the final topology of the circuit. Algorithms and design rules have been created to leverage area, power, and delay constraints (Brayton et al., 1987; De Micheli, 1994; Rudell, 1989). For instance, an algorithm to minimize the circuit delay might choose an implementation with the shortest logic depth. The task of finding the optimal transformation from one step to another is an intractable problem. Therefore, the estimation algorithms chosen to implement these transformations also have an impact on the topology of the final circuit.

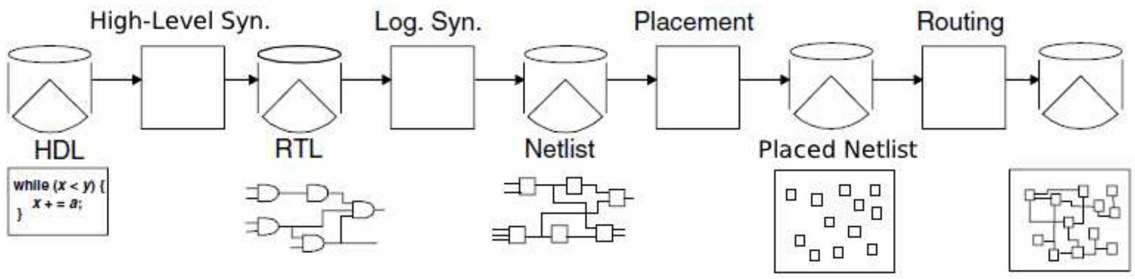


Figure 1.1: Electronic design automation flow for digital circuit design (adapted from Lavagno et al. (Lavagno et al., 2006)).

Existing fault injection studies examined the netlist of a circuit or suite of circuits, but did not necessarily examine the multiple netlists with the same functionality that can be generated from a behavioral circuit description. Current design flows of digital circuits incorporate the use of cell libraries, which contain standard implementations of common logic functions, to automate the synthesis process. The effects of soft errors on digital systems depend not only on the vulnerability of individual library cells (e.g., NAND2, INV), but also upon the various functional blocks and associated topologies constructed from them. Addressing reliability by considering the logical topology of a circuit is known as *reliability-aware synthesis* (Almukhaizim et al., 2006; Krishnaswamy et al., 2007; Mishchenko and Chatterjee, 2006). Existing reliability-aware synthesis techniques selectively harden nodes to minimize performance overhead and typically involve: (1) identifying vulnerable nodes through fault injection simulations, and (2) replacing vulnerable nodes with less vulnerable implementations. An examination of circuit topologies is necessary to implement reliability-aware synthesis. At the first glance, the wide variety of circuit functionality and library cells makes it difficult to generalize the results. However, high-level synthesis from hardware description languages (HDLs) relies upon a design compiler to construct topologies from the various library cells. Similar to a compiler for programming languages, the design compiler utilizes certain library cells more frequently and generates certain cascades of library cells that appear often as common subcircuits. To date, there has been little research investigating the correlation between synthesis trends and estimated soft error rates (SER) for a suite of circuits, taking into account the actual circuit functionality.

Additionally, while EDA tools contain algorithms to reduce area, power, and delay while preserving circuit functionality, little research has been conducted to identify a design characteristics that can be used in the EDA tool flow to estimate a circuit's vulnerability. Correlating synthesis trends with estimated soft error rates led to the identification of such characteristics. Although research has examined logical, electrical, and latch-window masking, translating that information into a reliability metric for logic synthesis remains an open challenge. This step is necessary to estimate SER autonomously. In order to develop this metric, the probabilities of generation, propagation, and latching of a transient were concurrently explored as it relates to the individual cell and the logical interconnection of cells. With the identification of a reliability estimation metric, the impact of logic synthesis on reliability was analyzed, and mitigation techniques were developed. Specifically, the vulnerability of a circuit to soft errors were reduced by reliability-aware synthesis. Examining the effects of circuit topology on soft error vulnerability in terms of: (1) the selection of individual library cells (2) the logical interconnection of library cells and (3) the adjacency of library cells in the final design layout, regardless of logical connectivity, provided insight in this regard. Additionally, the reliability metric obtained for a circuit design was used as a weighting factor when optimizing a circuit design for reliability in a manner similar to the use of area, power, and delay metrics.

1.1 Research Contributions

The contributions of this research are summarized as follows:

1. **Identification of design characteristics that can be used by EDA tools to estimate reliability during the logic synthesis step.** Logic depth, capacitance, and cell adjacency are presented as characteristics that relate to reliability (Limbrick et al., 2011; Limbrick and Robinson, 2012).

2. **Identification of the impact of optimization algorithms during the logic synthesis step on the reliability of a circuit.** Area- and Delay-optimized implementations are shown to have consistent reliability trends (Limbrick et al., 2011).

3. **Development of an approach to minimize circuit vulnerability based on cell selection.** In a standard cell library, there are multiple ways to vary the capacitance at a vulnerable node. Results indicate that a combination of strategies is necessary to limit performance overhead (Limbrick et al., 2012a,b).

These contributions extend the ability of a circuit designer to both estimate and improve the reliability of combinational logic in a large system (e.g., microprocessor) at an early design stage. Using a cell-level netlist and a table of error generation probabilities for each cell as inputs, a designer can: (1) estimate which cells have the highest probability of propagating and latching, and (2) determine which cell-replacement strategy is most cost-effective at improving the reliability of the vulnerable cell. Additionally, the improvement in reliability that can be gained by using this methodology is independent of improvements at lower levels of abstraction (e.g., device-level techniques).

1.2 Organization of Dissertation

The research described in this dissertation is organized as follows:

1. Chapter 2 provides the background information on radiation effects in digital electronics. It includes a brief description of the basic mechanisms of radiation-induced soft error effects and the mechanisms by which these effects can be masked.

2. Chapter 3 provides the background information on logic synthesis. The chapter explains the multiple abstraction layers considered when leveraging design constraints to produce a gate-level representation of a digital circuit.

3. Chapter 4 describes the methods used to model and evaluate the soft error rate of a circuit. It also includes a brief summary of tools that have implemented these methods.

4. Chapter 5 presents the cell and circuit characteristics that impact reliability (Contribution 1).

5. Chapter 6 describes the reliability trends associated with optimizing a circuit implementation under various design constraints (Contribution 2).

6. Chapter 7 demonstrates the techniques used to reduce the reliability of a circuit while leveraging area, power, and delay (Contribution 3).

7. Chapter 8 summarizes this work and provides some extensions for future study.

Chapter 2

Radiation Effects in Digital Electronics

The correct operation of digital microelectronics can be affected by the operating environment. In space, radiation from the sun and galactic cosmic rays can cause temporary or permanent failure of a device. In terrestrial environments, these radiation sources still affect the operation of the device but with less magnitude. Therefore, research into the construction of reliable circuits has relevance in both environments. The effect of these radiation sources on the operation of a device can be classified into three types of faults (Constantinescu, 2002): (1) permanent, (2) intermittent, and (3) transient. Permanent faults are irreversible device malfunctions. Intermittent faults are repeated temporary malfunctions at a circuit node that occur as a result of instability in hardware. Transient faults are temporary malfunctions caused by environmental conditions. Transient and intermittent faults were observed to be the predominant fault type in modern microelectronics (Lin and Siewiorek, 1990). In circumstances where faults result in an observable malfunction at the output, the fault is referred to as an *error*. An error is a deviation of a system from its correct state (Mukherjee, 2008). Figure 2.1 illustrates the abstraction levels of a microelectronic system. The view of the user varies by abstraction level, and therefore, so does the definition of an error. For example, an incorrect signal that is seen at the output of a circuit but not at a time when an application is using that signal would be considered an error at the circuit level but not at the application level. The circumstance where a fault does not become an error is a phenomenon known as *masking*. This work focuses on *soft errors*, the observable malfunctions at the output of a circuit that are caused by radiation-induced transient current pulses. This chapter describes the basic mechanisms by which radiation temporarily alters circuit operation in combinational logic. The chapter also contains an overview of masking situations and related mitigation techniques.

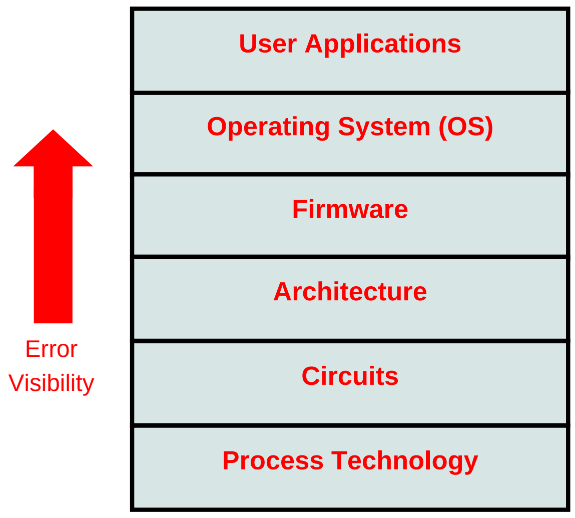


Figure 2.1: Abstraction levels for a digital integrated circuit. Faults become errors when they become observable at the level of abstraction of the user.

2.1 Soft Errors

When an alpha particle or a neutron strikes a circuit, it potentially generates charge sufficient enough to cause a malfunction. As seen in Figure 2.2, the particle can strike the drain of a transistor, interact with the molecular structure of the semiconductor material (usually silicon), generate electron-hole pairs, and eventually come to a rest when all of its energy is lost. These electron-hole pairs diffuse towards the device contacts. This diffusion creates current and interferes with the normal operation of the transistor. Additionally, the movement of charge carriers creates drift current that also disrupts normal operation. Particles can also release charge indirectly when the nuclear reactions between the particle and the struck device create secondary particles that deposit energy along their path similar to directly ionizing particles.

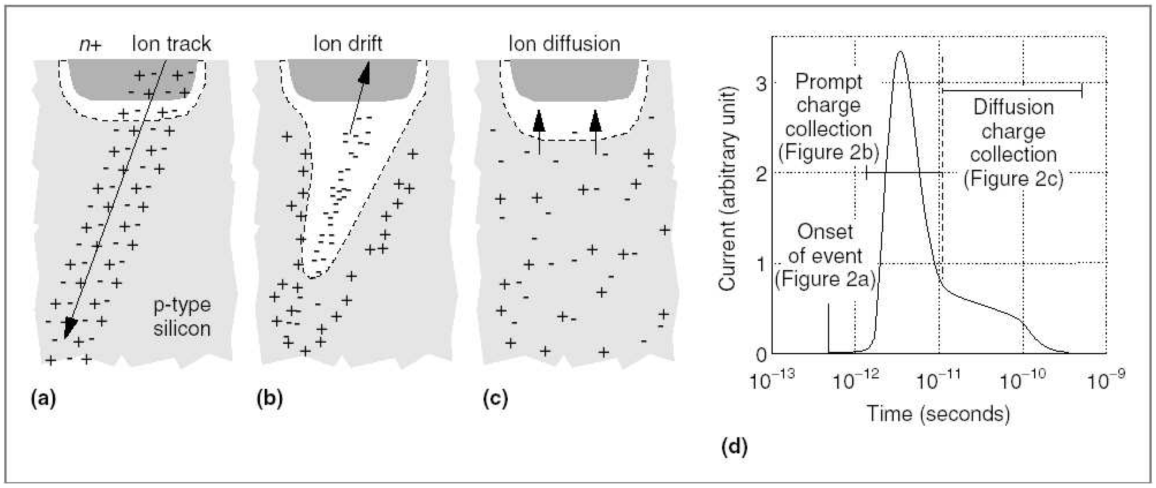
The interaction of the striking particle can be characterized by the total path length traveled before all of its energy is lost, referred to as the *range*. The energy lost per unit path length of a particle as it passes through a material is known as the *linear energy transfer* (LET), measured in MeV/cm^2/mg. The amount of charge generated by ionizing particles varies with technology. In CMOS technology, the “sensitive area” of the device has been considered to be the reverse-biased drain junction of a transistor biased in the off state (Dodd et al., 1996; Detcheverry et al., 1997), though this approximation is less accurate as technology scales. Dodd et al. (Dodd et al., 2001) showed that at LET ranging 11-26 MeV, the most sensitive area of a 0.6μm bulk CMOS technology is found to be at most the reverse-biased diffusion region of the NMOS transistor and that the reverse-biased diffusion region of the PMOS transistor becomes sensitive only for higher LETs.

The collected charge can result in the bit flip of a storage element, referred to as a *single event upset (SEU)* or a glitch in combinational logic, referred to as a *single event transient (SET)*. Soft errors can be characterized by the *critical charge*, the minimum charge to cause an SEU or SET, and the *pulse width*, the amount of time that the SET can be mistaken for a valid signal. Warren et al. used Monte-Carlo radiation transport code coupled with SPICE circuit level simulation to identify regions of SEU vulnerability in an SEU hardened flip-flop based on the amount of charge collected (Warren et al., 2008). This approach could be extended to analyze single-event effects in combinational logic by evaluating collected charge in combinational logic cells as well. A particle strike at a logic gate's input node can cause an incorrect output to occur for as long as the additional charge remains on the node. If a particle strikes the input of a storage cell (i.e., latch), then the incorrect output can be stored within that storage cell, provided that the strike occurs while the storage cell is accepting inputs.

The sensitivity of a circuit to a single event depends on the transient-generation probability. The actual transient shape (Figure 2.2(d)) is defined by: (1) the particle type and energy, (2) the transistor size of the driver, and (3) the load capacitance.

2.2 Masking Effects

Masking is the effect where a particle strikes a device but does not result in an error at the output. It can be divided into three main categories (Dodd and Massengill, 2003): logical, electrical, and latch-window masking. Each category is discussed in detail in this section. It is important to understand the mechanisms of masking because exploiting these effects provides an opportunity to improve the overall reliability of a circuit design.



n+
Ion track
p-type silicon
(a)
Ion drift
(b)
Ion diffusion
(c)
Current (arbitrary unit)
Prompt charge collection (Figure 2b)
Diffusion charge collection (Figure 2c)
Onset of event (Figure 2a)
Time (seconds)
(d)

Figure 2.2: Charge generation and collection in a reverse-biased junction: (a) formation of a cylindrical track of electron-hole pairs, (b) funnel shape extending high field depletion region deeper into substrate, (c) diffusion beginning to dominate collection process, and (d) the resultant current pulse caused by the passage of a high-energy ion (Baumann, 2005).

2.2.1 Logical Masking

The rate at which SETs in combinational logic produce an observable error at an output node depends upon the existence of an available path for the SET to propagate. If no path exists, then the fault is considered to be *logically masked*. Logical masking can occur in one logic cell or a combination of cells. For example, a two-input OR gate logically masks a strike on an input node if the other input has a high logic value. An example of logical masking in a combination of cells can be seen in Figure 2.3.

Logical masking can be evaluated using the *error propagation probability (EPP)*, the likelihood that an error in one component propagates to output of the circuit. EPP can be calculated with Equation 2.1:

$$\text{EPP} = \frac{\Sigma \text{ input combinations resulting in error}}{\Sigma \text{ all possible input combinations}} \tag{2.1}$$

This metric is useful when trying to characterize the logical masking of a circuit in all possible circumstances. However, more often a circuit is designed with an intended purpose that bounds the range of inputs that the circuit will encounter. For example, an arithmetic-logic unit could be designed to execute five operations based on a three-bit operation code.

Since a three-bit operation code is capable of representing eight operations, three bit combinations will be unused. In this case, calculating the EPP for all possible combinations when the operation code is set to an unused combination provides no information about logical masking. There is not a standard equation that de-rates the EPP by the probability of input use in a particular application, therefore each design must be evaluated separately to include this factor.

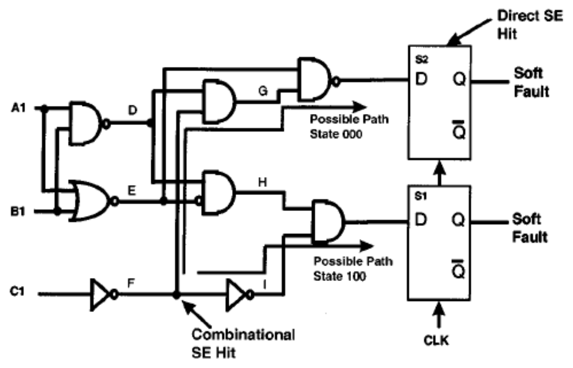


Figure 2.3: Example circuit showing possible paths for sequential soft fault creation. State 000 path is only possible with an input vector of (A1; B1; C1) = (0; 0; 0). State 100 path is only possible for an input vector of (1, 0, 0). Both combinational node hits and direct latch hits can contribute to SE soft faults. (Massengill et al., 2000).

2.2.2 Electrical Masking

The pulse that is generated when an ion strikes a device can be attenuated prior to reaching an output node. This occurrence is referred to as *electrical masking*. The probability of a signal being electrically masked depends on the characteristics of the generated pulse, the gates that the pulse traverses prior to the output node, and the electrical noise margin of the output node.

The generation of the SET pulse is directly influenced by the drain area of the transistor being struck. Within the same fabrication technology, larger drain areas will increase the likelihood of a direct strike, the amount of charge collected, and subsequently, the duration and amplitude of the pulse.

As the SET pulse propagates, the rise and fall times of the pulse increase, meaning that its amplitude decreases. This can be seen in Figure 2.4 where the shape of the pulse generated by the particle strike is transformed as it passes through subsequent gates. The propagation of an SET through a logic gate is based on the capacitance associated with the cell used and the resistances of conducting channel-connected transistor paths associated with the fault-injected node (Dahlgren, 1995).

The pulse that reaches the output is not observed if its amplitude is attenuated to the point that it falls out of the range of voltages that correspond to a low voltage (logic zero) for a one-to-zero bit-flip or a high voltage (logic one) for a zero-to-one bit-flip.

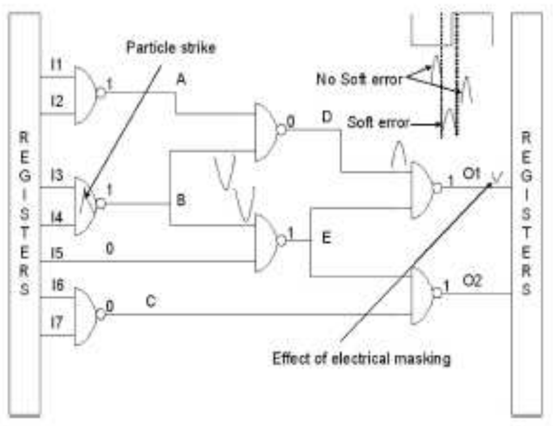


Figure 2.4: Example circuit showing electrical masking as a result of circuit delays caused by the switching delay of the transistors (Ramanarayanan et al., 2009).

2.2.3 Latch-Window Masking

The generated pulse can also be masked if the pulse does not reach the output node during the time that the corresponding storage element is capturing its input value. This occurrence is referred to as *latch-window masking* and is directly impacted by the width of the pulse and the timing requirements of the output storage element. An example of latch-window masking can be seen in Figure 2.5.

The *setup time* for a flip-flop is the minimum amount of time that data should be constant prior to a clock transition to ensure reliable function. The *hold time* is the minimum amount of time that data should be constant after a clock transition to ensure reliable function. The *latching window* is the time between the setup time and the hold time. Violating either requirement creates metastability in latches. As Figure 2.5 shows, SET pulses that fully span the latching window will not be masked. SET pulses that partially span the latching window may or may not be masked because of the metastability of the latch. SET pulses that are fully outside of the latching window will be masked.

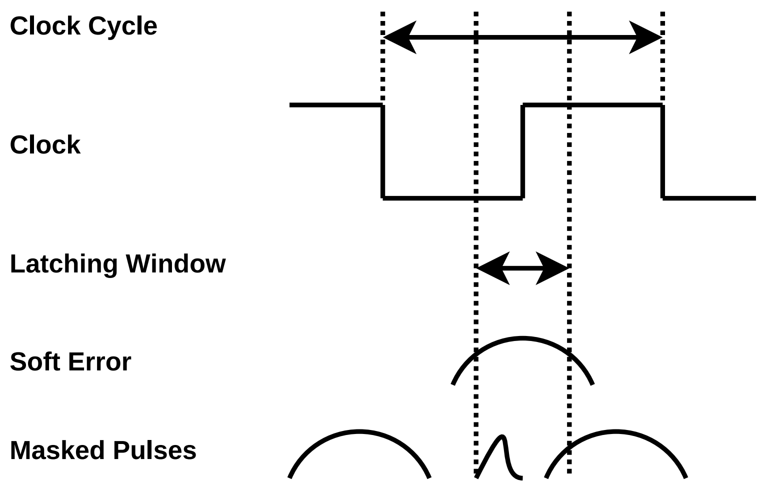


Figure 2.5: Latch-window Masking (Mukherjee, 2008).

2.3 Mitigation Techniques

Strategies to mitigate soft errors have been developed at the device level, the circuit level, and the architecture level (Nicolaidis, 2005). At each level, the mitigation strategy focuses on the inclusion of additional hardware or modification of existing hardware. Techniques that focus on the synthesis process, including the technique introduced in this thesis, are presented in Chapter 7.

At the device level, the fundamental technique utilized is to reduce the amount of collected charge at a node. One method is to introduce extra doping layers to limit substrate charge collection (Fu et al., 1985; Burnett et al., 1993). Another possibility is to use a specially designed radiation-hardened fabrication process. However, these processes are typically two or three generations behind the commercial state-of-the-art. At the circuit level, there are a number of techniques that fall under the category of radiation-hardened-by-design (RHBD) (Alexander et al., 1996; Mavis and Alexander, 1997; Anelli et al., 1999; Lacoe et al., 2000). Special latches that are tolerant or immune to SEU have been developed using spatial redundancy because of the emphasis on sequential logic (Mavis and Eaton, 2002; Calin and Nicolaidis, 1996; Hazucha et al., 2003). Temporal redundancy is also a technique that has been used to mitigate SETs (Nicolaidis, 1999). Spatial and temporal techniques can significantly impact the circuit area or the circuit delay if applied throughout the circuit without consideration of the actual circuit functionality. At the architecture level, error detection and correction (EDAC) can be used to monitor and correct errors in storage elements (Chen and Hsiao, 1984; Bossen and Hsiao, 1980). This approach requires that extra bits of information be stored with the data to reconstruct the original data in the event of an upset (Dodd and Massengill, 2003). Another technique is the use of triple modular redundancy (TMR) to replicate components for a majority voting scheme (Sosnowski, 1994). Control circuitry can use methods such as watchdog timers or lockstep operation to detect errors (Kinnison, 1998). There are also hardware/software redundancy techniques to reduce the soft error rate while executing the application (Austin, 1999; Ray et al., 2001; Weaver et al., 2004; Reis et al., 2005). These techniques vary in terms of the overhead costs required to mitigate errors.

Chapter 3

Logic Synthesis

In electronic design automation, there are several steps that translate the high-level functional description of a circuit design into a physical implementation of the circuit. A typical EDA design flow is presented in Figure 3.1. In this flow, design creation begins with a written hardware description at the register-transfer level (RTL). This RTL description is then mapped to a specific technology library as a netlist in a process known as logic synthesis. Exact locations for the circuit components in the netlist are assigned during the placement step. Finally, the routing step adds the wires needed to properly connect the placed components while obeying all design rules for the circuit. The final product is a physical layout in a graphical database format (e.g., GDSII).

This chapter focuses on the step that maps the hardware description of a circuit design to a cell library, called logic synthesis. A description of circuit design representations throughout the EDA flow is also presented.

3.1 Circuit Design Representations

Each step of the EDA design flow has its own representation of the circuit design. The RTL hardware description, cell-level netlist, and graphical database format layout are described in this section.

3.1.1 Register-Transfer Level

The formal specifications of an electronic circuit design, including the logical function of a block and the interaction between blocks, are expressed using *hardware description languages* (HDLs) such as Verilog and VHDL. HDL descriptions written at the register-transfer level (RTL) contain information on the combinational logic of the circuit and the transfer of this logic between hardware registers. This level of abstraction hides characteristics about the circuit, such as the exact logic gates used to implement the logic function and the tech-

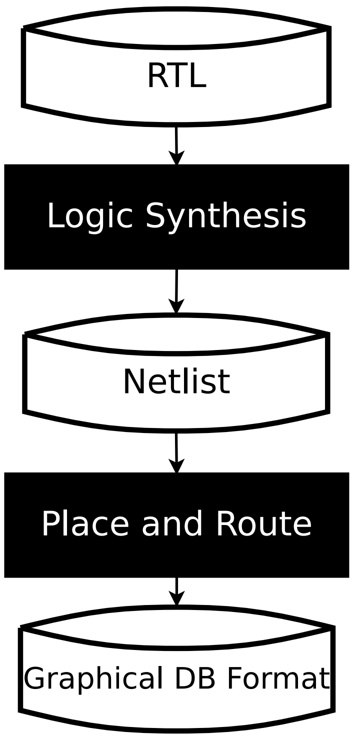


Figure 3.1: Electronic design automation flow for digital circuit design.

nology with which the gates are constructed. An example of Verilog-based RTL code for a 4-bit adder is shown in Figure 3.2.

```
module  adder (cin, cout, a, b, sum);
    input cin;
    input [3:0] a, b;
    output [3:0] sum;
    output cout;

    assign {cout, sum} = a + b + cin;

endmodule
```

Figure 3.2: Verilog-based hardware description of a 4-bit adder.

3.1.2 Cell-level Netlist

The logical interpretation of the design can be expressed as a gate-level netlist. This netlist specifies the physical cells that are used to implement the logic specified in the hardware description. The cells are selected from a given *cell library* that maps to the physical implementation and are chosen based on given design constraints, such as maximum area or maximum circuit delay.

A netlist file consists of instances of logical components and *nets*, the wires that connect these components. There are multiple formats with which a netlist can be written, including those supported by HDLs. Figure 3.3 shows an example of a cell-level netlist written for the RTL code of Figure 3.2.

```
module adder (sum cout, cin, a, b);
    input a, b, cin;
    output sum, cout;
    wire a, b, cin;
    wire sum, cout;
    wire n_0, n_1, n_2;

    XOR2X1 g14( .A(cin), .B(n_0), .Y(S) ) ;
    XOR2X1 g15( .A(a), .B(b), .Y(n_0) );
    OR2X1 g16( .A(n_1), .B(n_2), .Y(cout) ) ;
    AND2X1 g17( .A(cin), .B(n_0), .Y(n_1) ) ;
    AND2X1 g18( .A(a), .B(b), .Y(n_2) ) ;

endmodule
```

Figure 3.3: Verilog-based cell-level netlist of a 4-bit adder.

3.1.3 Graphical Database Format

The details of the physical layout of the circuit can be implemented in the design of the individual cells in the cell library. The final format is a physical layout in a graphical database format capable of representing planar geometric shapes, text labels, and other information about the layout in hierarchical form. An example of a standard cell in GDSII format is shown in Figure 3.4.

3.2 Cell Libraries

In the circuit design process, after the hardware description of the circuit is written and tested, it is mapped to a specific technology. This mapping from an HDL description to logic gates in a technology cell library is referred to as *logic synthesis*. Synthesizing the circuit

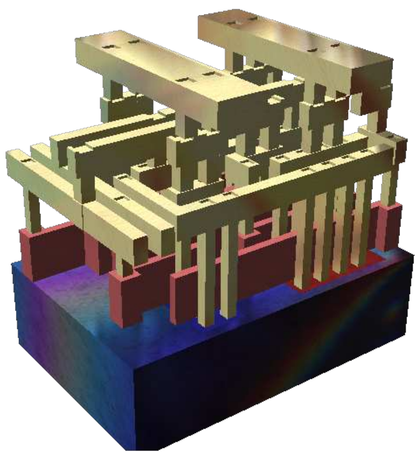


Figure 3.4: A rendering of a small GDSII standard cell with three metal layers (dielectric has been removed). The sand-colored structures are metal interconnect, with the vertical pillars being contacts, typically plugs of tungsten. The reddish structures are polysilicon gates, and the solid at the bottom is the crystalline silicon bulk. (Carron, 2007).

provides the fabrication layout for the design. At this level, the physical characteristics of the circuit, such as the area, maximum clock speed, and power consumption, can be obtained based on the technology used.

3.2.1 Universal Gates

Each hardware description or Boolean equation can be physically implemented using only NAND gates or only NOR gates. These gates are commonly called *universal gates* because of this property. For instance, an AND gate can be constructed from connecting the output of a NAND gate to the two inputs of another NAND gate, as seen in Figure 3.5.

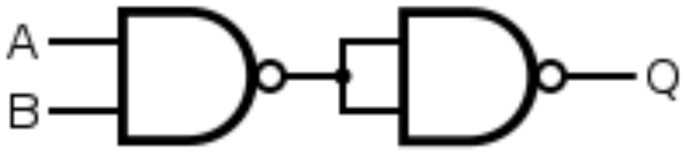


Figure 3.5: AND gate constructed from two NAND gates.

3.2.2 Standard Cells

Implementing logic functions using universal gates would be ideal if a NAND-only solution produced the most efficient circuit. However, in modern EDA, smaller, faster representations of logic functions can be implemented by using gates constructed from transistor level logic. These gates are commonly referred to as *standard cells*. An example of the transistor-level construction of an AND gate is shown in Figure 3.6. A list of commonly constructed custom gates are shown in Table 3.1.

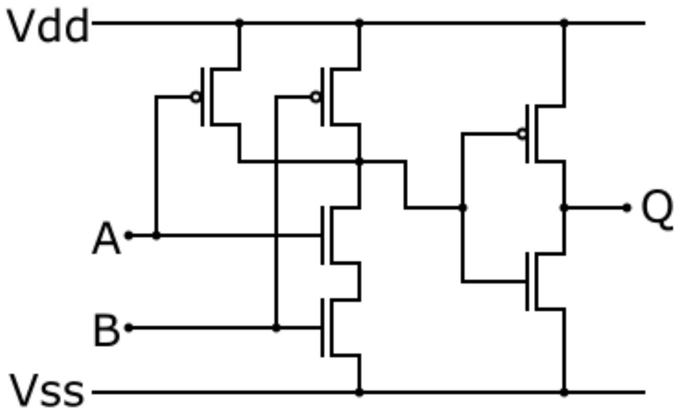
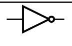
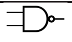
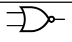
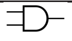
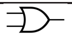
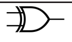
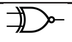
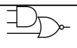
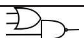
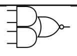
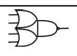


Figure 3.6: CMOS transistor implementation of an AND gate.

Table 3.1: List of commonly constructed standard cells

Cell Name	Symbol	Boolean function
INV1		$Y = \bar{A}$
NAND2		$Y = \overline{AB}$
NOR2		$Y = \overline{A + B}$
AND2		$Y = AB$
OR2		$Y = A + B$
XOR2		$Y = A \oplus B$
XNOR2		$Y = \overline{A \oplus B}$
AOI21		$Y = \overline{AB + C}$
OAI21		$Y = \overline{(A + B)C}$
AOI22		$Y = \overline{AB + CD}$
OAI22		$Y = \overline{(A + B)(C + D)}$

3.3 Design Implementation Space

The interaction between the register-transfer, cell, and physical layout levels influence the final implementation of the circuit design. The style used to write the architectural description alters the cell-level interpretation. The algorithms that enforce the design constraints, and the constraints themselves, affect the cells chosen from the physical-layout level. The physical layout of the devices alter the characteristics to be interpreted at the cell level. An example of this interaction between the cell level and physical-layout level can be seen in Figures 3.7 and 3.8. Figure 3.7 shows four possible implementations of the same Boolean function. Figure 3.8 shows the potential area and delay characteristics of the implementation.

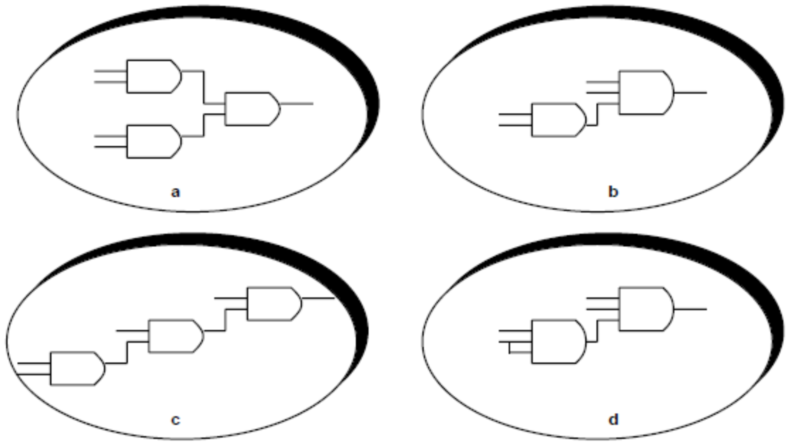


Figure 3.7: Example of design space: four implementations of a logic function (De Micheli, 1994).

3.4 Library Binding

The step in logic synthesis that selects the cells to be used for a given Boolean network is known as library binding or technology mapping. Understanding how tools map standard cells to logic functions provides insight into reliability-aware synthesis. Most tools use a combination of both heuristic and rule-based algorithms to accomplish this task. The following subsections cover both approaches.

3.4.1 Heuristic Algorithms

Algorithms for library binding involve manipulating random logic to comply with a design constraint and are used mainly for single-output combinational cells. Algorithms can manipulate the binding of cells to a hardware description using the structural or Boolean

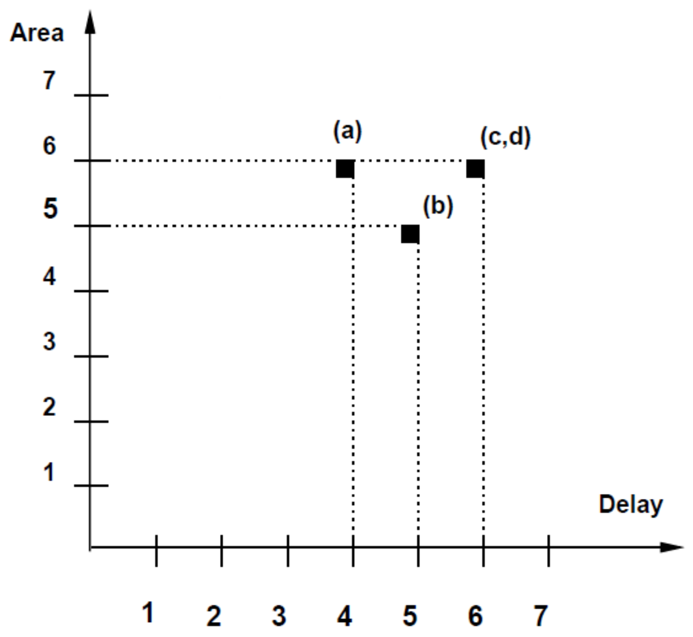


Figure 3.8: Example of a design evaluation space: (a) and (b) are area/delay trade-off points. (De Micheli, 1994).

approach. The structural approach models functions by patterns (e.g., trees, graphs) and relies on pattern matching techniques. The Boolean approach uses Boolean models to solve the tautology problem (i.e., identification of multiple equivalent representations of a Boolean equation).

Heuristic algorithms have three common steps (De Micheli, 1994) : (1) decomposition, (2) partitioning, and (3) covering. During decomposition, the network and library are cast in standard base functions, e.g., NAND2 and INV. Library-binding tools require the existence of at least a minimal set of base functions, e.g., one NAND2 and one INV for the Cadence Encounter RTL compiler, to ensure the existence of a solution. Network decompositions into base functions are not unique and therefore affect the topology of the circuit. For example (Hatchel and Somenzi, 1996), consider the logic function $f = (abcd + efgh + ijkl + mnop)'$. Using the four-input NAND gate as the base function produces only one implementation: a tree of five four-input NAND-gates. Representing all implementations for this same function using two-input NAND-gates and inverters produces 18 implementations. More implementations are ideal because they provide a greater opportunity for optimization. Partitioning is the step where the network is broken into cones of multiple-input, single-output networks. These network cones are referred to as *subject graphs*. Schemes to partition a network include: (1) partitioning vertices with multiple outgoing edges and (2) partitioning blocks consisting of all vertices that are tails of paths at primary outputs and deleting them from from the graph. Covering involves placing a library cell into each sub-network of the network. This cover is found by trying to match all cells in the library to that portion of the subject graph. Once a match is selected, the subject graph is labeled with the matching cell along with its area and timing attributes.

3.4.1.1 Covering Algorithms Based on Structural Matching

Structural matching involves identifying a pattern in a network that is identical to the structure of a cell in the library. Figure 3.9 shows the pattern graph for a full adder. Each cell has a structure that translates to the base functions chosen by the compiler. The cell structures that correspond to base functions NAND2 and INV can be seen in Figure 3.10. With this representation, the network can be covered through tree-based matching or tree-based covering.

In simple tree-based matching, one type of base function is used in decomposition, i.e., two-input NANDs or two-input NORs. Then an algorithm is used to determine if the pattern tree matches a subgraph in a subject tree. One such algorithm matches the root of the pattern tree to the vertex of the subject tree and visits their children recursively. The equivalence is determined by comparing the degrees (i.e., number of inputs and outputs) of the vertices and their children.

An algorithm for tree-based matching using automata was presented by Aho and Corasick (Aho and Corasick, 1975). This approach involves translating the trees into strings which allows for base functions to be arbitrary, since each function can be encoded by a single character. The string representation corresponds to a path (i.e., a sequence of vertices and edges) from the root to a leaf. The string has a character denoting each vertex in the path and its corresponding base function. Again, the set of base functions affects the topology of the circuit.

Optimum tree-based covering can be implemented through dynamic programming (Aho and Johnson, 1976; Keutzer, 1988). In a bottom-up fashion, the tree covering algorithm

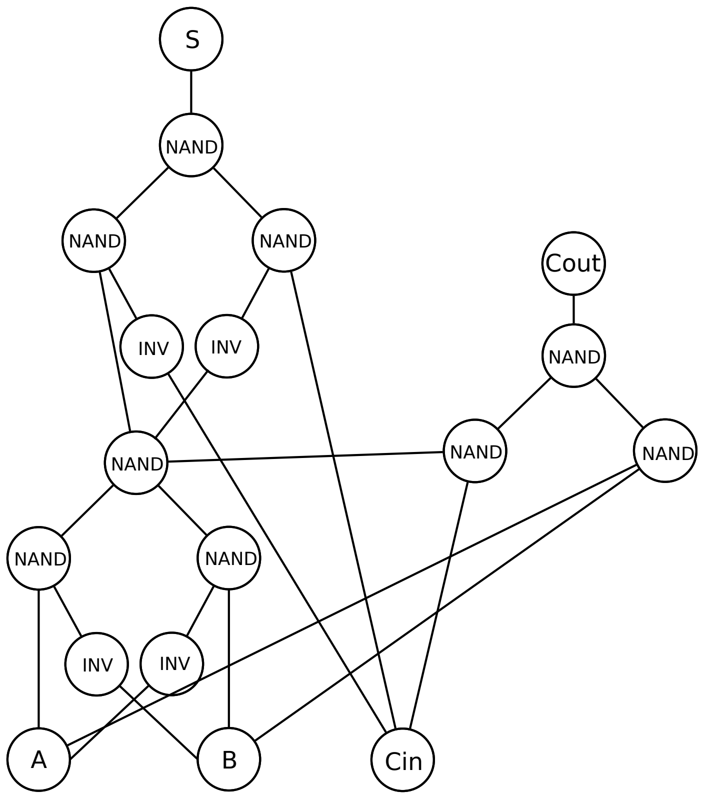


Figure 3.9: Pattern graph for full adder with base functions NAND2 and INV.

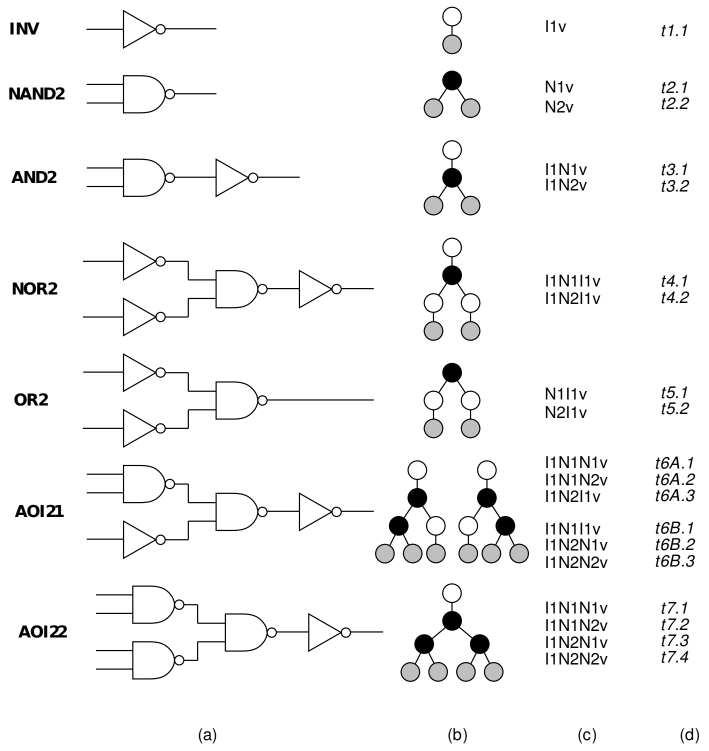


Figure 3.10: (a) Simple cell library (b) Pattern trees (INV = white, NAND2 = black, Input = gray).

matches locally rooted subtrees with the pattern trees. Once a match is found, the associated cell cost is attributed to that subtree. The set of possible solutions can be compared for overall cost.

3.4.1.2 Covering Algorithms Based on Boolean Matching

Boolean matching algorithms compare subnetworks and cells by representing them as *cluster functions* and *pattern functions* respectively. The cluster function is represented by $f(x)$ and the pattern function is represented by $g(y)$, where x and y denote the input variables of the subnetwork and library cells respectively. Through a series of tautology (i.e., satisfiability) checks, it is determined if there exists a permutation matrix P such that $f(x) = g(Px)$. The permutation matrix allows g to map to a location in f.

Once a match is determined, the network can be covered in a similar fashion to tree covering. Additionally, the solution is dependent upon the decomposition and partition into subject graphs.

3.4.2 Rule-based Systems

Rule-based systems for library binding attempt to mimic the decisions of a designer by applying repetitive transformations. These transformations are stored as entries in a database. Each entry in the rule database contains a circuit pattern, along with an equivalent replacement pattern in terms of one or more library elements. This equivalent pattern is associated with the best implementation for the design metric of interest. For example, a rule could be defined to replace adders in a design with an adder implementation in a cell library, essentially reducing the area of the circuit. Rules are also used in situations where designers want to handle special circumstances, e.g., high-fanout problems. Figure 3.11 shows three examples of simple transformation rules.

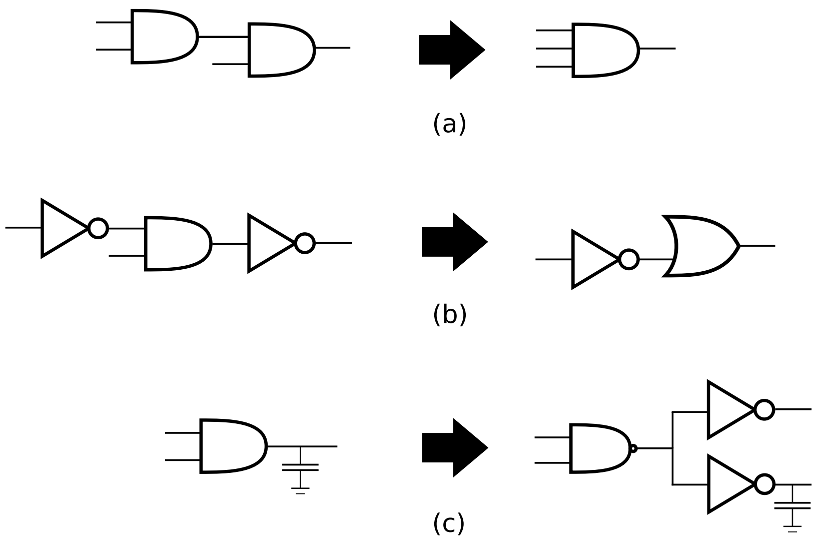


Figure 3.11: Example of three transformations rules (De Micheli, 1994): (a) Two cascaded two-input AND gates can be bound to a three-input AND gate. (b) A two-input AND gate with an inverted input and output can be bound to a two-input OR gate with an inverted input. (c) A two-input AND gate on a critical path with a high load can be replaced by a NAND gate followed by two inverters in parallel. The first inverter drives the critical path and the second the remaining gates, which amount to a large load.

Chapter 4

Modeling and Evaluating Soft Error Rate

When an alpha particle or neutron strikes a circuit, it potentially generates charge sufficient enough to cause a malfunction. The particle can strike the drain of a transistor and generate electron-hole pairs that diffuse towards the device contacts. The collected charge can result in a glitch in combinational logic, referred to as a *single event transient (SET)*.

SET pulses affect the output of combinational logic when it has sufficient duration to be latched by a memory element. This duration, i.e., *pulse width*, becomes important in characterizing the impact of the particle strike on the circuit. When considering an SET that causes the output of a cell to transition from "0" to "1", the pulse width is the period of time that the output voltage is greater than $V_{DD}/2$.

There is extensive research that has previously been conducted in relation to modeling and evaluating SET pulse as it pertains to the soft error rate (SER), including techniques that incorporate logic synthesis. Reliability can be determined using the probability of failure (POF) equation:

$$POF = P_{generation} \times P_{propagation} \times P_{latching} \tag{4.1}$$

where $P_{generation}$ is the probability that a transient will be generated in the circuit by the incident particle, $P_{propagation}$ is the probability that the transient will propagate to a storage element, and $P_{latching}$ is the probability that the transient will be latched. Each probability is discussed in this section. Additionally, a brief survey of SER tools is included here.

4.1 Probability of Generation

The probability of generation for enhancement-mode MOSFETs is influenced by: (1) the operating environment, (2) the doping characteristics of the device, (3) the device geometry, and (4) the bias conditions of the device. The environment of particles that interact with the circuits considered in this study are consistent with a terrestrial environment and have energy ranges of 0-15 MeV. These ranges are consistent with alpha particles and lightly ionizing neutrons. The doping characteristics are similar for the pMOS and nMOS transistors of each cell, and therefore have little impact when estimating the relative soft error rate that each cell contributes. The device geometry impacts SER estimation because a larger sensitive volume implies a larger ion strike target and a greater capacity for charge collection. This is counteracted by the increase in capacitance that a larger volume contains, meaning a shift in the critical charge necessary to cause an error. The vulnerable bias condition is the "off" drain of any transistor in a cell. While this fact is consistent across cells, the likelihood of a cell having transistors in the "off" state will be based on the inputs to that cell and the logic function of the cell.

In order to determine the probability of generation for each cell, the number of incidents per charge for a given device can be experimentally observed or simulated at the device level. Example data from device-level simulations are shown in Figure 4.1. The probability of occurrence of these particles, known as the flux, can be obtained.

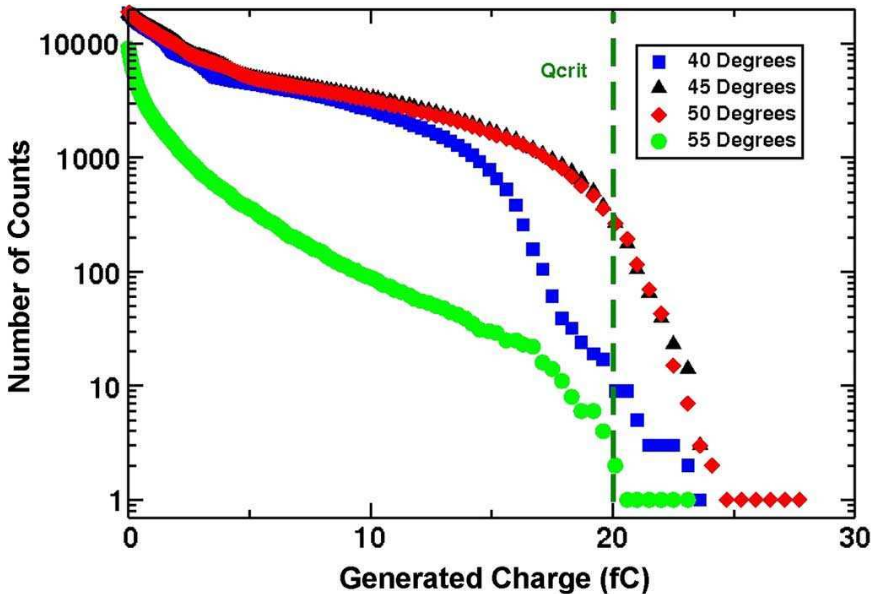


Figure 4.1: MRED simulations performed using 5.5-MeV alpha particles incident from several different incident angles. Only over this small range of angles is a 5.5-MeV alpha particle able to generate enough charge to produce a propagating SET (20 fC) (Gadlage et al., 2008)

Warren et al. showed the relationship between the deposited energy in a transistor from a single ionizing event and the charge collected at the transistor node Warren et al. (2008) by using Monte-Carlo radiation transport code coupled with SPICE circuit-level simulation. While charge can be generated and collected in multiple areas of a device, the drain area has been shown to most efficiently collect charge. A more detailed analysis would involve modeling the charge collection in multiple volumes of the device. Heavy-ion irradiations have identified that as technology scales, alpha-particle-induced transients are increasing within the drain area Gadlage et al. (2008); Narasimham et al. (2009).

Determining an exact probability of generation for each cell is beyond the scope of this dissertation. However, the soft error rate of the cell library used in this study, FreePDK45, has been previously estimated (Alexandrescu et al., 2011). In the study by Alexandrescu et al., the cells are characterized using a nuclear database that lists the various particles produced by the interaction of energetic neutrons with the cell atoms. The cross section for a sensitive volume is calculated, and a range of transient current pulses are injected for each cell. The failure-in-time (FIT) is calculated as one SET during a billion working hours for a MegaCell which is 2^{20} (1,048,576) identical cells. The results from the Alexandrescu study can be seen in Table 4.1.

4.2 Probability of Propagation

Error Propagation Probability (EPP) is the likelihood that an error in one component propagates to output of the circuit. This metric represents the logical masking that occurs when an error on a logic gate does not have a direct path to an output node. EPP can be calculated empirically or statistically. It can be calculated empirically by running fault injection simulations to determine the number of input combinations that result in an error by using Equation 1:

$$EPP = \frac{\Sigma \text{ input combinations that allow an internal error to reach an output node}}{\Sigma \text{ all possible input combinations}} \tag{4.2}$$

In this method, a bit flip is simulated by inserting XOR gates at desired fault locations with the original circuit node as one input and a fault enable signal at the other input. The

Table 4.1: Overall SER figures (assuming equiprobable states) for a selection of combinational cells. The SET pulse-width threshold is set to 50 ps (Alexandrescu et al., 2011).

Cell	**SER (FIT)**		**Cell**	**SER (FIT)**
AND2_X1	145		NOR2_X1	116
AND2_X2	126		NOR2_X2	29.2
AND2_X4	112		NOR2_X4	9.85
AND3_X1	142		OR2_X1	196
			OR2_X2	133
AND4_X1	122		OR2_X4	107
AND4_X2	109			
AND4_X4	102		OR3_X1	174
			OR3_X2	95.6
NAND2_X1	98.3		OR3_X4	80.9
NAND2_X2	21			
NAND2_X4	2.26		OR4_X1	136
			OR4_X2	61.9
NAND3_X1	129		OR4_X4	51.9
NAND3_X2	38.5			
NAND3_X4	0		XOR2_X1	405
			XOR2_X2	229
NAND4_X1	126			
NAND4_X2	38.4		XNOR2_X1	347
NAND4_X4	6.17		XNOR2_X2	198

output of the XOR gate becomes the input to the next stage in the circuit. The output of the circuit is compared to the expected output to see if an error occurred. Exhaustive fault injection using this method becomes an intractable problem for large circuits. Additionally, extracting key information about the path that the error uses to propagate would not be feasible. To address these issues, the propagation probability has been commonly approximated through statistical fault injection and probabilistic graphical models.

4.2.1 Statistical Fault Injection

The probability of propagation of soft errors from a struck node to a primary output can be calculated through exhaustive fault simulation by emulating the circuit design in software, flipping the logic value at the input of every node independently, and comparing the value of the primary output to an original value. This method is 100% accurate but is resource-intensive. The computational time can be reduced by simulating a subset of fault stimuli chosen at random locations throughout the circuit, a widely used method known as statistical fault injection (SFI) (Ramachandran et al., 2008; Leveugle et al., 2009). The major drawback of SFI is that the decrease in computation is directly proportional to a decrease in accuracy. The minimum number of experiments necessary (n) can be computed as

$$n = \frac{4{z_{\alpha/2}}^2 \times \mathrm{AVF}(1 - \mathrm{AVF})}{w^2} \tag{4.3}$$

where $z_{\alpha/2}$ denotes the value of the standard normal variable (constant) for the confidence level 100 $\times$ (1 - α)%, AVF (architectural vulnerability factor) is the EPP, and w is the width of the confidence interval at the particular confidence level (Allen, 1990). For example (Mukherjee, 2008), the number of experiments per bit necessary for an EPP of 0.30 for a 95% confidence level with a 10% error in the EPP in either direction is $n = 4 \times 1.96^2 \times 0.3(1 - 0.3)/0.06^2 = 896.37$. Considering circuit designs with thousands of bits, the number of tests needed can extend beyond millions. Therefore, SFI would not be a reasonable method for evaluating the vulnerability of an entire space of circuit implementations.

4.2.2 Probabilistic Graphical Models

In (Parker and McCluskey, 1975), calculating the probability that the output of a general combinational network is a logic 1 given the probabilities for each input being a logic 1 was described. Proofs were given for AND, OR, and INV gates. This work was extended by (Asadi and Tahoori, 2010) to account for reconvergent paths and to estimate the probability of soft error propagation. The authors converted a cell-level verilog netlist to a *directed acyclic graph*, calculating the signal probability as it traversed the path affected by the injected fault. A directed acyclic graph (DAG) (Koller and Friedman, 2009) is a collection of vertices and directed edges where each edge connects to vertices without the presence of loops. An example of a DAG is shown in Figure 4.2. The EPP calculated using this method was 95% accurate compared to random fault injection and 4-5 orders of magnitude faster. The equations used for on-path signal propagation can be observed in Figure 4.2.

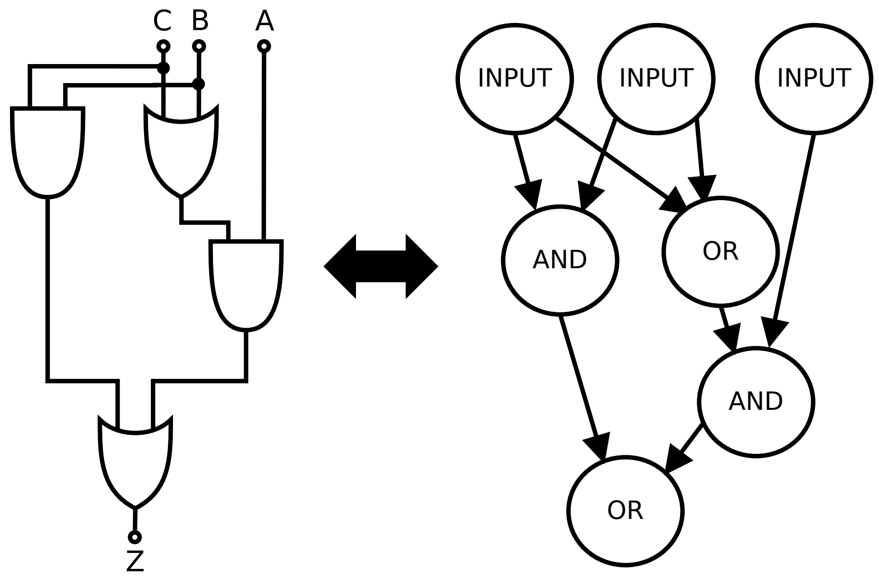


Figure 4.2: Example circuit and associated DAG representation.

4.2.3 Additional EPP equations

An expansion of the study by Asadi and Tahoori (Asadi and Tahoori, 2005) is necessary to investigate a larger set of cells, including complex functions like “exclusive-or”. The equations for the additional cells used in this dissertation were derived using the basic probabilities of gates (Parker and McCluskey, 1975) and the notion of a signed error signal

Table 4.2: Equations for computing probability at the output of a gate in terms of its inputs (Asadi and Tahoori, 2010). P_0 represents the probability that the signal value is 0. P_1 represents the probability that the signal value is 1. P_a represents the probability that the signal value has an error. $P_{\bar{a}}$ represents the probability that the signal value has an error and that error value has been inverted.

AND	$P_1(out) = \prod_{i=1}^{n} P_1(X_i)$
	$P_{\bar{a}}(out) = \prod_{i=1}^{n}[P_1(X_i) + P_a(X_i)] - P_1(out)$
	$P_a(out) = \prod_{i=1}^{n}[P_1(X_i) + P_{\bar{a}}(X_i)] - P_1(out)$
	$P_0(out) = 1 - [P_1(out) + P_a(out) + P_{\bar{a}}(out)]$
OR	$P_0(out) = \prod_{i=1}^{n} P_0(X_i)$
	$P_{\bar{a}}(out) = \prod_{i=1}^{n}[P_0(X_i) + P_a(X_i)] - P_0(out)$
	$P_a(out) = \prod_{i=1}^{n}[P_0(X_i) + P_{\bar{a}}(X_i)] - P_0(out)$
	$P_1(out) = 1 - [P_0(out) + P_a(out) + P_{\bar{a}}(out)]$
NOT	$P_0(out) = P_1(in)$
	$P_a(out) = P_{\bar{a}}(in)$
	$P_{\bar{a}}(out) = P_a(in)$
	$P_1(out) = P_0(in)$

(Asadi and Tahoori, 2010). An example of this derivation is shown in Fig. 4.3. With each output case of a logic gate specified, the probability can be calculated by summing the input equations of the cases that produce a particular output. For instance, the probability that the output of an XOR gate is 0 is equal to the probability that both inputs have an error (Case 3) plus the probability that both inputs have the inversion of the original error (Case 7) plus the probability that both inputs equal 0 plus the probability that both inputs equal 1.

The extended set of probability equations for each cell in this study has been derived and is provided in Table 4.3. Identifying the corresponding signal probability for each cell in a cell library provides an accurate view of the impact on EPP from restricting the use of cells.

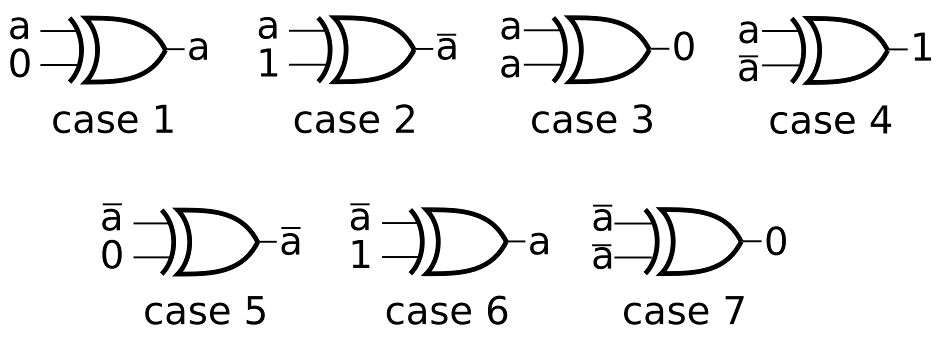


Figure 4.3: Output cases for inputs involving an error (a) or the inverse of an error ($\bar{a}$) that was propagated from a previous gate. The inputs of Cases 1, 2, 4, 5, and 6 can be mirrored to produce equivalent cases (not shown).

4.2.4 Determining Input Probabilities

Using probabilistic graph models to estimate error propagation probability requires knowledge of signal probability for each node. Each signal probability of an internal node is based on the probabilities of the preceding nodes. For the input nodes, the signal probability is based on the application that is executed on the circuit. When the application is known, the input probabilities can be included in the graph model. In this dissertation, the application for each circuit was unknown. Therefore, equiprobable states (i.e., P(0) = 0.5, P(1) = 0.5) were used to approximate the input vectors from an application.

4.3 Probability of Latching

The probability of latching is calculated as the difference of the transient pulse width and the latching window divided by the clock period (Shivakumar et al., 2002). The clock period used in this study was 1 ns (i.e., 1 GHz clock speed) and the latch-window value was taken from the specifications of a 1x drive strength D flip-flop.

Table 4.3: Probability equations for additional CMOS cells. P_0 represents the probability that the signal value is 0. P_1 represents the probability that the signal value is 1. P_a represents the probability that the signal value has an error. $P_{\bar{a}}$ represents the probability that the signal value has an error and that error value has been inverted.

NAND	$P_0(out) = \prod_{i=1}^{n} P_1(X_i)$ $P_{\bar{a}}(out) = \prod_{i=1}^{n}[P_1(X_i) + P_a(X_i)] - P_0(out)$ $P_a(out) = \prod_{i=1}^{n}[P_1(X_i) + P_{\bar{a}}(X_i)] - P_0(out)$ $P_1(out) = 1 - [P_0(out) + P_a(out) + P_{\bar{a}}(out)]$
NOR	$P_1(out) = \prod_{i=1}^{n} P_0(X_i)$ $P_{\bar{a}}(out) = \prod_{i=1}^{n}[P_0(X_i) + P_a(X_i)] - P_1(out)$ $P_a(out) = \prod_{i=1}^{n}[P_0(X_i) + P_{\bar{a}}(X_i)] - P_1(out)$ $P_0(out) = 1 - [P_1(out) + P_a(out) + P_{\bar{a}}(out)]$
XOR	$P_0(out) = P_0(a) \times P_0(b) + P_1(a) \times P_1(b) + P_a(a) \times P_a(b) + P_{\bar{a}}(a) \times P_{\bar{a}}(b)$ $P_a(out) = P_a(a) \times P_0(b) + P_0(a) \times P_a(b) + P_{\bar{a}}(a) \times P_1(b) + P_1(a) \times P_{\bar{a}}(b)$ $P_{\bar{a}}(out) = P_a(a) \times P_1(b) + P_1(a) \times P_a(b) + P_{\bar{a}}(a) \times P_0(b) + P_0(a) \times P_{\bar{a}}(b)$ $P_1(out) = 1 - [P_0(out) + P_a(out) + P_{\bar{a}}(out)]$
XNOR	$P_1(out) = P_0(a) \times P_0(b) + P_1(a) \times P_1(b) + P_a(a) \times P_a(b) + P_{\bar{a}}(a) \times P_{\bar{a}}(b)$ $P_a(out) = P_a(a) \times P_1(b) + P_1(a) \times P_a(b) + P_{\bar{a}}(a) \times P_0(b) + P_0(a) \times P_{\bar{a}}(b)$ $P_{\bar{a}}(out) = P_a(a) \times P_0(b) + P_0(a) \times P_a(b) + P_{\bar{a}}(a) \times P_1(b) + P_1(a) \times P_{\bar{a}}(b)$ $P_0(out) = 1 - [P_1(out) + P_a(out) + P_{\bar{a}}(out)]$

4.3.1 Worst-case Pulse Width

The transient pulse width can be used to compare the vulnerability of each logic cell to soft errors. Although the data from Alexandrescu et al. (Alexandrescu et al., 2011) provide a soft error rate to compare cells, the latch-window masking contribution cannot be determined because the pulse width information is not available. The *worst-case pulse width* is determined to estimate this contribution. This metric provides a way to compare the pulse width of each cell around a given charge. However, it does not factor in the probability of occurrence of an ion strike depositing the given charge.

There is extensive research that has previously been conducted in relation to modeling the pulse width of an SET in combinational logic. Messenger developed an approximate analytical solution for the pulse width generated by lightly ionizing particles (Messenger, 1982), in the double-exponential formula shown in Equation 1:

$$I_{SET}(t) = I_0(e^{-t/\tau_1} - e^{-t/\tau_2}) \tag{4.4}$$

In this equation, $I_{SET}(t)$ is the approximate pulse generated by the SET, I_0 is approximately the maximum current, τ_1 represents the collection time-constant of the junction, and τ_2 is the ion-track establishment time constant. I_0 can be represented as $\frac{Q}{(\tau_1 - \tau_2)}$. τ_1 and τ_2 are process-related factors. This equation has been used in SPICE simulations (Dharchoudhury et al., 1994) by connecting a piece-wise linear current source to the output of the affected node. Figure 4.4 shows an example schematic of a two-input NAND gate and the current source representing the DSET. The capacitance, CL, is the load capacitance with values ranging from 0 to the maximum output capacitance.

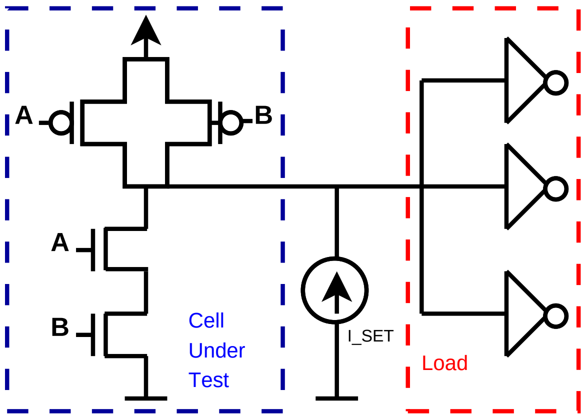


Figure 4.4: Two-input NAND gate with SET current source (Zhou and Mohanram, 2006).

The width of the SET pulse depends on the logic state of the struck cell at the time of the particle strike. If the struck device does not alter the output logic, then its effects can be disregarded. Additionally, for particles with low linear energy transfer (LET), charge collection by drift and diffusion dominate. Charge accumulated by the diffusion of minority carriers is proportional to collection depth. Since the collection depth of pMOS transistors is smaller than that on nMOS transistors, more charge, and subsequently a wider pulse, is created during strikes on nMOS transistors. This statement does not assume that nMOS transistors always have a higher soft error rate than pMOS transistors. The drain area (i.e., sensitive area) of a pMOS transistor is often larger than that of an nMOS transistors for certain cells. However, the collection depths of the devices relate to the worst-case pulse. Additionally, Zhou and Mohanram (Zhou and Mohanram, 2006) showed that increasing transistor size reduces worst-case pulse width, which suggests that the worst-case pulse width generated by the pMOS transistor would be further reduced. Therefore, in identifying the worst-case pulse width, nMOS transistor strikes were ranked higher than pMOS transistor strikes. Finally, SETs cause state changes when nMOS transistors are reverse-biased (off).

Table 4.4: Comparison of pulse widths generating from nMOS and pMOS transistors for a 180-nm technology. "PW" represents the pulse width of the cell. When NOR2 has inputs a,b = {1,1}, the result is 0 (as denoted by an asterisk) because the output voltage peaks at 0.4V (less than 1/2 of a VDD of 1.8V)

Cell	**Input**	**nMOS PW** (10^{-10}s)	**pMOS PW** (10^{-10}s)
INV	a=0	4.16	n/a
	a=1	n/a	1.94
NAND2	a,b = {0,0}	2.35	n/a
	a,b = {0,1}	4.19	n/a
	a,b = {10}	4.36	n/a
	a,b = {11}	n/a	3.33
NOR2	a,b = {00}	5.71	n/a
	a,b = {01}	n/a	1.96
	a,b = {10}	n/a	1.89
	a,b = {11}	n/a	0*

Translating the above discussion to logic, the worst-case pulse width occurs when at least one input is logic "0". For a NOR gate, the states where at least one nMOS transistor is off are: "00", "01", and "10". For "01" and "10", a strike on the off nMOS transistor would not change the logic value of the output because a strike on an off nMOS transistor can only drive signals low. Since the output in this logic state is already "0", the strike has no impact. For the "00" input state, a strike on an off nMOS transistor would change the output logic value from "1" to "0". Therefore, "00" is the most sensitive state for NOR gates. Table 4.5 shows the logic state that allows the worst-case pulse width for the cells used in this study. In some cases (e.g., NAND) there is more than one state because the inputs "01" and "10" have approximately equal effects on the output.

Table 4.5: Logic state. "AND" and "OR" have an additional sensitive state because the inverter stage is vulnerable to strikes (denoted by asterisk).

Cell	**Worst Logic State (nFET)**	**Worst Logic State (pFET)**
INV	0	1
AND2	01,10,11*	11, {01,10,00}*
OR2	00,11*	01,11,00*
NAND2	01,10	11
NOR2	00	01,11
XOR2	00	00
XNOR2	00	00

The Predictive Technology Model (PTM) (Cao et al., 2000), developed by the Nanoscale Integration and Modeling (NIMO) Group at Arizona State University, provides predictive model files for future transistor technologies that are compatible with standard circuit simulators, such as SPICE, and scalable with a wide range of process variations. The critical charge and time constants were taken directly from (Zhou and Mohanram, 2006).

The FreePDK45 design kit (Stine et al., 2007) supplies techfiles, display resources, design rules and scripts to permit layout design and rule checking for a generic 45-nm process. It has been used to characterize the vulnerability of logic cells to soft errors (Alexandrescu et al., 2011). The study in this dissertation used a subset composed of five two-input logic functions.

The following assumptions were made for a particle strike. A particle with an LET of 1 MeV-cm^2/mg deposits approximately 10 fC/μm of electron-hole pairs along its track (Dodd and Massengill, 2003; Mavis and Eaton, 2002). Limiting the charge collection depth to $2\mu m$ and an environment of particles that do not exceed 15 MeV-cm^2/mg (Zhou and Mohanram, 2006), charge deposition was simulated with a peak value of 300 fC and a minimum value of 15 fC.

Table 4.6: Transient pulse widths obtained from simulating a 15 MeV-cm^2/mg ion strike on FreePDK45 cells. The pulse width of the cell normalized to an inverter cell is also included.

Cell	Pulse Width (ps)	Normalized Pulse Width
INV	464	1
NAND2	475	1.02
NOR2	643	1.39
AND2	631	1.36
OR2	811	1.75
XOR2	808	1.74
XNOR2	639	1.38

4.4 Tools for Modeling and Evaluating Soft Error Rate

Several methods that incorporate the various masking factors have been proposed to model and evaluate SER, such as the SEUTool(Massengill et al., 2000), the SERA tool (Zhang and Wang, 2006), the SEAT tool (Ramanarayanan et al., 2009), the SERD algorithm (Rao et al., 2007), the MARS-C tool (Miskov-Zivanov and Marculescu, 2006), and the MARS-S tool (Miskov-Zivanov, 2007). Each method essentially examines a synthesized circuit described as a netlist. Of the tools previously mentioned, only the SEAT-LA soft error estimation tool for combinational logic and the SEUTool can be used specifically for designs that use SER-characterized cell libraries. Although the circuit parameters of the netlist (e.g., gate sizes (Zhou and Mohanram, 2006)) could be modified to reduce SER, the original topology of the circuit can have inherent vulnerabilities from the cascading of different library cells. Also, circuit-level techniques can impose drastic area, delay, or power penalties if used throughout the design without regard to the circuit's functionality. Selective hardening techniques have been used to address this problem (Srinivasan et al., 2005) but seldom been exploited in logic synthesis. A reliability-aware logic synthesis is needed to choose among the various library cells to leverage the circuit-level mitigation strategies for soft errors (Tosun et al., 2005). Mitigation during logic synthesis can reduce the baseline SER of the design and does not preclude the use of other techniques, such as radiation-hardened latches and gates, temporal or spatial redundancy, error detection and correction (EDAC) circuitry, or triple modular redundancy (TMR).

4.4.1 SEUTool

SEUTool was created by Massengill et al. at Vanderbilt University (Massengill et al., 2000). With this tool, nodes within the circuit hierarchy are identified and ranked according to their drive capabilities, capacitance (including fanout), and critical area.

The probability of generation is described as the ratio of the critical area of node to the total circuit area. The relationship between collected charge and the shape of the noise signal pulse is determined based on the total nodal capacitance and the pull-up/pull-down current drive at each node. A path analysis is performed to determine whether the noise signal at each node can propagate through an active path to a latch or register storage location. The probability of latching is determined by using a pulse capture model for each latch based on the electrical characteristics of the latch and a simulated pulse-width-to-latch arrival time.

This tool is useful for coupling the probabilities of generation, propagation, and latching. However the tool only tracks the worst-case path, i.e., the path to a latch with the highest probability of capture, which obscures the topological information necessary to rank all nodes.

4.4.2 SEAT

Ramanarayanan et al. (Ramanarayanan et al., 2009) model the three common masking effects (logical, electrical, latch-window) concurrently by: (1) capturing the current-voltage transfer characteristics for multiple current pulses at the input nodes, (2) accumulating the delay information for logic cells, (3) estimating the timing window for flip-flops by sweeping a voltage pulse of a specific width and height at the input of the flip-flop using HSPICE simulation, and (4) modeling the voltage glitch propagation using the mathematical equations proposed in the work. This tool is useful in terms of calculating the masking effects concurrently and efficiently. However, this tool has only been used in conjunction with HSPICE to pre-characterize a cell library. This estimation ignores mechanisms that cannot be captured in a SPICE-level model such as pulse quenching (Albhin et al., 2009). Additionally, the cell library used with this tool contained only four cells, which does not account for the trends that could be observed using a broader and more complex range of cells.

4.4.3 TFIT and SoCFIT

IROC Technologies developed commercial software (i.e., TFIT, SOCFIT) that predicts the soft error rate for commercial IC's. The TFIT software is a simulation tool that predicts the FIT of each cell in a library of a specific technology using device-level analysis of particle interactions. The SOCFIT software predicts the failure rate (FIT) and the impact of functional, timing, and logic de-rating using either RTL or gate netlist representation. The details of the analysis are intentionally obscured by IROC Technologies because their software is commercially available. The results from these tools were used in part to validate reliability rankings for logic cells.

Chapter 5

Cell and Circuit Characteristics that Impact Reliability

The feasibility of reliability-aware synthesis methods is dependent upon the ability to estimate reliability at an early design stage. This chapter investigates the relationship between circuit characteristics and reliability. Key findings of this dissertation include the identification of logic depth as strongly correlated to the error propagation probability of the ISCAS85 Brglez and Fujiwara (1985) benchmark circuits.

The use of circuit characteristics as a predictor of soft error rate has been covered in recent literature. In relation to the probability of a soft error being generated, cell supply voltage, sizing parameters, and transient waveform descriptions have been used to predict the resulting upset rate (Ness et al., 2007; Zhang and Wang, 2006). In relation to the probability that an error propagates, Krishnaswamy et al. showed that the likelihood of an error propagating is directly related to signal observability and probability (Krishnaswamy et al., 2007). The authors incorporated these characteristics to improve reliability at an early design stage. However, the authors do not address the synthesis parameters that alter signal observability and probability. Additionally, the signal observability and probability are passive characteristics in that they are indirectly manipulated by the synthesis tool, while logic depth and node fanout are actively limited by synthesis algorithms.

To date, there has not been a comprehensive study of the correlation between the circuit characteristics of combination logic (e.g., logic depth and node fanout) and reliability. Such an analysis is necessary to gauge the effectiveness of reliability-aware synthesis techniques and predict logical trends. This chapter fills the void by calculating the reliability for multiple circuits and identifying how these results relate to common circuit characteristics.

5.1 Identifying Trends

In order to determine the relationships between circuit characteristics, constraints, and EPP, multiple benchmark circuits were synthesized with various performance constraints and restrictions on the use of certain cells within the cell library. The logic depth and fanout of each logic node were identified for each netlist. Then, the netlists from these synthesized designs were used as input to the EPP tool. These data enabled an investigation to determine the impact on EPP from: (1) area-constrained vs. timing-constrained implementations, (2) map effort, (3) incrementally increasing delay, and (4) removing cells from the cell library.

For this study, *logic depth* of each node is defined as the number of nodes in the shortest path from that circuit node to an output node. *Fanout* is defined as the number of circuit nodes whose input is connected to the output of the circuit node of interest. The Pearson product-moment correlation coefficient (Rodgers and Nicewander, 1988) (Equation 5.1) was used to determine the linear relationship between: (1) logic depth and EPP and (2) fanout and EPP. In Equation 5.1, r is the sample correlation coefficient, x_i and y_i are the sample variables, $\bar{x}$ and $\bar{y}$ are the sample means, s_x and s_y are the sample standard deviations.

$$r = \frac{1}{n-1}\Sigma\frac{(x_i-\bar{x})(y_i-\bar{y})}{s_x s_y} \tag{5.1}$$

Through this analysis, the viability of using logic depth and fanout as predictors for reliability can be ascertained. Constraints like area and delay have consistent trends with these characteristics, so this result indirectly provides a prediction on reliability for a given constraint.

5.2 Circuit Characteristics

The logic depth, fanout, and cell frequency results for individual nodes in the ISCAS85 combinational logic benchmark circuits are included in this section. Additionally, this section contains the analysis of the impact of synthesis compiler constraints on the error propagation probability of these circuits.

5.2.1 Logic Depth

Figures 5.1 and 5.2 show the correlation trends for logic depth vs. EPP and fanout vs. EPP respectively. Logic depth has a strong negative correlation with EPP for both area-constrained and timing-constrained implementations. This result indicates that as logic depth decreases, the probability that a fault reaches an output node increases. This result was expected since a shorter path has less opportunity for logical masking.

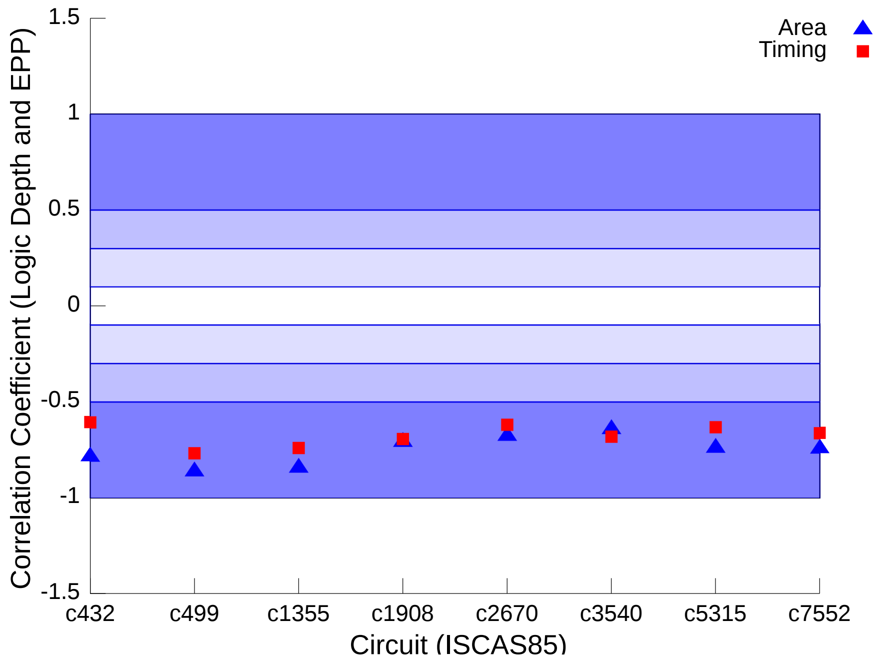


Figure 5.1: Correlation coefficient between logic depth and EPP. Ranges for the correlation coefficient, r, are separated by shaded regions: $|r| > 0.5$ represents a strong correlation, $0.3 < |r| < 0.5$ represents a medium correlation, $0.1 < |r| < 0.3$ represents a weak correlation, $|r| < 0.1$ represents no correlation. A negative r implies an inverse relationship.

5.2.2 Fanout

Fanout shows a weak negative correlation with EPP (Fig. 5.2) across multiple benchmark circuits. Additionally, the correlation varies by circuit from a medium negative correlation to no correlation. This result may not be intuitive since a greater fanout implies that more fault paths are created when that node is struck. However, each fault path has the ability to interact with another fault path and nullify the fault in a process known as *reconvergence*. Reconvergence of paths reduces the likelihood of an error in some cases and therefore prevents a linear relationship between fanout and EPP.

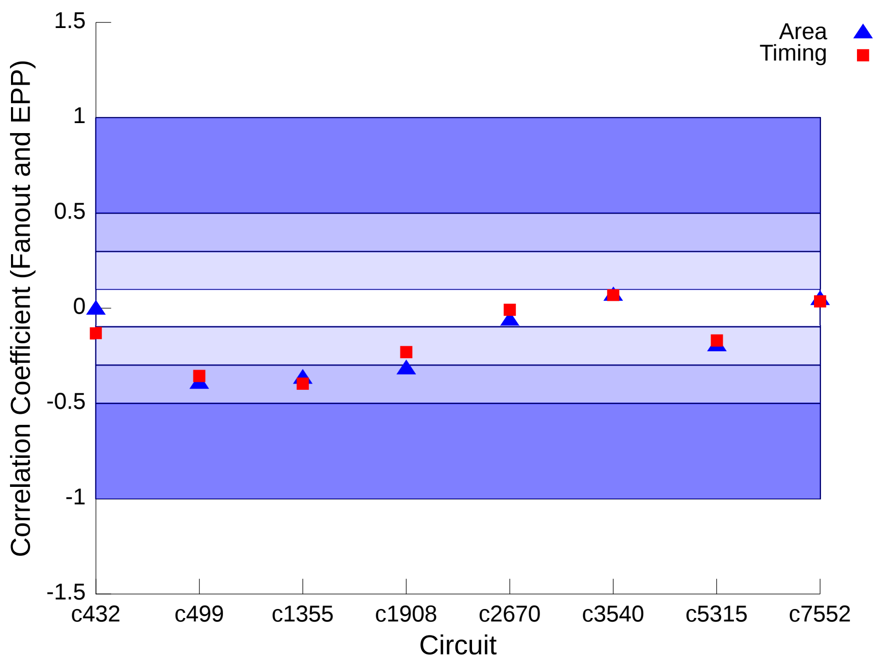


Figure 5.2: Correlation coefficient between fanout and EPP. Ranges for the correlation coefficient, r, are separated by shaded regions: $|r| > 0.5$ represents a strong correlation, $0.3 < |r| < 0.5$ represents a medium correlation, $0.1 < |r| < 0.3$ represents a weak correlation, $|r| < 0.1$ represents no correlation. A negative r implies an inverse relationship.

5.3 Cell Characteristics

In addition to analyzing the impact that the layout characteristics of a circuit have on reliability, the physical characteristics of the cell also become important. In this section, the drain area and capacitance of a logic cell are analyzed.

5.3.1 Drain Area

The region of a logic cell that is susceptible to ionization, i.e., *sensitive area*, must be considered in the analysis of the soft error rate. The sensitive area influences the spatial probability that a particle will strike a device. If two cells have the same logic function, physical composition, and general layout, but one cell has a larger sensitive area, then the cell with the larger sensitive area will have a higher frequency of particles that deposit charge in the cell. The sensitive area must be determined for each cell in order to compare the overall sensitivity towards soft errors.

The sensitive area of a logic cell can be approximated as the reverse-biased drain area

(Dodd and Massengill, 2003) though this approximation may underestimate the sensitive area for nanometer technologies. For low-energy neutron strikes, the sensitive drain areas can be ascertained by identifying the drains that are *blocked* from the source and ground voltages (Nicolaidis, 2011), as shown in Figure 5.3. The blocked drain area is important for this study because cell libraries have multiple implementations of a logic function, each with different performance characteristics. The drain area for each of these implementations may differ and therefore the SER of the cells will differ. As an example, Figures 5.4 and 5.5 represent a 1x and 2x drive strength two-input NAND cell respectively. The NAND2x2 cell has more sensitive area than the NAND2x1 cell. This implies that the NAND2x2 cell will have more particle strikes because the target area is larger. While the probability of particle interaction is higher for NAND2x2, the NAND2x1 cell has a higher SER because its capacitance, and therefore its critical charge, is much higher. This phenomenon is addressed in Section 5.3.2. Still, the drain area can be an important metric when deciding between two logically-equivalent cells that also have the same capacitance at the struck node. For instance, the AND2x2 and AND4x1 cells (shown in Figures 5.6 and 5.7) have the same input capacitance and can be used to represent the same logic function (assuming two sets of inputs of the AND4x1 cell are tied together). However, the blocked drain areas of the AND2x2 cell are sized greater than the blocked drain areas of the AND4x1 cell. Since the probability of occurrence is higher for the AND2x2 cell while the critical charge is constant, the SER for the AND2x2 cell should be higher. This result is validated by the SER values in Table 4.1.

5.3.2 Capacitance

The generation of a transient pulse is dependent on the capacitance at the struck node. As a nodes capacitance increases, the width of the transient pulse decreases. Three methods (shown in Figure 5.8) to increase the capacitance at a node through standard cell selection are evaluated in this study: (1) selecting a higher drive strength, (2) selecting a cell with more inputs than necessary and connecting the extra inputs to valid inputs, and (3) inserting a subcircuit using different logic gates that is functionally equivalent to the replaced cell. Increasing the drive strength is equivalent to manually sizing the transistor for soft-error

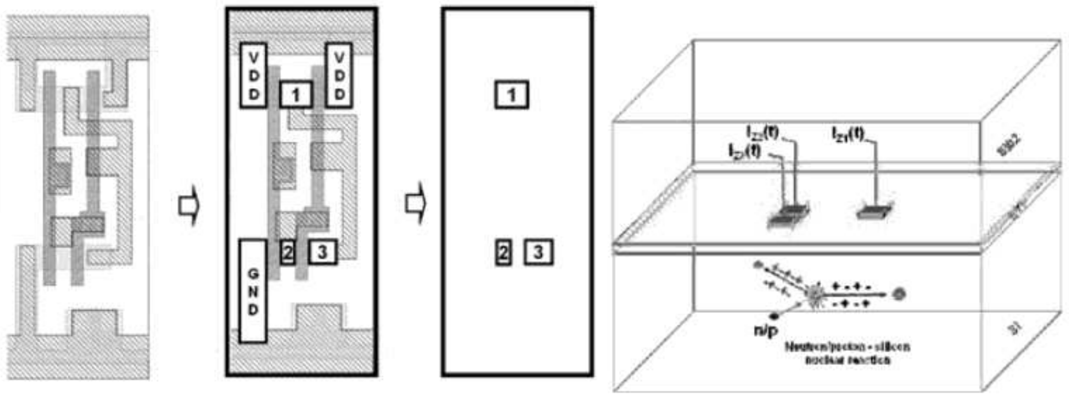


Figure 5.3: Sensitive regions of NAND2 cell (Nicolaidis, 2011). Zone 1 represents sensitive area that causes output to flip from 0 to 1. Zones 2 and 3 represent sensitive area that causes output to flip from 1 to 0.

immunity.

The width of the transient pulse that is generated from an ion strike on a transistor node is impacted by the capacitance of that node. The minimum charge needed to upset a logic node, i.e., *critical charge*, has been approximated as the product of the total capacitance, C_i, at a given node by the power supply voltage, V_{DD} (shown in Equation 5.2) (Nicolaidis, 2011):

$$Q_{crit} = C_i \times V_{DD} \tag{5.2}$$

As the capacitance at a given node increases for a given supply voltage, the critical charge increases as well.

The maximum charge that can be deposited into a transistor node has been approximated as the product of the linear energy transfer, L_{Δ}, and the charge collection depth, D_{coll}, (shown in Equation 5.3).

$$Q_{max} = L_{\Delta} \times D_{coll} \tag{5.3}$$

An LET of 1 MeV cm^2/mg has been approximated to deposit 10 fC/μm of charge along its path (Dodd and Massengill, 2003; Mavis and Eaton, 2002). The range of LET for the particles in this study (i.e. terrestrial environment) is 1-15 MeV cm^2/mg. Therefore the

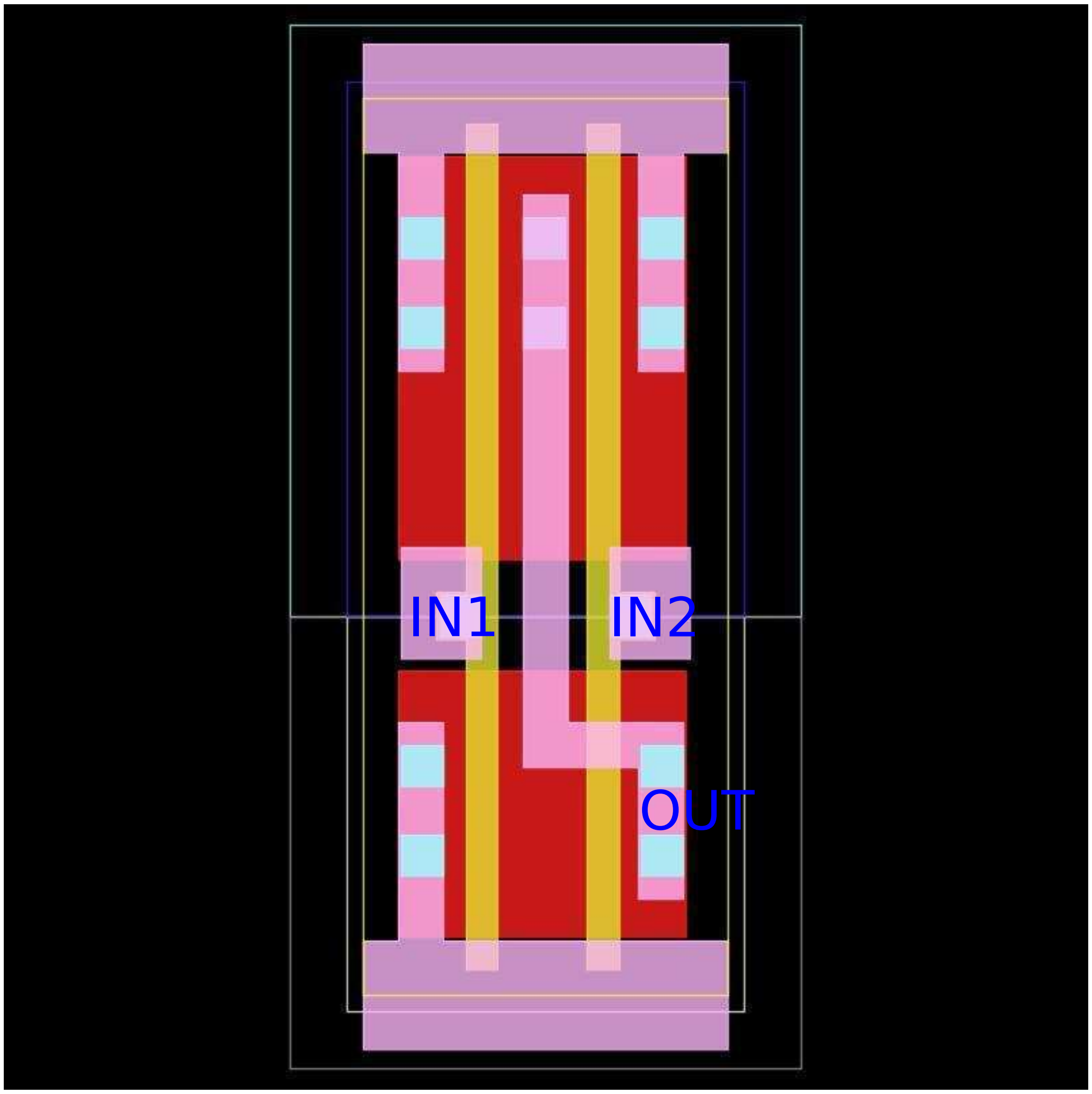


Figure 5.4: Layout of NAND2x1 cell in the FreePDK45 cell library.

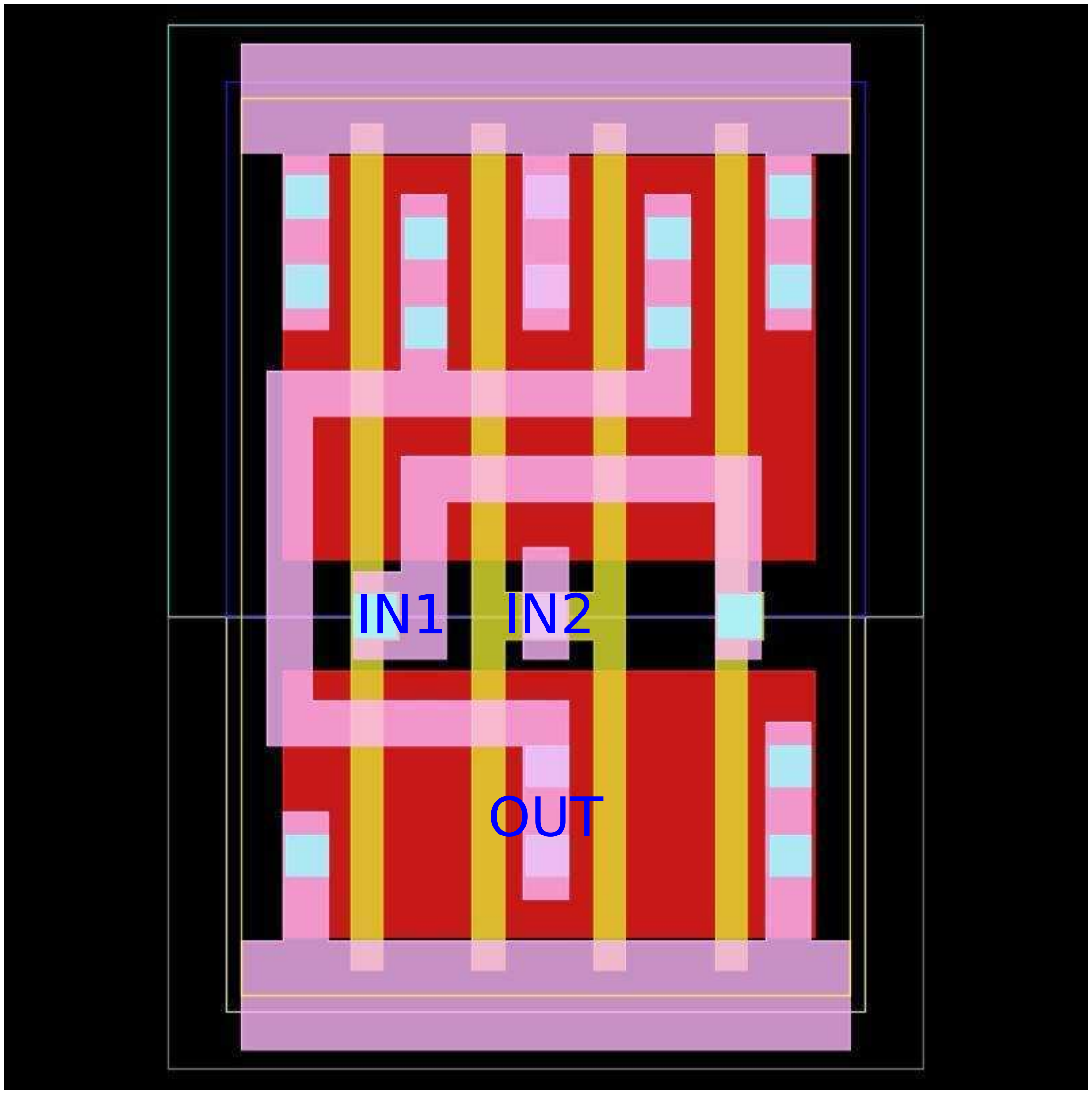


Figure 5.5: Layout of NAND2x2 cell in the FreePDK45 cell library.

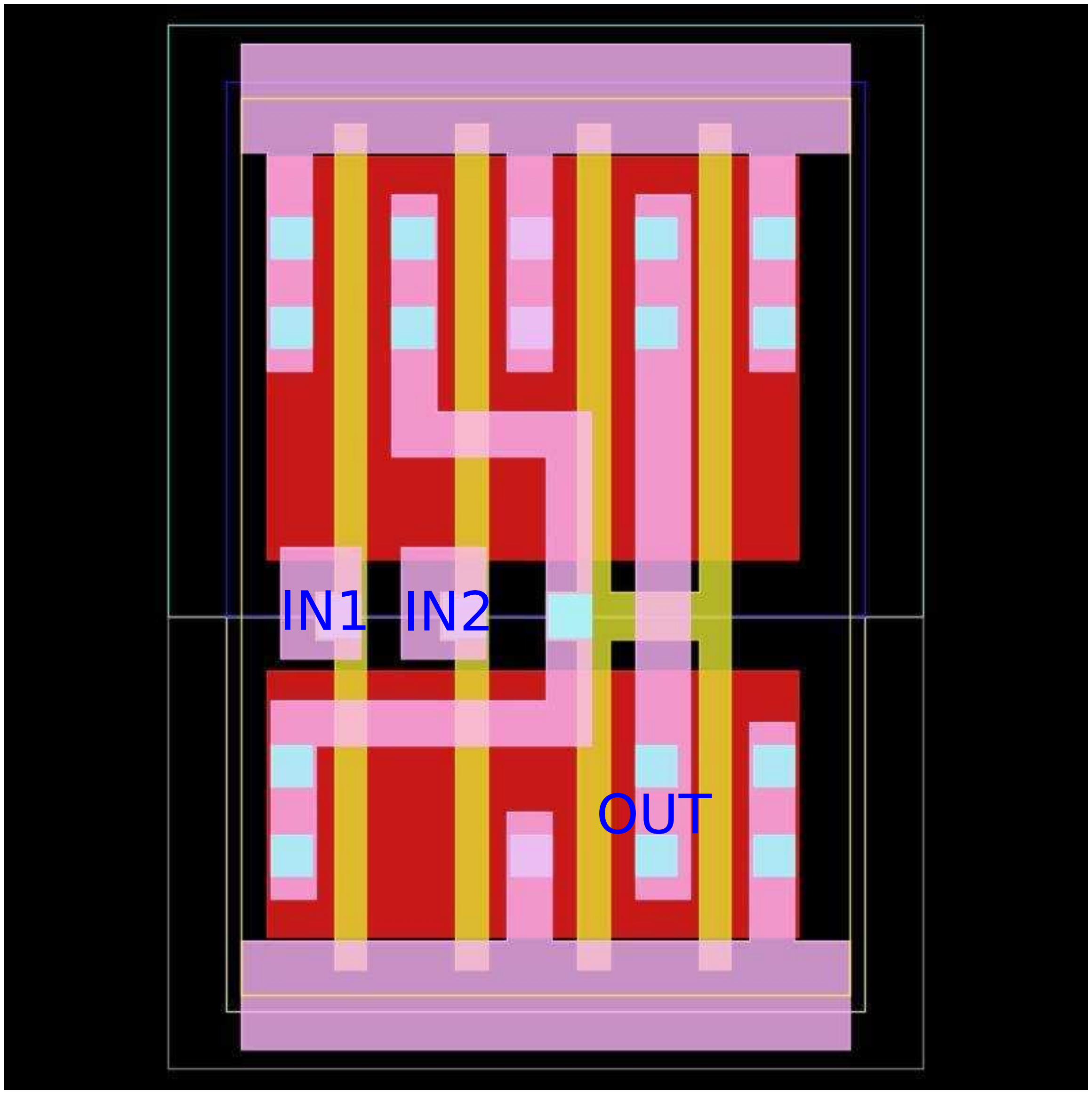


Figure 5.6: Layout of AND2x2 cell in the FreePDK45 cell library.

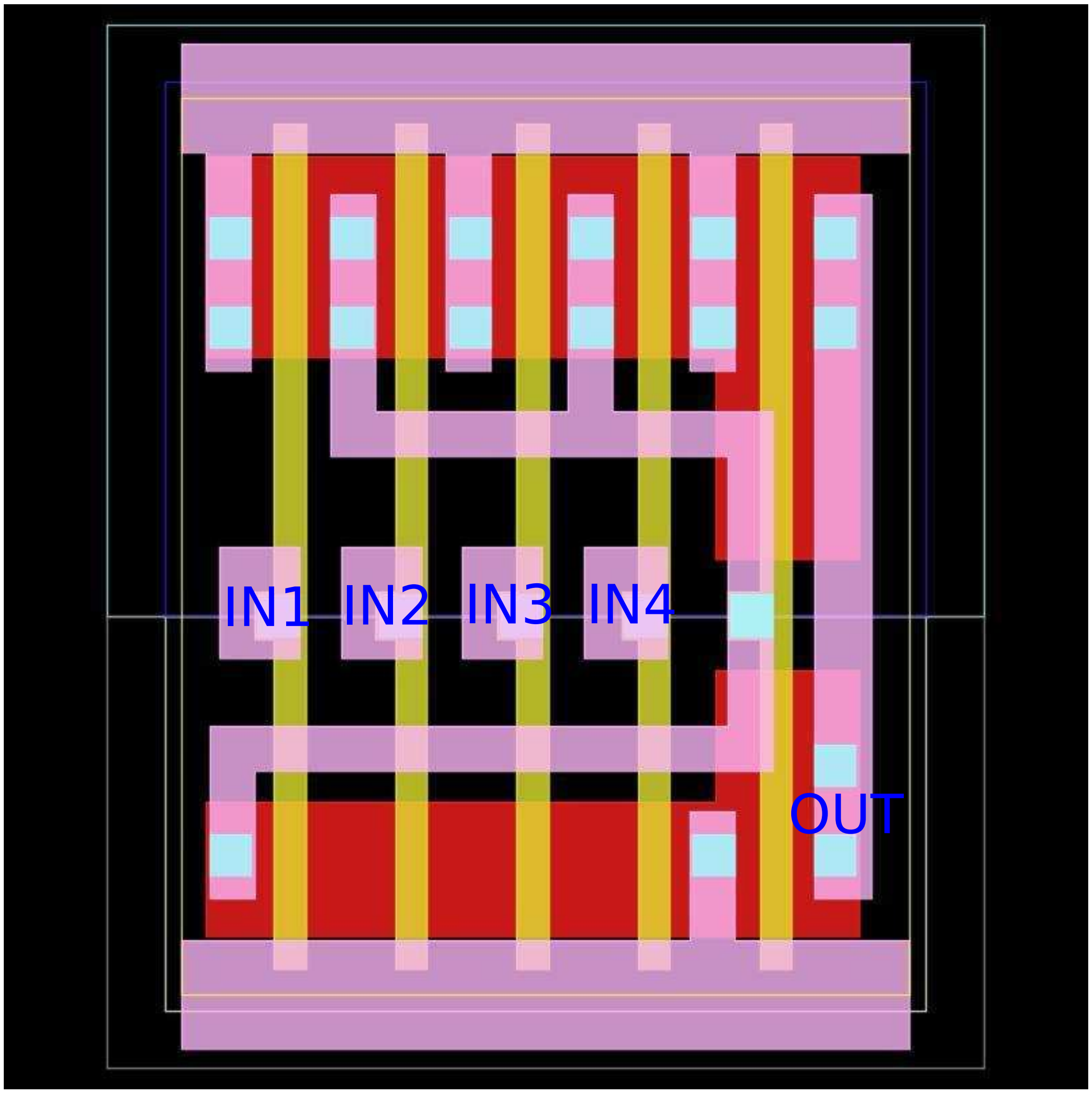


Figure 5.7: Layout of AND4x1 cell in the FreePDK45 cell library.

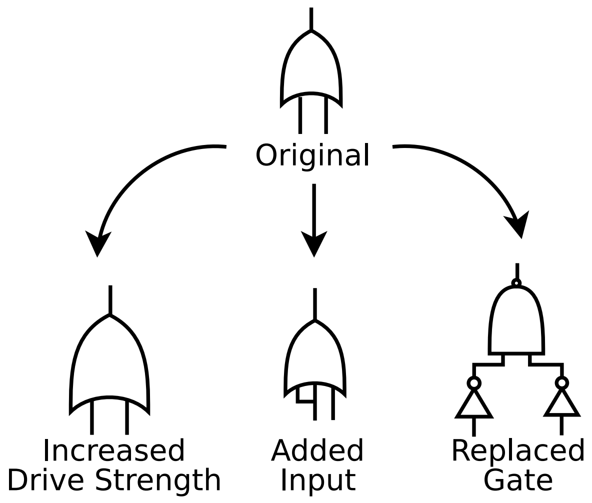


Figure 5.8: Selective node hardening techniques that use standard cells.

upper bound for charge that is needed to simulate a transient pulse is 30 pC, assuming a charge collection depth of 2 μm. The transient pulse width has been approximated using a SPICE double-exponential current source connected to the struck transistor with Equation 5.4 (Messenger, 1982; Zhou and Mohanram, 2006).

$$I_{SET}(t) = \frac{Q}{(\tau_1 - \tau_2)}(e^{-t/\tau_1} - e^{-t/\tau_2}) \tag{5.4}$$

The differences in internal capacitance between logic cells partially accounts for the differences in transient pulse width. Amongst simple cells with the same logic function but different drive strengths (e.g., NAND2x1 vs. NAND2x2), the cells with the higher drive strengths are composed of more transistors in parallel, and therefore have higher internal capacitance (see Figure 5.9). As previously stated, a higher internal capacitance translates to a smaller pulse width.

Amongst complex cells with the same logic function but different drive strengths (e.g., AND2x1 vs. AND2x2), the cells with the higher drive strengths are composed of either: (1) more transistors in parallel, or (2) larger transistors. Both cases imply higher internal capacitance. In Figure 5.10, the AND cell is separated into its NAND stage and inverter stage. The 2x drive strength AND cell has double the transistor size for the NAND stage and twice as many transistors in the inverter stage.

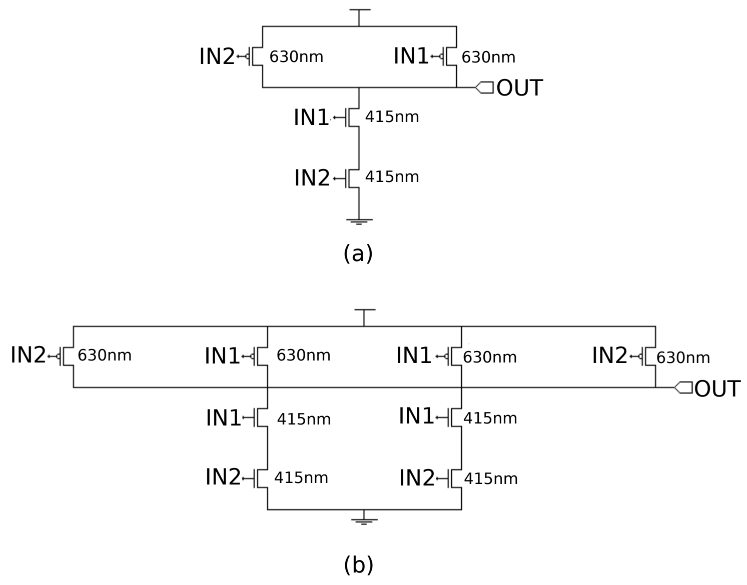


Figure 5.9: Schematic designs for the (a) NAND2x1 and (b) NAND2x2 cells. The extra transistors in parallel for the NAND2x2 cell add to the capacitance, and therefore the critical charge of each node compared to the NAND2x1 cell. Adapted from (Stine et al., 2007)

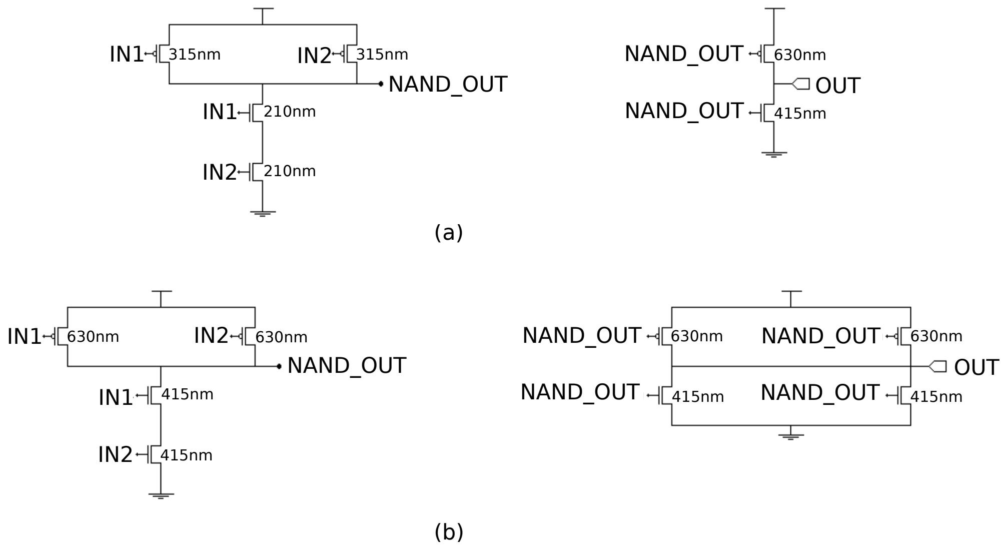


Figure 5.10: Schematic designs for the (a) AND2x1 and (b) AND2x2 cells. The larger transistor sizes in the NAND stage and the extra transistors in parallel for the Inverter stage of the AND2x2 cell add to the capacitance, and therefore the critical charge of each node compared to the AND2x1 cell. Adapted from (Stine et al., 2007)

The four-input AND cell (Figure 5.11b) can be used to implement a two input AND cell by tying two inputs together. Coupling the inputs has the same effect on capacitance as using larger transistors. The two-input AND cell with 2x drive strength (Figure 5.11a) therefore has an equivalent capacitance for the NAND stage of the cell, but double the capacitance for the inverter stage of the cell. This difference has an effect on performance and therefore increases a designer's options for reliability-aware synthesis.

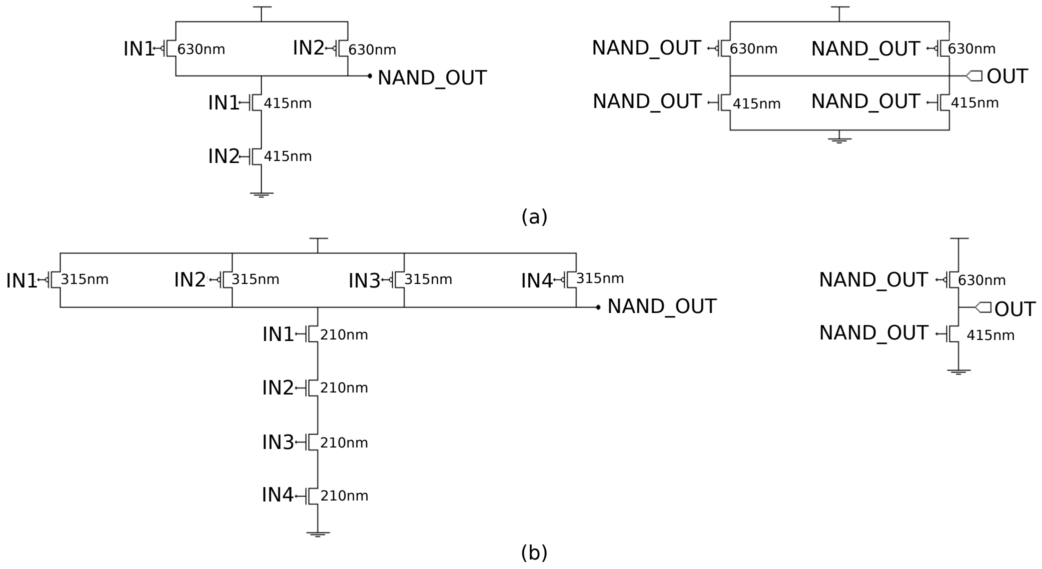


Figure 5.11: Schematic designs for the (a) AND2x2 and (b) AND4x1 cells. When tying two sets of the inputs of the AND4x1 together, the capacitance, and therefore the critical charge of each transistor is equivalent to that of the AND2x2 cell. The inverter stage of the AND2x1 has larger transistors and a greater critical charge. Adapted from (Stine et al., 2007)

Chapter 6

Synthesis Constraints that Impact Reliability

In order to analyze the soft error rate of a circuit, the original topology of the circuit must be considered. The topology of the circuit is guided by the design constraints and can have inherent vulnerabilities from the cascading of different library cells. An analysis of the impact of design constraints on topology and therefore reliability can aid in developing techniques that mitigate soft errors through topological modifications. Mitigation during logic synthesis can reduce the baseline SER of the design and does not preclude the use of other techniques (Nicolaidis, 2005), such as radiation-hardened latches and gates, temporal or spatial redundancy, error detection and correction (EDAC) circuitry, or triple modular redundancy (TMR).

Re-synthesis techniques have been introduced as a way to improve the reliability of synthesized circuits without the expense of re-writing pre-existing synthesis tools (Almukhaizim et al., 2006; Krishnaswamy et al., 2007). These techniques involve synthesizing a circuit using standard tools, identifying vulnerable nodes, and re-mapping those nodes to have less vulnerability. None of these studies have investigated the impact that the circuit's topology has on reliability and therefore the effectiveness of their technique. The original topology is a function of the constraints assigned during synthesis. For instance, topologies derived from reducing delay in a circuit will most likely contain shorter paths for a signal to reach an output node when compared to topologies derived from reducing area. Correlating reliability to synthesis variables provides a means to estimate the effectiveness of re-synthesis techniques. Additionally, such an analysis is necessary to identify soft-error mitigation techniques that exploit topological trends associated with a given design constraint.

To date, there has not been a comprehensive study of the impact that logic synthesis constraints have on reliability. Such an analysis is necessary to gauge the effectiveness of reliability-aware synthesis techniques and predict logical trends. This chapter investigates the impact that area and delay optimizations have on the topology, and subsequently the reliability of a circuit. The key results from this chapter show that different synthesis constraints will map the logical sensitivity onto different circuit topologies in a consistent way. This analysis also shows that area-constrained implementations are more likely to propagate errors, with as much as a 50% greater probability, but contain a smaller variance within a circuit design.

6.1 Identifying Trends Between Constraints and Reliability

In order to determine the relationships between design constraints and EPP, the ISCAS85 (Brglez and Fujiwara, 1985) and LgSynth91 (Yang, 1991) combinational logic benchmark circuits were synthesized with various performance constraints, and restrictions on the use of certain cells within the cell library. The ISCAS85 combinational logic benchmark circuits have cell-level netlists that approximately range from 250 to 2500 logic cells. The LgSynth91 combinational logic benchmark circuits have cell-level netlists that approximately range from 5 to 7500 logic cells. The netlists from these synthesized designs were used as input to the EPP tool described in Chapter 4. These data enabled an investigation to determine the impact on EPP from: (1) area-constrained vs. timing-constrained implementations, (2) map effort, (3) incrementally increasing delay, and (4) removing cells from the cell library.

The impact of area-constrained optimizations vs. timing-constrained optimizations was investigated by synthesizing the minimum-area implementation and the minimum delay implementation for each circuit. Optimizations occur through repetitive algorithmic steps which affect the topology of a circuit with uniformity. The optimizations for minimum area differ from those for minimum delay and therefore create different topological patterns. The impact of these patterns can be identified through this investigation.

The mapping effort setting of the Cadence Encounter RTL compiler was also studied. This setting allows for circuit designs to be synthesized with low, medium (default) and high map efforts. This attribute corresponds to the exhaustiveness of the search for the optimal circuit implementation. A high map effort configures the compiler to perform the most exhaustive search and consequently, takes the longest to compile. By varying this setting, the impact of mapping effort can be observed. Since Cadence Encounter RTL compiler does not allow specification of an area constraint (i.e., the compiler always finds the minimum area after other constraints are met), map effort provides an indirect way to observe trends related to increasing area constraints.

Trends related to increasing timing constraints can be directly observed by providing a target maximum delay to the compiler. The impact of timing-constrained optimizations was analyzed by synthesizing a circuit with the maximum (i.e., unconstrained) delay, the minimum possible delay, and three intermediate delay values. The step size was determined by dividing the slack between the minimum and maximum delay by four. Using this iterative approach provides a means to determine whether tighter timing constraints consistently impact reliability in the same direction.

The impact on EPP of removing cells from a cell library was studied by synthesizing the minimum-area implementation of each circuit while restricting the use of one cell per synthesis. For example, the c432 circuit was synthesized seven times. One implementation was the minimum-area implementation using all cells in the library. The remaining six implementations were synthesized with one of six possible cells (AND, OR, NAND, NOR, XOR, XNOR) removed. The Cadence RTL compiler prohibits the use of a cell library that does not contain at least one inverter, therefore the inverter could not be removed. Coupled with the frequency of cell usage from the unaltered library, this information explains: (1) whether certain logic cells have an inherent impact on the EPP of a circuit and (2) whether removal of a cell causes the compiler to choose implementations that are more reliable.

6.2 Area- vs. Timing-Constrained Implementations

This section examines the topological trends associated with the the ISCAS85 benchmark circuits and the LGSynth91 benchmark circuits that are constrained for area and delay.

6.2.1 Cell Frequency

The EPP formulas discussed in Chapter 4 imply that the overall EPP of the circuit is impacted by the frequency of usage of each gate. For instance, a fault on one input bit of a 2-input XOR gate (Cases 1, 2, 5, and 6) always results in an error at the output of the

XOR gate. However, a fault on one input bit of an 2-input AND gate results in an error at the output of the AND gate in only 50% of cases (when the input bit without the fault is 1).

The frequency of usage of each cell is affected by synthesis constraints and optimizations. Figures 6.1 and 6.2 show the frequency of cell usage for the ISCAS85 benchmark circuits when optimized for area and timing. Despite differences in topology among the benchmark circuits studied, clear synthesis trends can be identified. The area-constrained implementations use NAND and NOR cells infrequently while the timing-constrained implementations have a more balanced distribution of cells. Additionally, the area-constrained implementations have a greater frequency of XOR/XNOR cells than the timing-constrained implementations.

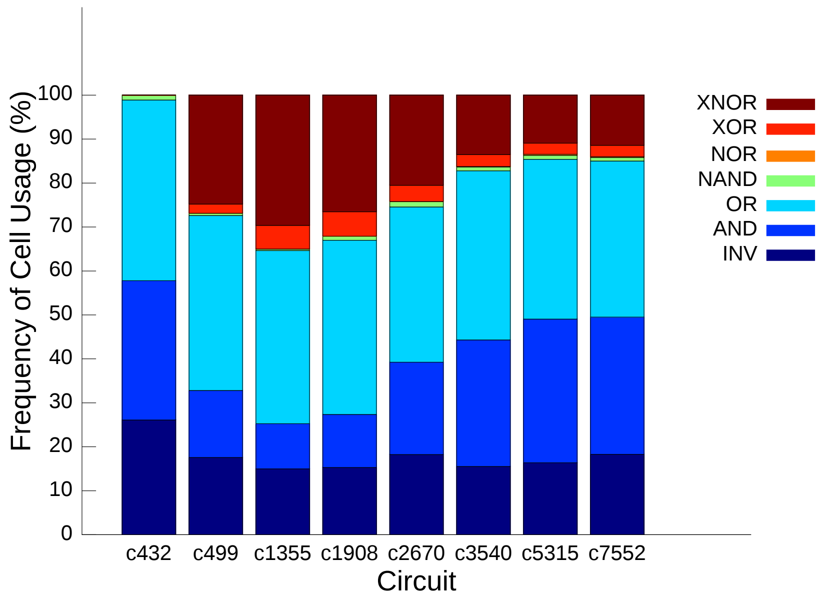


Figure 6.1: Frequency of cell usage for area-constrained implementations of selected ISCAS85 benchmark circuits.

The usefulness in mapping cell frequency trends to a given constraint derives from the inherent vulnerability of each cell. For example, the results from Alexandrescu et al. (Alexandrescu et al., 2011) showed that for the FreePDK45 cell library, the XOR cell has a highest SER of any logic cell used in their study. Since the area-constrained implementations have a greater percentage (and total number) of XOR cells, the area-constrained implementations can be predicted to have a higher SER.

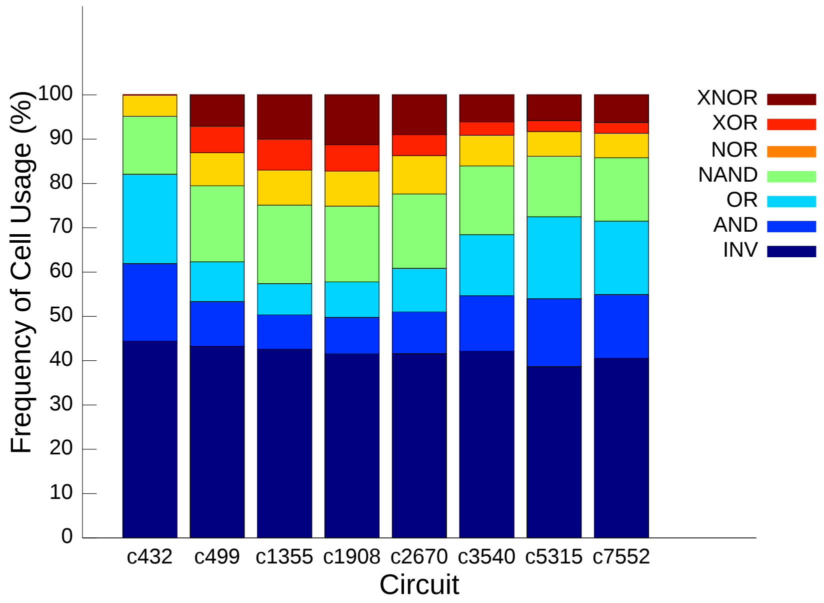


Figure 6.2: Frequency of cell usage for delay-constrained implementations of selected ISCAS85 benchmark circuits.

6.2.2 Reliability Comparisons by FIT

The MegaFIT (10^6 FIT) rate of the FreePDK45 cell library was calculated by Alexandrescu et al. Alexandrescu et al. (2011). Combining the data from that study with the cell frequency data introduced in the previous section shows that the area-constrained implementations have a higher FIT rate than the timing-constrained implementations (Figures 6.3 and 6.4).

6.2.3 Reliability Comparisons by EPP

In order to understand the impact that topology has on error propagation probability, the data were analyzed in two ways: (1) the average error propagation probability indicates the vulnerability of an entire circuit, and (2) the "greater-than-0.5" metric indicates how many individual cells within the design have an EPP greater than 50%. The average EPP is important because it informs mitigation strategies that attempt to harden an entire circuit. For example, the use of a radiation-hardened cell library would improve the average soft error rate of a circuit. The "greater-than-0.5" metric is important for mitigation strategies that implement selective node hardening. For example, selective transistor sizing for reli-

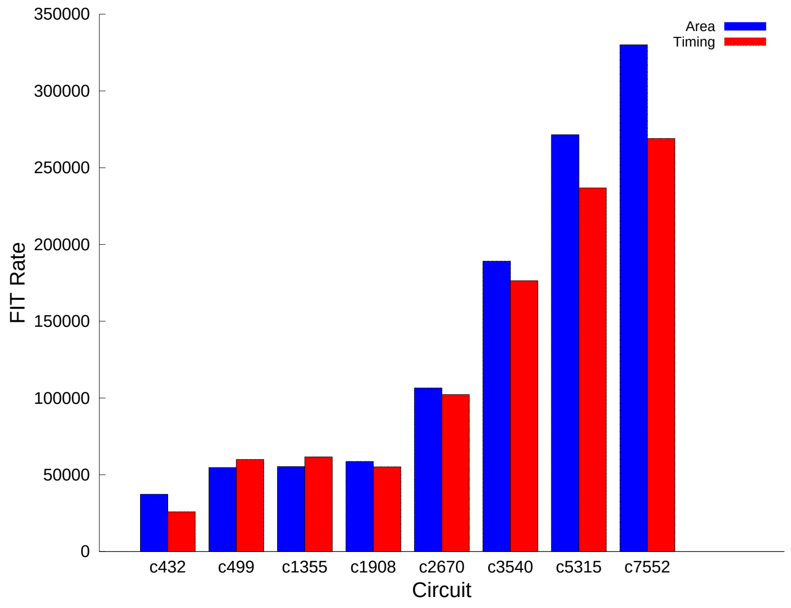


Figure 6.3: FIT rate for selected ISCAS85 benchmark circuits (range: 200-7000 cells), separated by area- and timing-constrained implementations.

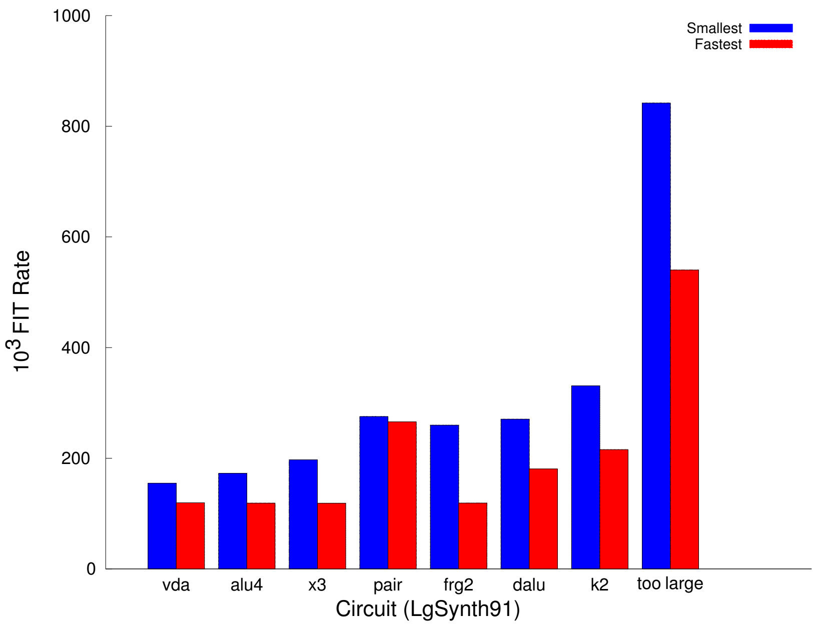


Figure 6.4: FIT rate for selected LgSynth91 benchmark circuits(range: 90-10000 cells), separated by area- and timing-constrained implementations.

ability improvement requires identification of the highly vulnerable nodes to be hardened (Zhou and Mohanram, 2006).

The area- and timing constrained implementations of the ISCAS85 benchmark circuits are compared for error propagation probability in Figures 6.5 and 6.11. The area- and timing constrained implementations of the LgSynth91 benchmark circuits are compared for error propagation probability in Figures 6.6, 6.7, 6.8, 6.9, 6.10, 6.12, 6.13, 6.14, 6.15, and 6.16. In the figures, the LgSynth91 benchmark circuits are arranged by size from most number of cells to least number of cells. The area-constrained implementations have a higher average EPP than the timing-constrained implementations (Figures 6.5, 6.6, 6.7, 6.8, 6.9, and 6.10). This result is consistent with the frequency of XOR and XNOR cells previously discussed. However the number of cells of the timing-constrained implementations with EPP greater than 0.5 are consistently greater than the number of cells of the area-constrained implementations with EPP greater than 0.5. This result, shown in Figures 6.11, 6.12, 6.13, 6.14, 6.15, and 6.16 imply that timing-constrained implementations contain more high-EPP cells than area-constrained implementations, even when the average EPP is lower. This result is probably due to dueling mechanisms: (1) timing optimizations create shorter worst-case paths which increases the probability of an error reaching the output and (2) timing optimizations create longer best-case paths (i.e., the logic depth metric used in this study) in an attempt to balance the worst-case path, which decreases EPP through increased logical masking. Fig. 6.17 supports this theory as there are more nodes with a logic depth of 1 or 2 for the timing-constrained implementation. Additionally, the timing-constrained implementations have nearly twice the frequency of inverters as the area-constrained implementations. Since inverters along a highly-vulnerable path do not logically mask errors, the higher frequency of inverters in timing-constrained implementations could skew both the average EPP and the number of cells with EPP greater than 0.5. The higher frequency of inverters also explains why the timing-constrained implementations have longer logic depths than area-constrained implementations, even though the delay for timing-constrained implementations is smaller.

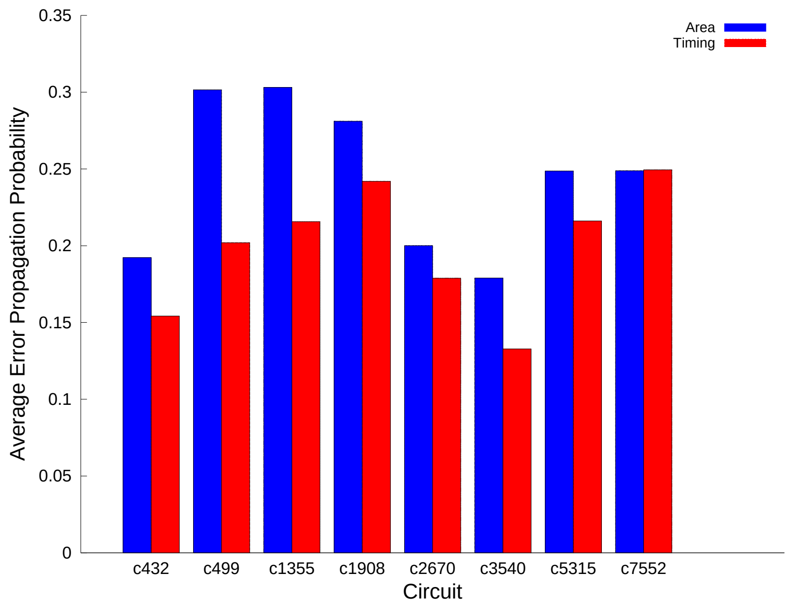


Figure 6.5: Error propagation probability average for the selected ISCAS85 benchmark circuits (range: 200-10000 cells) separated by area-constrained and timing-constrained implementations.

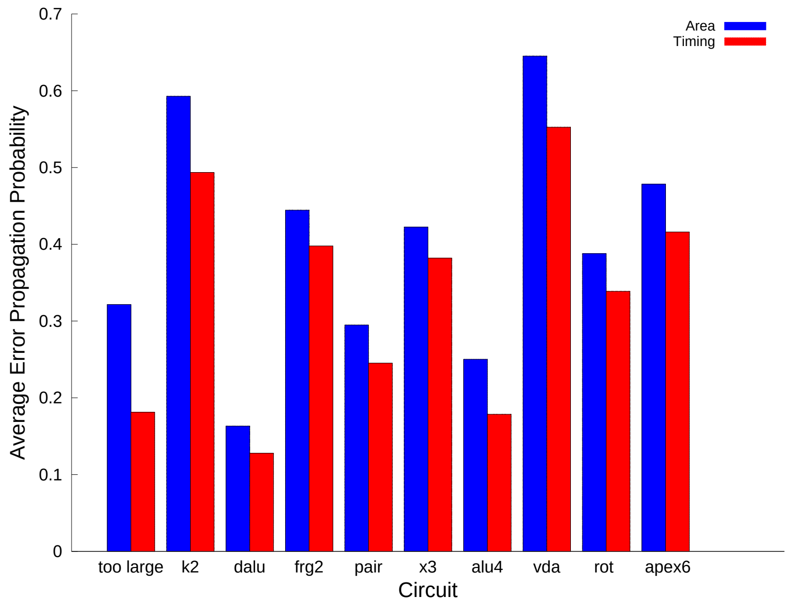


Figure 6.6: Error propagation probability average for the selected LgSynth91 benchmark circuits (range: 8000-700 cells) separated by area-constrained and timing-constrained implementations.

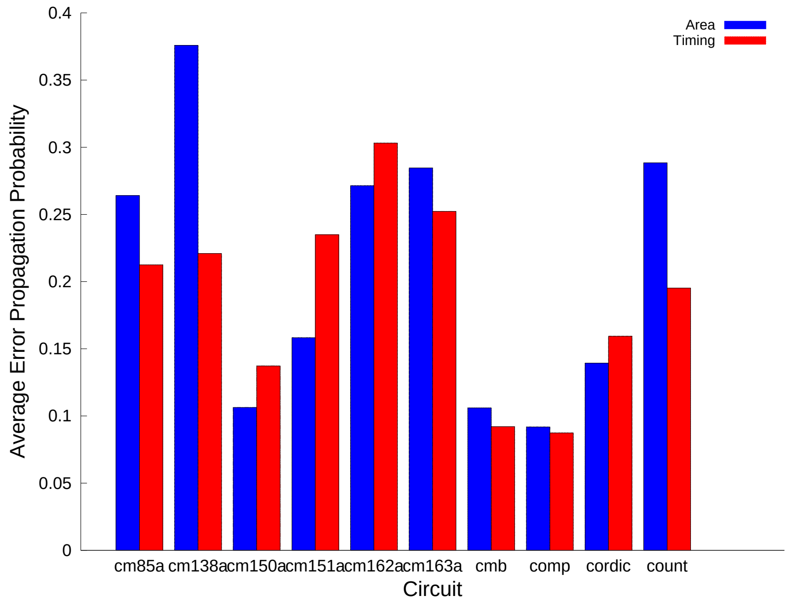


Figure 6.7: Error propagation probability average for the selected LgSynth91 benchmark circuits (range: 80-400 cells) separated by area-constrained and timing-constrained implementations.

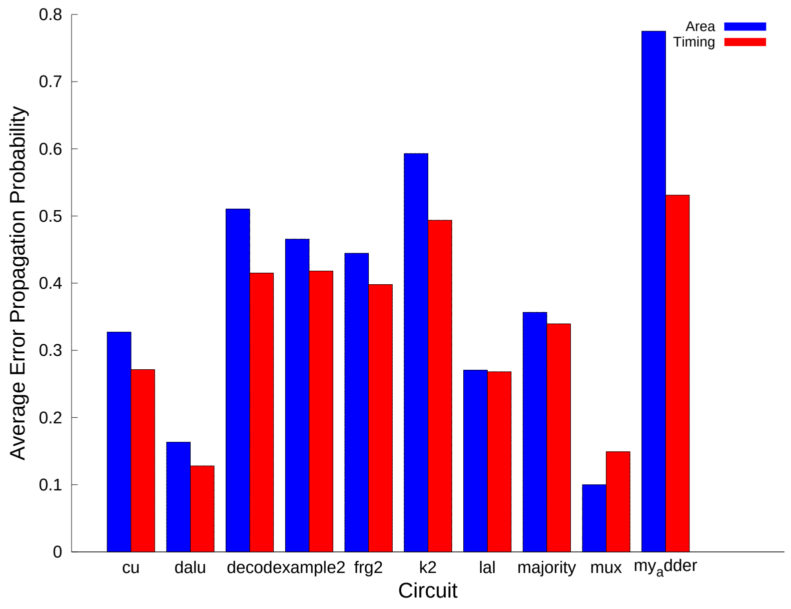


Figure 6.8: Error propagation probability average for the selected LgSynth91 benchmark circuits (range: 80-400 cells) separated by area-constrained and timing-constrained implementations.

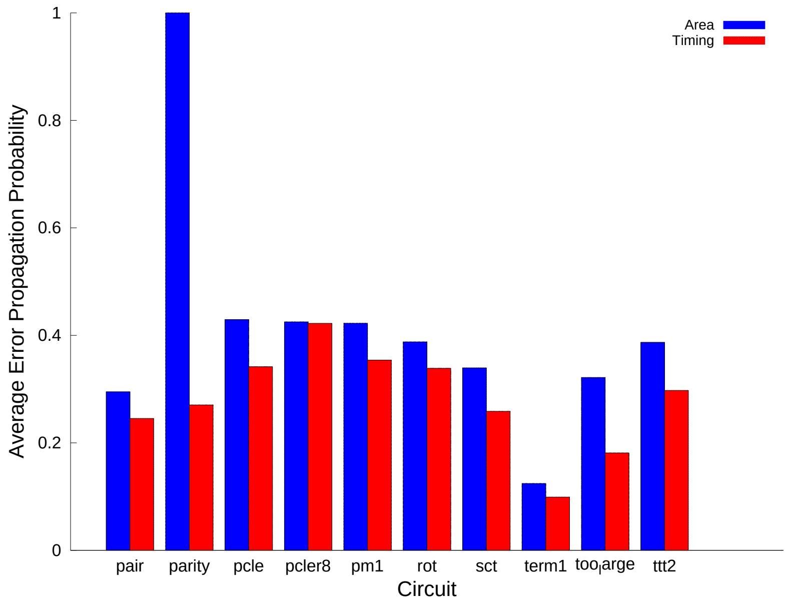


Figure 6.9: Error propagation probability average for the selected LgSynth91 benchmark circuits (range: 300-1700 cells) separated by area-constrained and timing-constrained implementations.

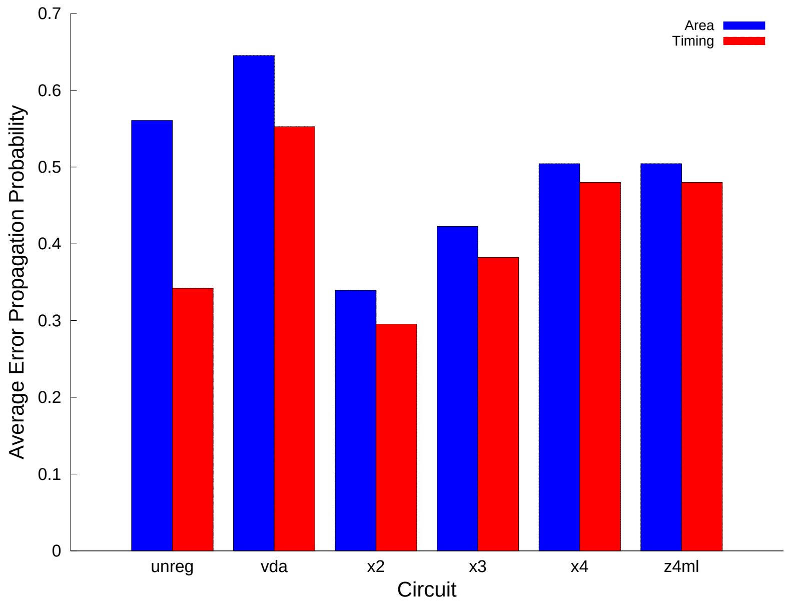


Figure 6.10: Error propagation probability average for the selected LgSynth91 benchmark circuits (range: 150-2000 cells) separated by area-constrained and timing-constrained implementations.

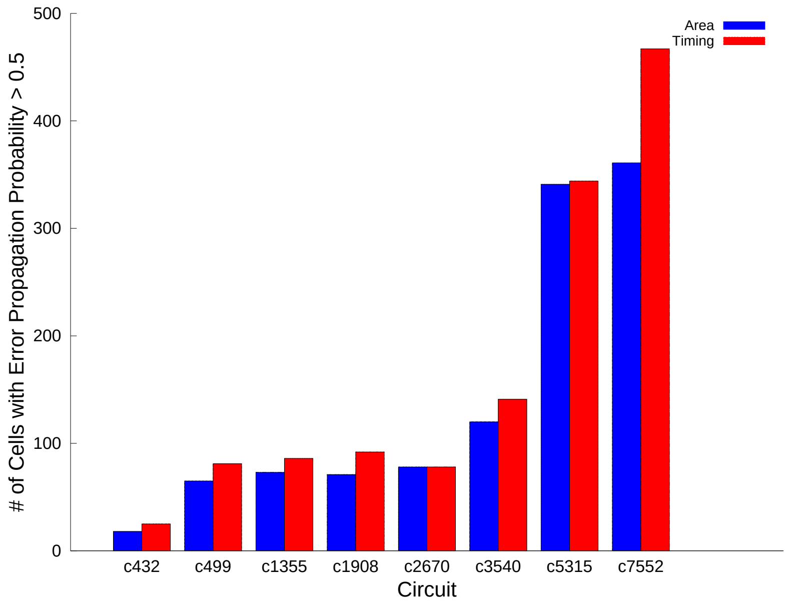


Figure 6.11: Number of cells with EPP greater than 0.5 for the ISCAS85 benchmark circuits (range: 200-10000 cells) separated by area-constrained and timing-constrained implementations.

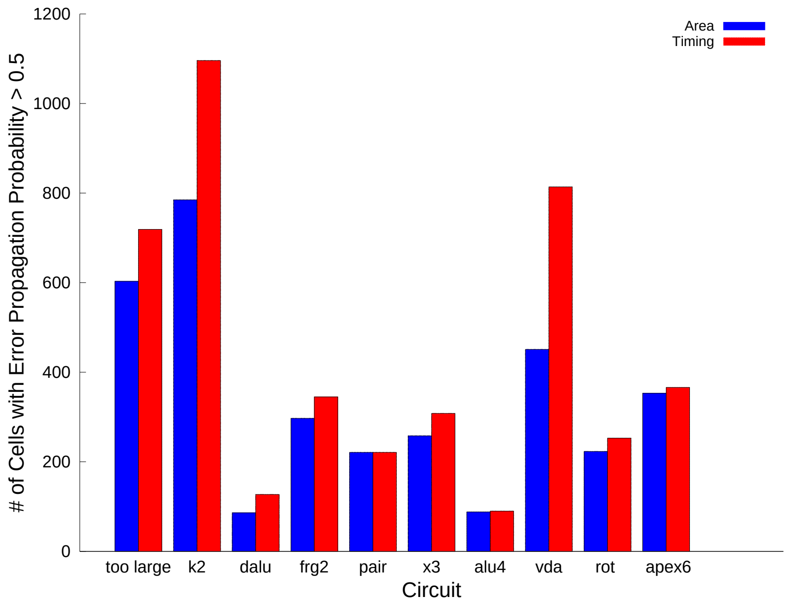


Figure 6.12: Number of cells with EPP greater than 0.5 for the LgSynth91 benchmark circuits (range: 8000-700 cells) separated by area-constrained and timing-constrained implementations.

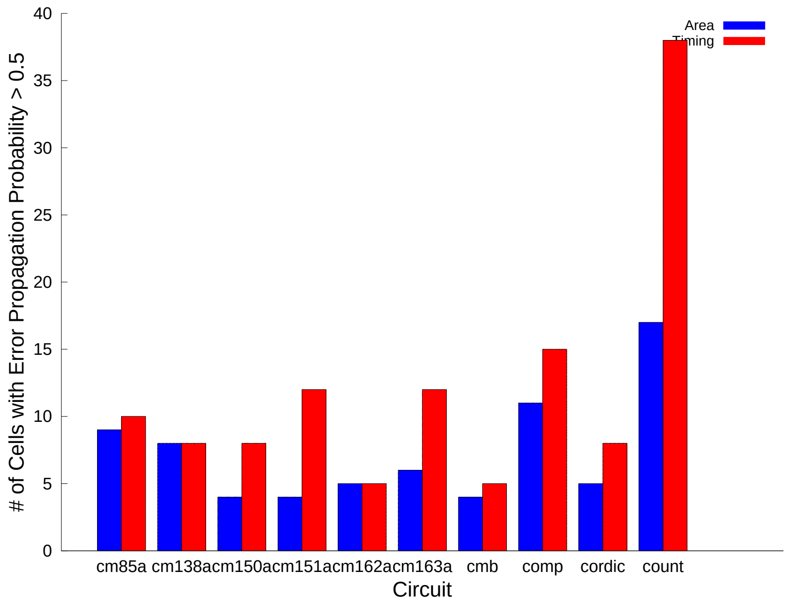


Figure 6.13: Number of cells with EPP greater than 0.5 for the LgSynth91 benchmark circuits (range: 80-400 cells) separated by area-constrained and timing-constrained implementations.

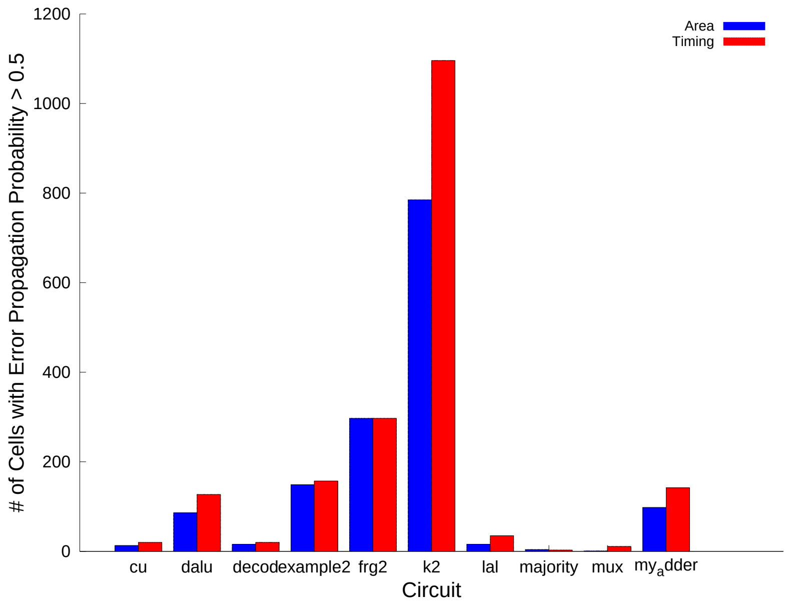


Figure 6.14: Number of cells with EPP greater than 0.5 for the LgSynth91 benchmark circuits (range: 80-400 cells) separated by area-constrained and timing-constrained implementations.

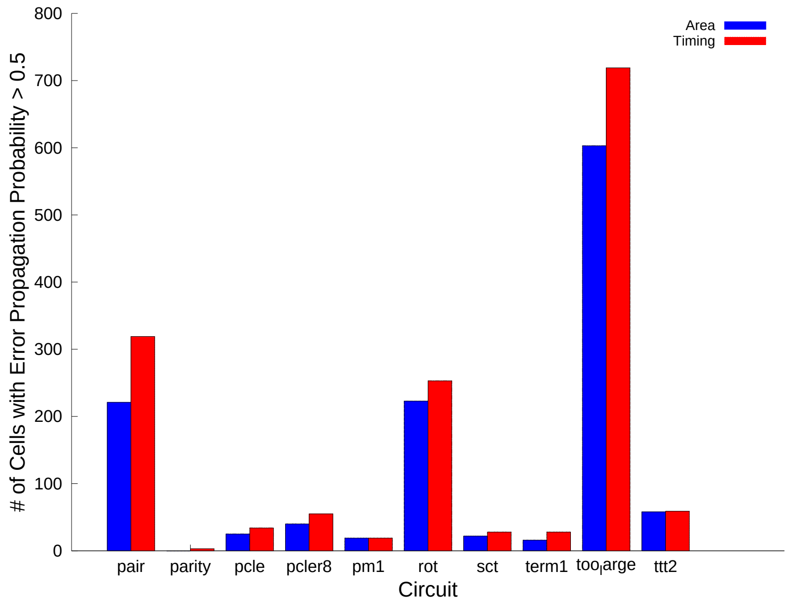


Figure 6.15: Number of cells with EPP greater than 0.5 for the LgSynth91 benchmark circuits (range: 300-1700 cells) separated by area-constrained and timing-constrained implementations.

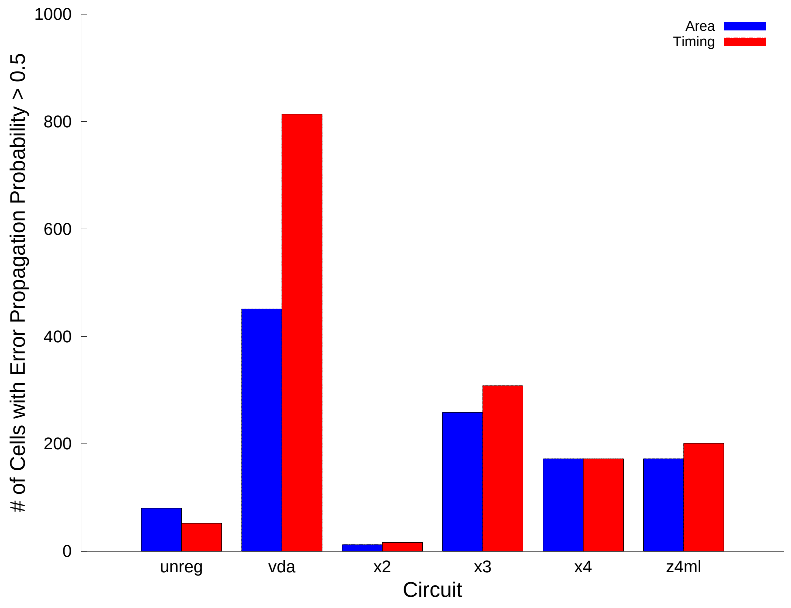


Figure 6.16: Number of cells with EPP greater than 0.5 for the LgSynth91 benchmark circuits (range: 150-2000 cells) separated by area-constrained and timing-constrained implementations.

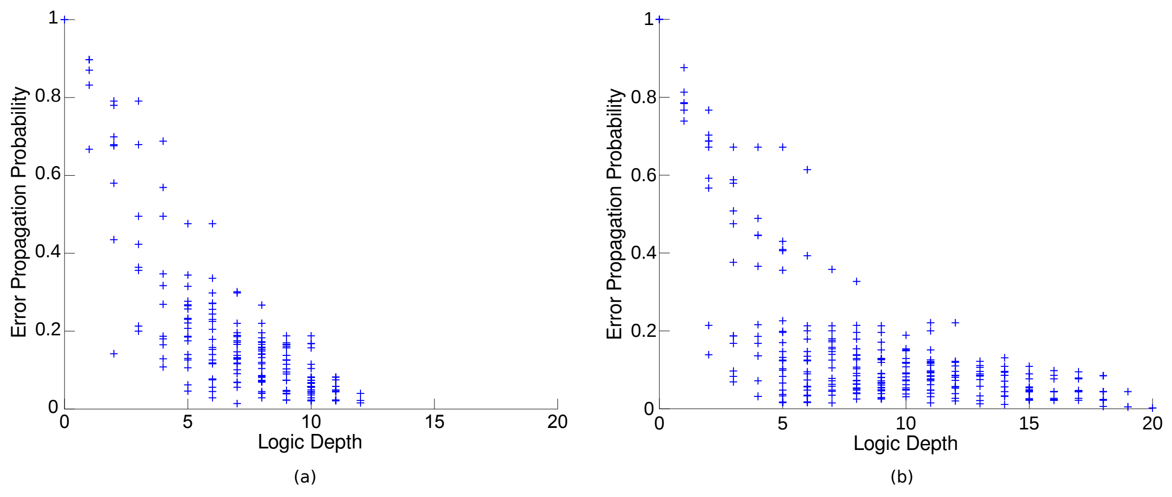


Figure 6.17: Error propagation probability vs. logic depth for each node in the c432 circuit. (a) area-constrained implementation (b) timing constrained implementation.

6.3 Increasing Area and Delay Constraints

This section describes the impact that increasing performance (i.e., area and delay) constraints have on reliability. Each performance constraint is increased iteratively to show trends in reliability. The area constraint cannot be modified in commercial synthesis tools. Therefore the map effort was modified while the delay was unconstrained. Modifying the compiler settings in this way likely provides an indirect way of modifying the area constraints. The delay constraint was modified from minimum to maximum delay.

6.3.1 Reliability Comparisons of Increasing Area by EPP

Figure 6.18 shows the effect of varying map effort during synthesis. For most of the circuits used in this study, the “medium” and “high” map effort produced identical results. The average EPP increases as the map effort increases. The frequency of cell usage does not provide insight because the distribution of cells for the same optimization (in this case, area) does not vary significantly. The increase in EPP could be caused by smaller implementations of a circuit having more vulnerability due to greater fanouts of each gate. However, Figures 6.19 and 6.20 contradict this assertion by showing no correlation between fanout and EPP. In situations where the fanout nodes continue along the same path, reconvergence may actually decrease the likelihood that an error will propagate to an output node.

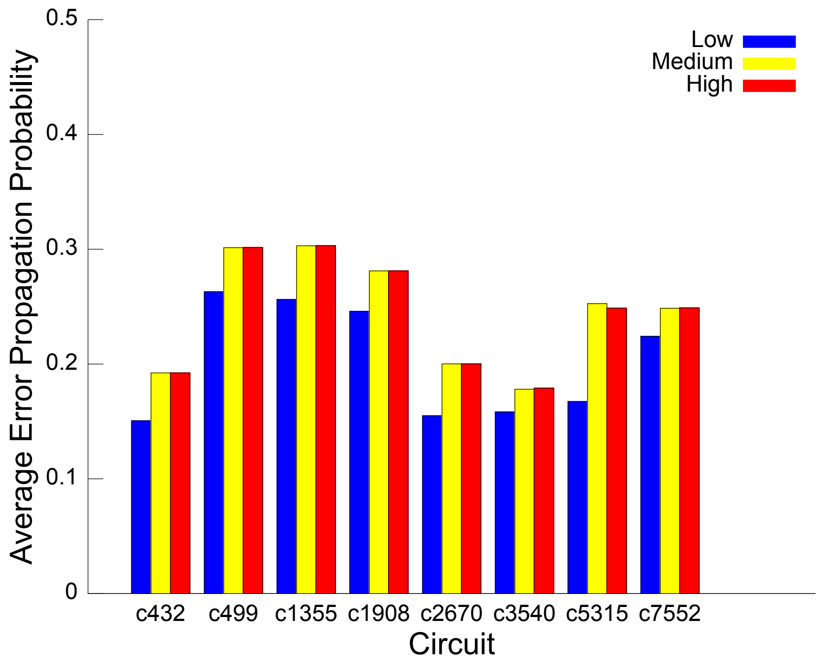


Figure 6.18: EPP for the selected ISCAS85 benchmark circuits separated by map effort.

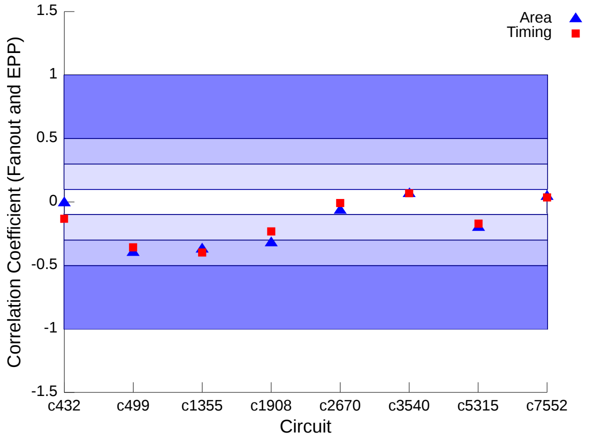


Figure 6.19: Correlation coefficient between fanout and EPP. Ranges for the correlation coefficient, r, are separated by shaded regions: $|r| > 0.5$ represents a strong correlation, 0.3 $< |r| < 0.5$ represents a medium correlation, $0.1 < |r| < 0.3$ represents a weak correlation, $|r| <$ 0.1 represents no correlation. A negative r implies an inverse relationship.

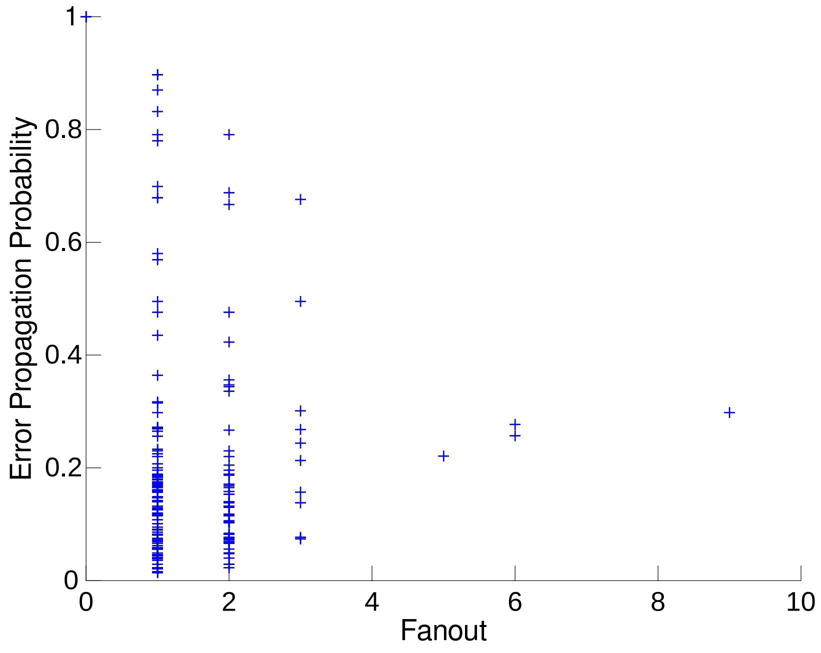


Figure 6.20: Error propagation probability vs. node fanout for the c432 circuit

6.3.2 Reliability Comparisons of Increasing Delay by EPP

Table 6.1 shows the results from varying the delay-optimization incrementally from minimum constraint (Timing 1) to maximum constraint (Timing 5). The step size was determined by dividing the slack between the minimum and maximum constraint by 4. The results show that a decrease in delay does not always correspond with an increase in average EPP. It is also worth noting that the number of cells with an EPP greater than 0.5 increases steadily with decreasing delay, meaning that even if the average does not increase, the number of highly vulnerable cells is increasing. The trend in the number of cells with an EPP greater than 0.5 differing from the average EPP trend is consistent with the analysis of area-constrained implementations versus timing-constrained implementations. This trend was expected since a design with minimum delay will have shorter paths and therefore fewer gates for an error to propagate to reach an output. Figures 6.21 and 6.22 show the reliability of the ISCAS85 benchmark circuits for minimum delay and maximum delay implementations. The trends for the average EPP and the number of cells with an EPP greater than 0.5 are the same as those mentioned for the c432 circuit.

Table 6.1: EPP of c432 for varying circuit delay optimizations. The symbol "Δ" represents delay step size and is calculated as the maximum delay minus the minimum delay divided by 4.

Constraint	**Delay (ns)**	**Cell Count**	**Avg EPP**	**# EPPs > 0.5**
Min Delay	833	268	0.15	25
Min Delay + Δ	1773	196	0.14	21
Min Delay + 2Δ	2713	196	0.14	21
Min Delay + 3Δ	3653	181	0.15	18
Max Delay	4593	172	0.20	19

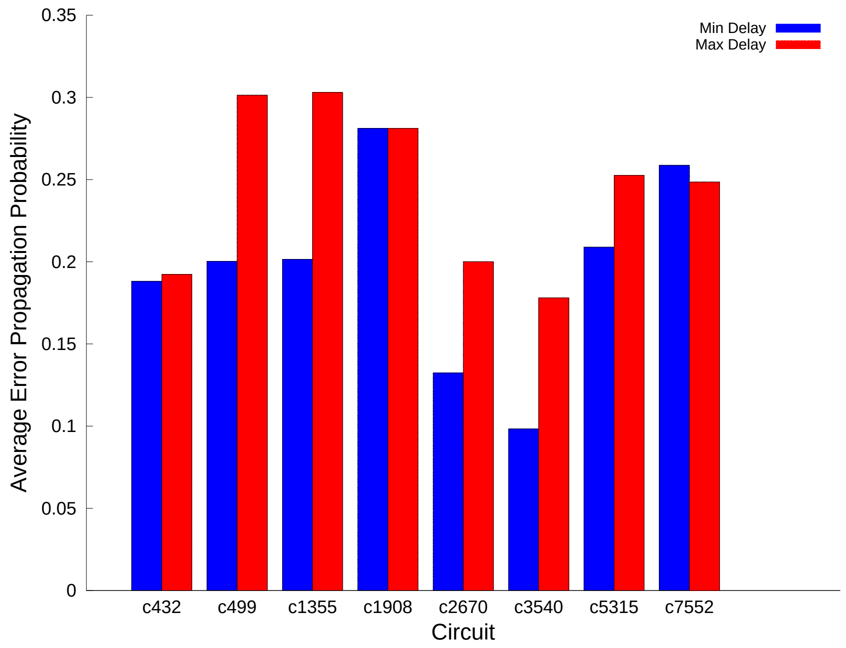


Figure 6.21: Average EPP for the ISCAS85 benchmark circuits separated by mininum delay and maximum delay implementations.

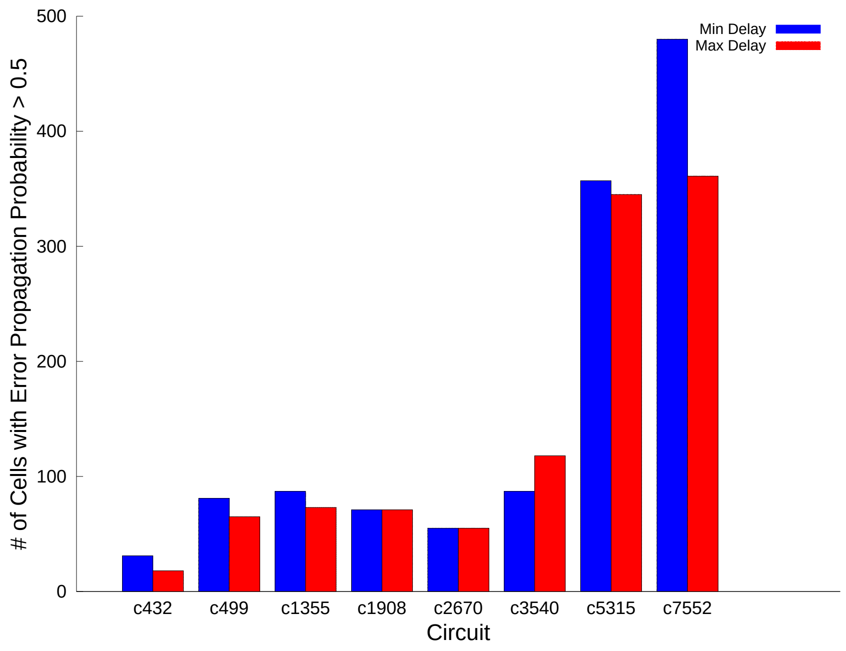


Figure 6.22: Number of cells with EPP greater than 0.5 for the ISCAS85 benchmark circuits separated by mininum delay and maximum delay implementations.

6.4 Reliability Comparisons of Cell Availability by EPP

When restricting the cell selection, there was not an identifiable trend in average EPP, however, cell-restricted circuits more often showed an increase in the EPP compared to the original implementation (Figure 6.23). The absence of an identifiable trend could be due to the removal of one cell causing an increase in use of the cell's complement. For instance, removal of the XOR gate caused greater than 30% average increase in XNOR gates used. Since XOR and XNOR have a similar logical vulnerability, the removal of one of these cells should have a marginal impact on EPP.

6.5 Summary

The results mentioned in the chapter provides evidence that the influence of constraints on topology correspond to a relationship between topology and reliability. Timing-constrained implementations contain more highly vulnerable cells than area-constrained implementations. The results for removing a cell from the library indicate that the EPP may increase when removing cells from the library but currently do not reflect a definite trend. Finally,

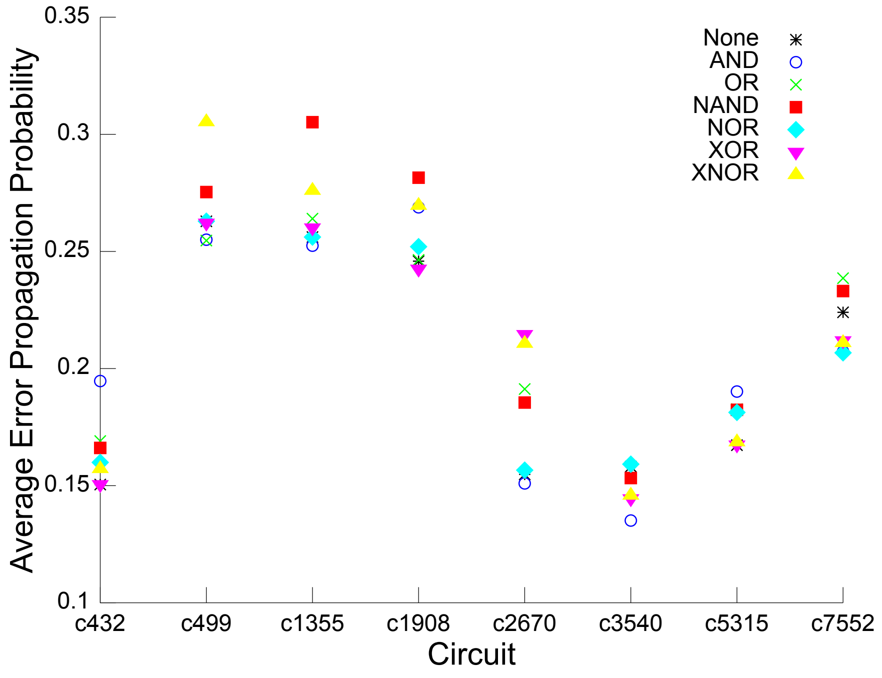


Figure 6.23: EPP for the selected ISCAS85 benchmark circuits separated by implementations that have one cell restricted during synthesis. There is not a cell that consistently increases or decreases the EPP of a circuit. Additionally, the impact of removing cells is negligible in some circuits (e.g. c5315) and noticeable in other circuits (e.g. c2670).

the results for varying a timing-constrained implementation from minimum constraint to maximum constraint show that EPP increases with tighter delay constraints, which could be attributed to the shortened delay paths.

The effectiveness of reliability-aware synthesis techniques are presented in literature without knowledge of the constraints used to synthesize the test circuits used in the work. The results in this chapter indicate that topology impacts reliability, therefore, analysis of proposed techniques across constraints needs to be considered.

Chapter 7

Reliability-Aware Synthesis

The first step in mitigating soft errors is identifying vulnerable nodes and vulnerable cells. This step was addressed in Chapter 5. Once the vulnerable nodes of a circuit and the vulnerable cells in a cell library were obtained, reliability trends relating to synthesis constraints were identified in Chapter 6. In this chapter, the key results are presented for mitigating the impact of soft errors using techniques during synthesis. The majority of existing techniques modify optimization steps, traditionally used to reduce area, power, or delay of a circuit, in order to reduce circuit SER. A survey of common techniques are discussed in this chapter. Additionally, there are opportunities to exploit the design space created by certain logic synthesis steps. Reliability-aware synthesis focuses on reducing the baseline soft error rate for a synthesized circuit and does not preclude the use of other mitigation techniques. Tools for electronic design automation (EDA) can be extended with the capability to account for the potential soft error rate of individual library cells or cascaded chains of cells. A hardware design compiler could also analyze highly sensitive paths and implement them with additional protection against radiation-induced soft errors (e.g., clock signals, control lines). These opportunities are also discussed in this chapter.

The results presented in this chapter illustrate the improvement in reliability that is obtained through selectively hardening vulnerable nodes by replacing them with unaltered cells in the cell library. This technique allows for reliability to be improved at the logic synthesis design stage, i.e., before the use of mitigation techniques that physically alter the cell library. As the results presented in this chapter indicate, the reliability can be improved by as much as 20% with less than 1% power overhead.

7.1 Post-synthesis Mitigation Techniques

In addition to soft error mitigation techniques that are implemented during the synthesis process, techniques implemented after the traditional synthesis steps have been employed.

A subset of these techniques is included in this section.

7.1.1 Rewiring

Rewiring is an optimization technique that involves removing a wire from a circuit design that violates a constraint and transforming the remaining logic to be functionally equivalent to the original circuit. Almukhaizim et al. employed an automatic-test-pattern-generation-based rewiring method to generate functionally-equivalent yet structurally-different implementations of a logic circuit based on simple transformation rules (Almukhaizim et al., 2006). The results showed that the SER of a circuit could be reduced through topological transformations without impacting area, power, or timing constraints. This method only dealt with the logic level and does not take advantage of technology-specific design characteristics.

7.1.2 Resynthesis/Rewriting

Krishnaswamy et al. (Krishnaswamy et al., 2007) improved circuit SER by using two circuit modification techniques: don't-care-based resynthesis and local rewriting. *Observability-don't-cares (ODCs)* are logical input values that do not impact the output node. For instance, when a two-input AND gate has one input value set to zero, the second input has no impact on the output and is considered an ODC. The study uses the ODC information to characterize the vulnerability of nodes, then uses pre-existing redundant logic (as identified by the ODCs) to harden highly vulnerable nodes. *Logic rewriting* is a synthesis optimization technique that involves replacing subcircuits in a design with smaller subcircuits that are functionally equivalent (Mishchenko and Chatterjee, 2006). Krishnaswamy et al. modified this technique to replace subcircuits that optimize for area and reliability simultaneously. Both techniques were limited to the solution that produced the most logical masking and did not account for electrical or latch-window masking.

The impact of gate-sizing through cell selection was studied by Rao et al. (Rao et al., 2009). As shown in Figure 7.1, cells with higher drive strengths increase attenuation of SET pulses. By optimizing combinational logic for reliability through selective gate-sizing, Rao et al. show that the overall SER of a circuit can be reduced up to 30.1x while accruing an area overhead of 46% and no delay penalties. This work showed that SER can be reduced by considering pre-existing cells in a library to accomplish gate-sizing. However, the study provided an algorithm that optimized reliability following other optimizations and therefore did not consider the impact that previous optimizations (e.g., area and delay) had on reliability.

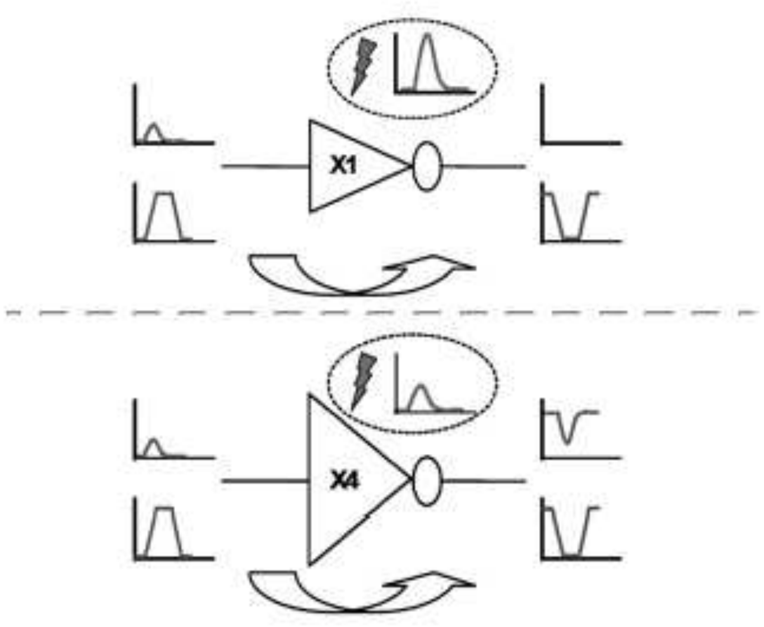


Figure 7.1: Qualitative comparison (in terms of electrical properties) between INVX1 and INVX4. An increase in drive strength results in an increase in pulse attenuation. Top (Circled) = Injected waveforms and Sides = Propagated waveforms. (Rao et al., 2009).

A technology mapping algorithm that optimizes for fault sensitivity was introduced by Wo et al. (Wo and Koren, 2005). In that study, pre-existing technology mapping algorithms were rewritten to create a synthesis engine that was reliability-aware. While helpful for the design of logic synthesis tools, this method cannot be used in situations where a circuit designer relies on commercial synthesis engines. The minimum fault sensitivity optimization was compared to minimum area and delay optimizations. However, the study does not consider optimizing a design for reliability while still leveraging the impact of other metrics.

7.2 Covering-based Reliability-Aware Synthesis

Reliability-aware logic synthesis can be used to mitigate a circuit's response to radiation-induced soft errors. This chapter analyzes the impact of using covering-based reliability-aware logic synthesis to reduce both the pulse width and the drain area of a circuit. Using our targeted cell library, several benchmark circuits were analyzed to identify equivalent, less-vulnerable implementations to cover complex cell patterns (i.e., exclusive-or, and-or-invert) while minimizing penalties. Results showed that replacing cells with alternative implementations can lower a circuit's typical pulse width by greater than 30% and typical drain area by greater than 40%. The respective area penalties incurred are less than 115% and 65%.

The combinational benchmark circuits for this study were chosen from the ISCAS85 suite and synthesized using a digital cell library for 90-nm bulk CMOS technology (Atkinson et al., 2011). The library contains 1034 cells that implement basic (e.g., NAND, NOR) and complex (e.g., AND-OR-INV, XOR) logic. The library includes variants of cells with different drive strengths. However, for this study, the focus was on cells with the same drive strength, namely 1x. Circuits were synthesized to create netlists using the Synopsys Design Compiler.

The sensitivity of a circuit to a single event depends on the transient-generation probability. The actual transient shape will be defined by: (1) the particle type and energy, (2) the transistor size of the driver, (3) the load capacitance, and (4) the layout strategy.

Table 7.1: Pulse width and drain area results from (Atkinson et al., 2011)

Cell	**Area (μm^2)**	**Drain Area (μm^2)**	**Pulse Width (ps)**
INV	2.82	0.24	240
NAND2	4.70	0.16	320
AOI21	5.64	0.56	340
AND2	5.64	0.30	240
OAI21	6.59	0.42	360
NOR2	7.53	1.18	320
OR2	9.41	1.04	300
XNOR2	10.35	0.27	450
XOR2	11.29	0.34	370

Table 7.2: Pulse width decrease, drain area decrease, area increase, and delay increase for complex logic cell implementations

OAI21

Implementation	Pulse Width Decrease	Drain Area Decrease	Area Increase	Delay Increase
OR → NAND	11.11%	-147.84%	114.29%	100.00%
4 x NAND	11.11%	**62.81%**	185.71%	66.67%
2 x NOR → NAND	11.11%	-178.84%	200.00%	133.33%
OR → AND → INV	16.67%	-147.84%	171.43%	233.33%
2 x INV → 2 x NAND	11.11%	44.21%	128.57%	66.67%
4 x INV → 2 x AND	**33.33%**	29.41%	242.86%	333.33%

Table 7.3: Pulse width decrease, drain area decrease, area increase, and delay increase for complex logic cell implementations

AOI21

Implementation	Pulse Width Decrease	Drain Area Decrease	Area Increase	Delay Increase
NAND → INV → NOR	5.88%	-110.68%	166.67%	50.00%
2 INV → NOR → NOR	5.88%	-110.68%	166.67%	75.00%
AND → OR → INV	11.76%	-87.26%	166.67%	150.00%
2 AND → 2 INV	**29.41%**	46.67%	166.67%	125.00%
1 NAND → 1 INV → 1 AND	5.88%	46.67%	166.67%	75.00%
2 NAND → 2 INV	5.88%	**57.85%**	166.67%	25.00%

Table 7.4: Pulse width decrease, drain area decrease, area increase, and delay increase for complex logic cell implementations

XOR2

Implementation	Pulse Width Decrease	Drain Area Decrease	Area Increase	Delay Increase
4 x NAND	13.51%	**54.42%**	66.67%	-44.44%
4 x NOR → INV	13.51%	-241.74%	191.67%	11.11%
XNOR → INV	-21.62%	21.86%	16.67%	11.11%
2 x INV → 2 x AND → OR	18.92%	-203.75%	133.33%	11.11%
4 x INV → 3 x AND	**35.14%**	13.49%	150.00%	37.50%

Table 7.5: Pulse width decrease, drain area decrease, area increase, and delay increase for complex logic cell implementations

XNOR2

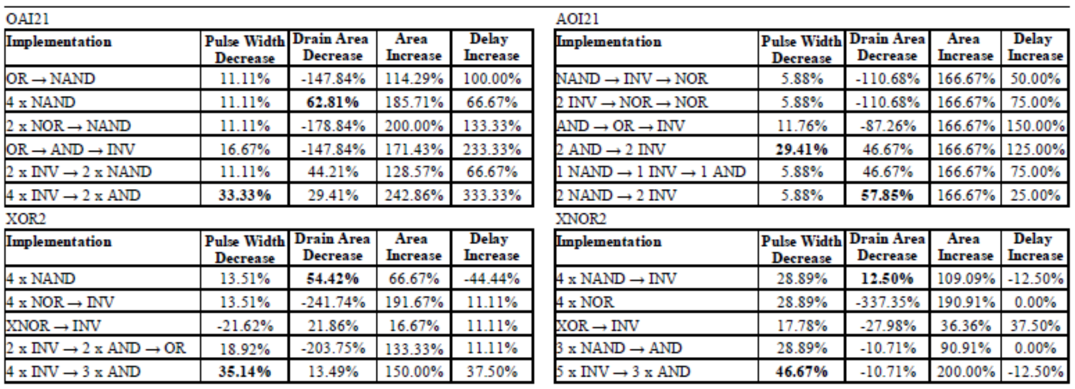
Implementation	Pulse Width Decrease	Drain Area Decrease	Area Increase	Delay Increase
4 x NAND → INV	28.89%	**12.50%**	109.09%	-12.50%
4 x NOR	28.89%	-337.35%	190.91%	0.00%
XOR → INV	17.78%	-27.98%	36.36%	37.50%
3 x NAND → AND	28.89%	-10.71%	90.91%	0.00%
5 x INV → 3 x AND	**46.67%**	-10.71%	200.00%	-12.50%

The pulse width and drain area values for each cell used in this study were calculated using TCAD and Accuro simulations and presented in (Atkinson et al., 2011). In (Atkinson et al., 2011), the drain area of each cell was identified to be most sensitive to ion strikes. For this study, the pulse width and drain area are used as the reliability metrics to rank cells. A subset of the results from the study is included in Table 7.1. The cell-level netlists were examined for the frequency of use of each cell individually as well as the frequency of cell subcircuits (i.e., a cell and the cascaded fan-in that supports the cell). Two-cell-depth subcircuits were used for this study that were composed of: (1) an individual cell and (2) all cells connected to the input nodes of (1). For individual cells, the AOI21, OAI21, XOR2, and XNOR2 cells were identified as having a relatively high pulse width and frequency of use.

7.2.1 Replacing Cells for Reduced Pulse Widths and Drain Areas

The usage of library cells and their individual contributions to the single event transient (SET) pulsed width and drain area were examined for multiple circuits. To compare the individual contributions among implementations, the worst-case value of a metric was taken as the reliability metric of the most vulnerable cell for that implementation. For instance, with an implementation that contains one OR2 cell and one NAND2 cell, the highest pulse width is 320 ps. This result is compared to the pulse width for the OAI21 cell of 360 ps. When comparing the contributions of the implementations to a benchmark circuit, the "typical" pulse width and "typical" drain area of the entire circuit were taken to be the pulse width per cell and drain area per cell respectively. Although the pulse width values are tied to a specific cell library, this methodology could be applied to any cell library once pulse width values (or any other reliability metric) are obtained. The strategy used to reduce the pulse width and drain area was to: (a) synthesize circuits to a given cell library, (b) identify frequently used and highly vulnerable cells, and (c) remove those cells from the library for re-synthesis of the circuit. The *efficiency* of the mitigation strategy was defined by taking the ratio of the decrease in vulnerability to the increase in cell area, as in,

$$\text{pulse width improvement efficiency} = \frac{\text{pulse width decrease}}{\text{area increase}} \tag{7.1}$$



OAI21

Implementation	Pulse Width Decrease	Drain Area Decrease	Area Increase	Delay Increase
OR → NAND	11.11%	-147.84%	114.29%	100.00%
4 x NAND	11.11%	**62.81%**	185.71%	66.67%
2 x NOR → NAND	11.11%	-178.84%	200.00%	133.33%
OR → AND → INV	16.67%	-147.84%	171.43%	233.33%
2 x INV → 2 x NAND	11.11%	44.21%	128.57%	66.67%
4 x INV → 2 x AND	**33.33%**	29.41%	242.86%	333.33%

AOI21

Implementation	Pulse Width Decrease	Drain Area Decrease	Area Increase	Delay Increase
NAND → INV → NOR	5.88%	-110.68%	166.67%	50.00%
2 INV → NOR → NOR	5.88%	-110.68%	166.67%	75.00%
AND → OR → INV	11.76%	-87.26%	166.67%	150.00%
2 AND → 2 INV	**29.41%**	46.67%	166.67%	125.00%
1 NAND → 1 INV → 1 AND	5.88%	46.67%	166.67%	75.00%
2 NAND → 2 INV	5.88%	**57.85%**	166.67%	25.00%

XOR2

Implementation	Pulse Width Decrease	Drain Area Decrease	Area Increase	Delay Increase
4 x NAND	13.51%	**54.42%**	66.67%	-44.44%
4 x NOR → INV	13.51%	-241.74%	191.67%	11.11%
XNOR → INV	-21.62%	21.86%	16.67%	11.11%
2 x INV → 2 x AND → OR	18.92%	-203.75%	133.33%	11.11%
4 x INV → 3 x AND	**35.14%**	13.49%	150.00%	37.50%

XNOR2

Implementation	Pulse Width Decrease	Drain Area Decrease	Area Increase	Delay Increase
4 x NAND → INV	28.89%	**12.50%**	109.09%	-12.50%
4 x NOR	28.89%	-337.35%	190.91%	0.00%
XOR → INV	17.78%	-27.98%	36.36%	37.50%
3 x NAND → AND	28.89%	-10.71%	90.91%	0.00%
5 x INV → 3 x AND	**46.67%**	-10.71%	200.00%	-12.50%

Figure 7.2: Pulse width decrease, drain area decrease, area increase, and delay increase for complex logic cell implementations.

$$\text{drain area improvement efficiency} = \frac{\text{drain area decrease}}{\text{area increase}} \tag{7.2}$$

The results for the pulse width, drain area, cell area, and timing delay values can be seen in Table 7.2. Figure 7.3 shows the impact that removing the AOI21, OAI21, XOR2, and XNOR2 cells can have on the both pulse width and drain area with respect to the area of a complex cell. It can be observed that the best implementation of each complex standard cell is a combination of the highest ranked cells in terms of pulse width and drain area. For instance, in terms of pulse width, the implementations of OAI21 with the highest pulse width improvement efficiency are composed of AND2 and INV cells. This result is consistent with pulse quenching studies such as (Albhin et al., 2009), where the effect of the transient is minimized by charge sharing of closely spaced nodes. For the XOR2 and XNOR2 cells, the pulse width efficiency is higher (Figure 7.3) because the original XOR2 and XNOR2 cells had pulse widths that were higher than any other cells.

Additionally, substituting for the best implementation of OAI21, AOI21, and XNOR21 as seen in Figure 7.4, causes a typical pulse width efficiency of between 20-40% across all circuits in the study. In terms of drain area, the drain area efficiency was greater than 50% across all circuits in the study and in some cases exceeds 100%. This analysis accounts for the frequency of use of the cells within the overall design. If timing delay were the metric of concern, then it is worth noting that certain implementations have either a delay decrease or no change in delay.

Each complex cell implementation will have internal logical masking that provides ad-

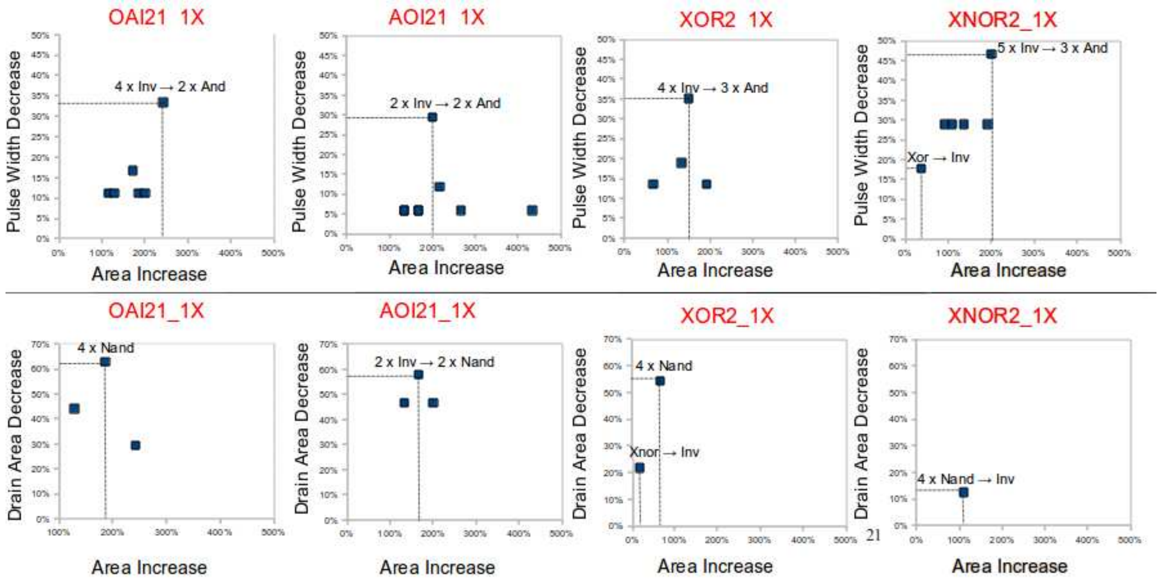


Figure 7.3: Pulse width and drain area calculations for alternative implementations of individual cells. The percentages of pulse width decrease and area increase are defined with respect to the pulse width and area values of the original cell. The most efficient implementations are labeled.

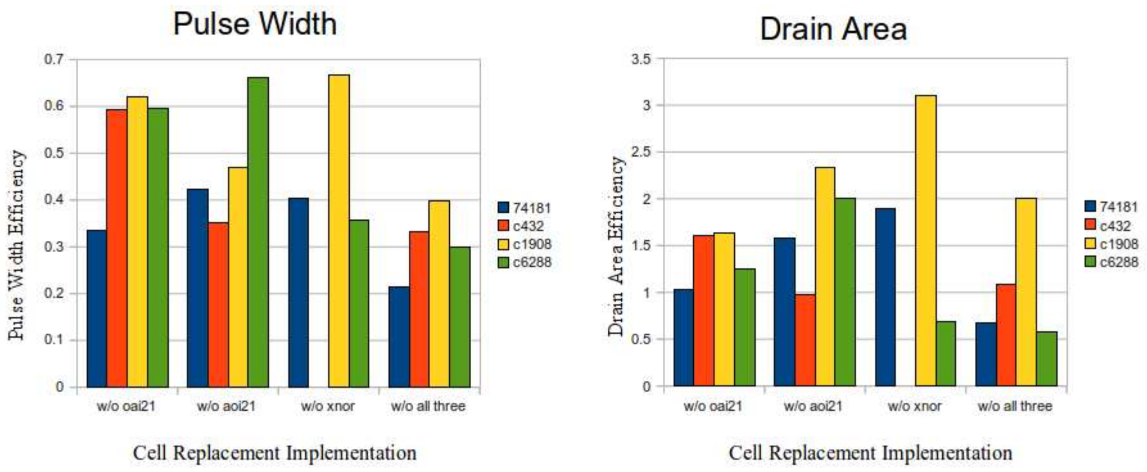


Figure 7.4: Pulse width decrease and drain area decrease per area penalty ratio for multiple benchmark circuits.

ditional reliability. The EPP of the implementations with the smallest drain area and pulse width can be seen in Table 7.6.

Table 7.6: EPP of implementations with best pulse width and drain area

Cell	**Best Pulse Width (ps)**	**EPP**	**Best Drain Area (μm^2)**	**EPP**
AOI21	240	0.50	0.24	0.58
OAI21	240	0.47	0.16	0.38
XOR2	240	0.75	0.24	0.85
XNOR2	240	0.78	0.27	0.88

7.2.2 Replacing the XOR Pattern Tree for reduced EPP

The overall SER of a circuit is dependent upon the logical masking characteristics. The error propagation probability (EPP), the likelihood that an error in one component propagates to output of the circuit, as an estimation for the impact of this characteristic. EPP was calculated with Equation 7.3:

$$\text{EPP} = \frac{\Sigma \text{ input combinations resulting in error}}{\Sigma \text{ all possible input combinations}} \tag{7.3}$$

The impact of implementation choices on the EPP of a circuit was studied by investigating implementations of the exclusive-or (XOR) function. A direct comparison of EPP for all implementations cannot be made at the cell level because complex cells are implemented with transistor-level logic. However, an analysis of universal gate implementations is relevant in libraries that implement an XOR cell as a combination of 2-input NAND cells. From Figure 7.3, it can be observed that the implementation with the largest decrease in drain area was composed of all NAND cells. Using this result to guide our analysis, the EPP of that implementation was calculated (Figure 7.5). This implementation has an EPP of 0.85, further reducing the number of errors seen at the output.

This analysis was extended to include multiple implementations of the XOR logic function (see Figure 7.6). Each implementation has different internal EPPs and performance impact. Calculating EPP through exhaustive fault injection simulation showed that the NAND5 implementation of the XOR function typically has the highest EPP-to-Performance

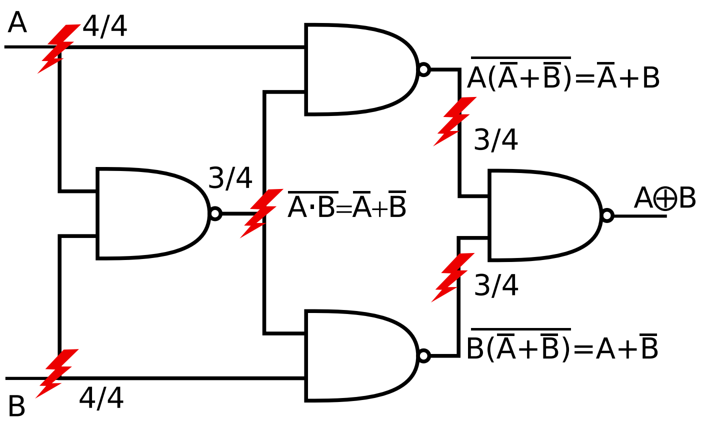

A
4/4
$\overline{A(\overline{A}+\overline{B})}=\overline{A}+B$
3/4
3/4
$\overline{A\cdot B}=\overline{A}+\overline{B}$
$A\oplus B$
3/4
$\overline{B(\overline{A}+\overline{B})}=A+\overline{B}$
B
4/4

Figure 7.5: Error locations for XOR logic function constructed from 4 NAND gates

ratio for area, power, and speed (Figure 7.7).

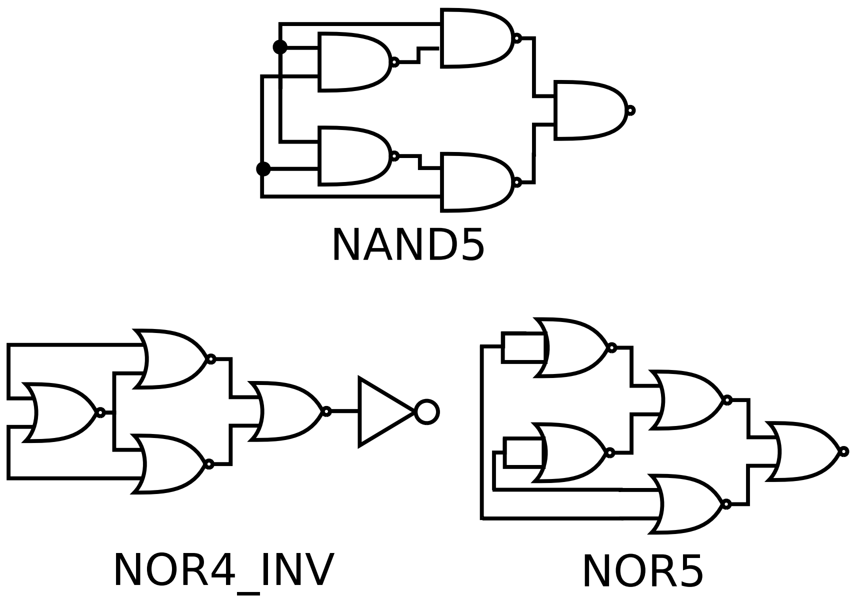
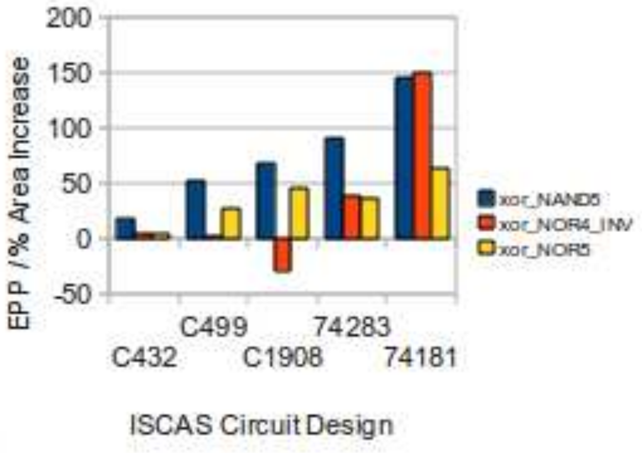
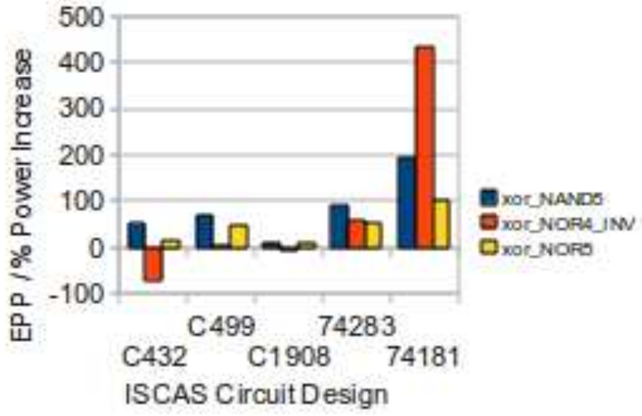


Figure 7.6: Three implementations of the XOR logic function.

The results from this study can be directly inserted into the logic synthesis flow. Recall from Chapter 3 that a list of pattern trees are created for each logic function. The NAND5 implementation of the XOR function has improved reliability compared to the NAND4 implementation with small performance penalty. Therefore the pattern tree for the NAND4 implementation can be replaced with the pattern tree of the NAND5 implementation (Figure 7.8) during synthesis.





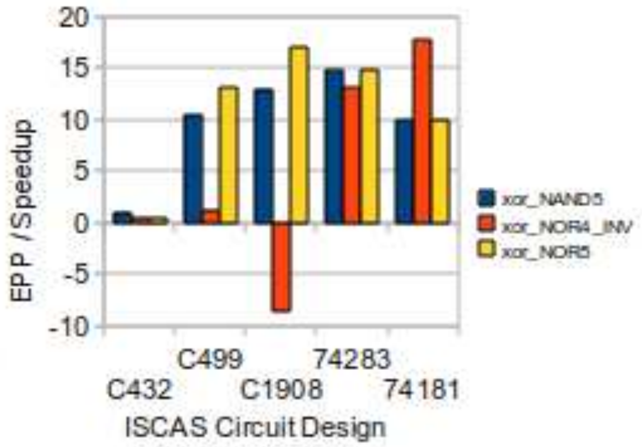


Figure 7.7: Ratio of the decrease in EPP over the increase in performance penalty for each circuit implementation normalized by the NAND4 implementation.

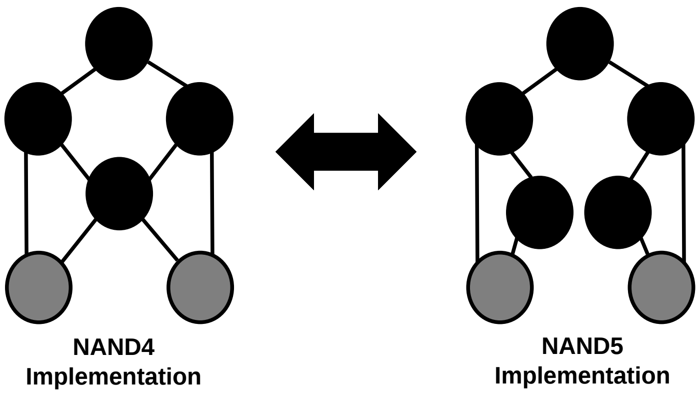


Figure 7.8: Pattern trees for the NAND4 and NAND5 implementations of the XOR logic function (INV = white, NAND2 = black, Input = gray).

7.2.3 Replacing Common Subcircuits for Reduced EPP, Drain Area, and Pulse Width

The common subcircuits generated by the design compiler were also examined to determine whether a different implementation of the logic function could be built with less vulnerable logic cells. The subcircuits were identified by parsing the cell-level netlists for cells that were directly connected to each other. Of the 36 unique subcircuits contained within the 74181 design, two subcircuits were identified that occurred four times, which represented the most frequent cases. The logical functions for those two common subcircuits were extracted and then reimplemented as circuits with cells that cumulatively have a greater reliability. These circuit transformations were performed on the common subcircuits of the 74181 design (Figures 7.9 and 7.10).

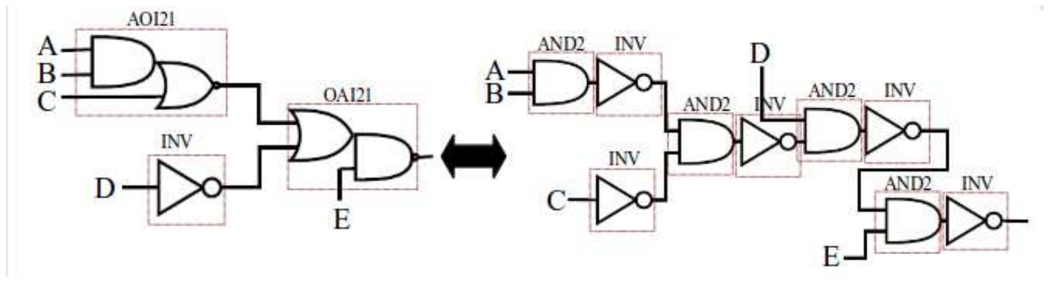


Figure 7.9: Circuit 1. Alternative implementation for a common subcircuit of the 74181 circuit which implements the boolean equation E·D·(C + A·B) + !E

The impact of common design compiler-generated subcircuit replacement on pulse width improvement efficiency can be observed in Figure 7.11. The replacement circuits for Circuits

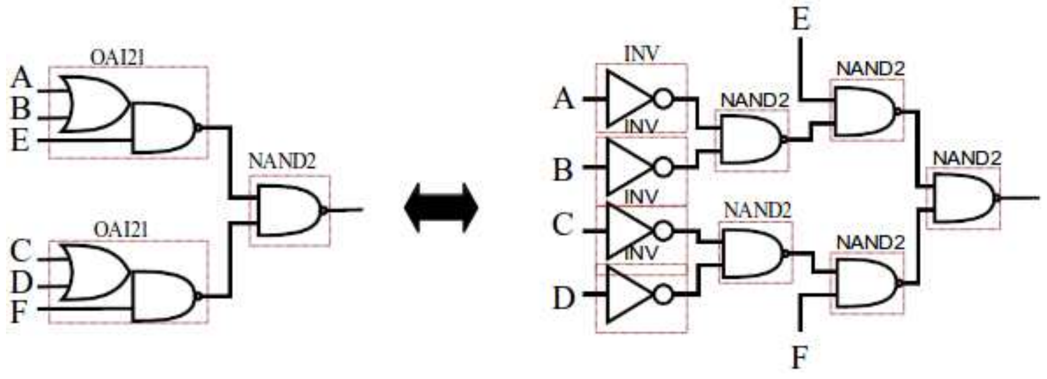


Figure 7.10: Circuit 2. Alternative implementation for a common subcircuit of the 74181 circuit which implements the boolean equation E·(A + B) + F·(C + D)

1 and 2 had approximately the same pulse width improvement efficiency despite Circuit 1's replacement circuit having twice the pulse width improvement as Circuit 2's replacement (Table 7.7). Also, the best implementations for each Boolean equation could not be formed by using the substitutions identified in Section 7.2.2 because this method does not account for the advantage of redundant circuitry. For instance, replacing the OAI21 cell in Circuit 1 with an implementation that contained an INV cell on the inputs would create a situation where two inverters are cascaded and therefore can be removed. However, the automated cell replacement method could encounter this effect randomly. Therefore, a check for redundant gates after individual cell replacement should further reduce the area impact of replacement with no effect on reliability.

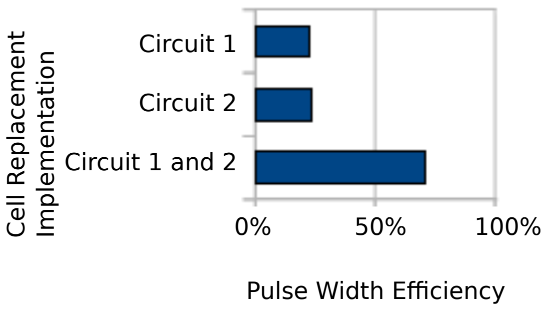


Figure 7.11: Pulse width efficiency calculations for the two common subcircuits of the 74181 circuit

Synthesis engines use certain library cells and cell chains with higher frequency. These

Table 7.7: Typical pulse width and area results for common subcircuit replacement within the 74181 design

Implementation	**Pulse Width (ps)**	**Area (μm^2)**
Original Circuit 1	1313.3	15.1
Replacement Circuit 1	240	36.7
Original Circuit 2	346.7	17.9
Replacement Circuit 2	284.4	34.8

frequently used cells can be more vulnerable than alternative implementations. The probability of an incident particle generating a transient can be reduced by removing highly vulnerable logic cells from the library, thus creating an improved cell library by construction. The results showed that replacing cells with alternative implementations can lower a circuit's typical pulse width by greater than 30% and typical drain area by greater than 40%. Replacing common subcircuits also provide improvement in the typical pulse width generated from ion strikes. For the results of the 74181 circuit, typical pulse width decreased more than 40%, and the pulse width efficiency was up to 70%.

7.3 Rule-based Reliability-Aware Synthesis

Radiation can cause soft errors in microelectronics. Soft errors are logical faults in a circuit's operation that do not reflect a permanent malfunction of the device. These errors are the result of particle strikes typically caused by the following (Srour and McGarrity, 1988; Dodd and Massengill, 2003): (1) alpha particles from package decay, (2) cosmic rays that produce energetic protons and neutrons, and (3) thermal neutrons.

Recent studies have shown that faults on a small set of circuit nodes disproportionately contribute to the majority of errors at the output (Mahatme et al., 2010; Srinivasan et al., 2005; Zhou and Mohanram, 2006). Therefore, these highly vulnerable nodes have been targeted for implementing cost-effective hardening techniques such as transistor sizing (Zhou and Mohanram, 2006). While transistor-sizing is useful in the full-custom integrated circuit design flow, very little research has considered the cost-effectiveness of selective hardening using the standard cell design methodology. Additionally, studies have investigated performance tradeoffs for Radiation-Hardened-By-Design (RHBD) standard cells (Lunardini et al., 2004), but not for high-performance standard cell designs that require high reliability.

The study in this section examines the impact that selective hardening has on the area, power, and delay of a circuit design. An open-source standard cell library was characterized for cell pulse width. Then, vulnerable circuit nodes in a set of combinational logic benchmark circuits were identified and replaced with more reliable cells. Finally, the impact of node hardening on performance was calculated. Results show that replacing vulnerable two-input cells with four-input cells improve reliability with little impact to power and area. Additionally, this method outperforms the replacement of cells with versions that have a higher drive strength (equivalent to transistor sizing) for each metric except delay. The increase in delay can be controlled by hardening vulnerable nodes that are not on critical timing paths.

7.3.1 Replacing Cells for Reduced Pulse Width

Figure 7.13 shows the transient pulses generated at the output of a NAND cell with the replacement options shown in Figure 7.12. The pulse widths for cells with higher drive strengths as well as extra inputs are smaller than the pulse width of the original cell, providing several options to improve reliability. For example, a NAND4x1 cell generates transient pulses similar to a NAND2x2 cell, but with different cost. This analysis was also conducted on the NOR cell, but it yielded opposite results for the three-input and four-input replacements (i.e., the pulse widths were greater than the two-input cell). The result for the NOR cell could be caused by the driving capacitance of the NOR cell since input "00" is in series and therefore decreases in value with additional inputs. The driving capacitance for the NAND cell for input "10" is in parallel, and therefore increases with increasing inputs.

Figures 7.16, 7.17, and 7.18 show the area, delay, and power trade-offs for improving the reliability of a two-input AND cell with the replacements shown in Figure 7.15. In the figures, *efficiency* is defined as the percentage decrease in pulse width divided by the percentage increase in the associated performance metric (e.g., area). The extra-input cell (i.e., "4-input AND") reduces the pulse width more efficiently in relation to power (Figure 7.18) than using the higher-drive-strength (i.e., "4x Drive Strength") or the alternate-logic cell (i.e., "NAND->INV"). Note that using three-input cells only reduces pulses generated

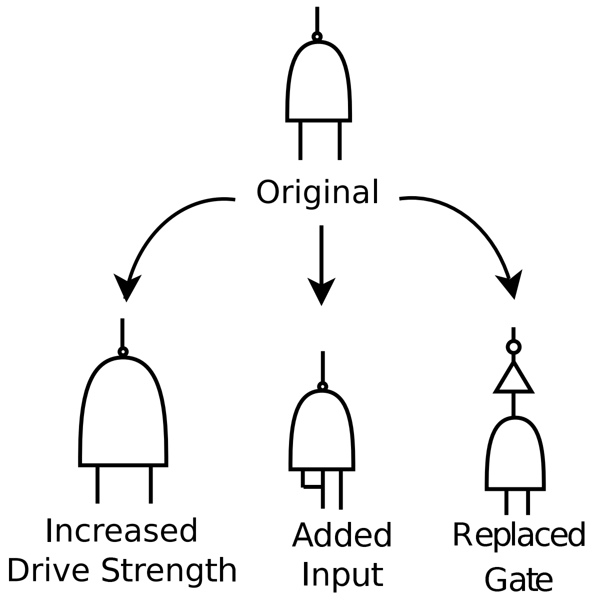


Figure 7.12: Selective node hardening techniques that use standard cells to replace a two-input NAND cell.

from the transistors connected to the redundant input. Also, increasing drive strength does not alter delay, so these results were omitted from Figure 7.17.

7.3.2 Replacing Subcircuits for Reduced POF

Selected ISCAS85 benchmark circuits (Brglez and Fujiwara, 1985) were characterized for reliability, chip area, worst-path delay, and leakage power in order to evaluate the efficiency of the hardening techniques discussed in this chapter. Reliability was determined using the probability of failure (POF) equation:

$$POF = P_{generation} \times P_{propagation} \times P_{latching} \tag{7.4}$$

where $P_{generation}$ is the probability that a transient will be generated in the circuit by the incident particle, $P_{propagation}$ is the probability that the transient will propagate to a storage element, and $P_{latching}$ is the probability that the transient will be latched. For the analysis in this chapter, $P_{generation}$ is assumed constant across all nodes in the circuit. $P_{propagation}$ is calculated probabilistically using a method introduced in (Asadi and Tahoori, 2005) and extended in (Limbrick et al., 2011). $P_{latching}$ is calculated as the difference of the transient

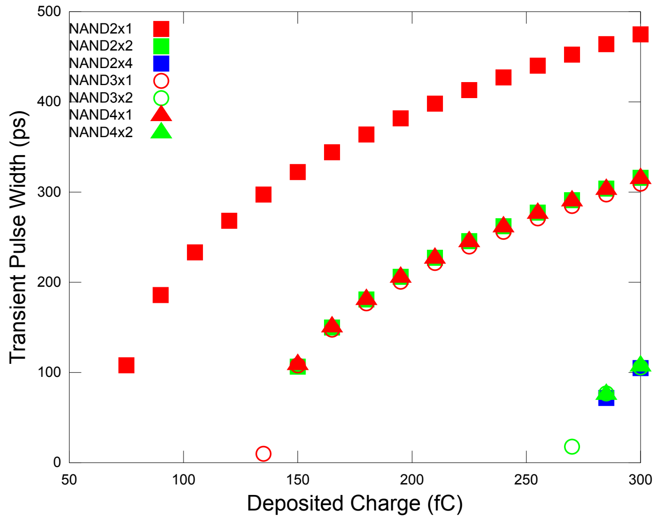


Figure 7.13: Pulse width vs. deposited charge for a two-input NAND cell with multiple drive strengths (“nand2x1”, “nand2x2”, “nand2x4”) and three- and four-input NAND cells functioning as a two-input NAND cell (“nand3x1”, “nand3x2”, “nand4x1”, “nand4x2”).

pulse width and the latching window (i.e., setup time + hold time) divided by the clock period (Shivakumar et al., 2002). The clock period used in this study was 1 ns (i.e., 1 GHz clock speed) and the latch-window value was taken from the specifications of a 1x drive strength D flip-flop.

All two-input NAND and AND cells with $P_{propagation}$ greater than 0.5 were replaced with four-input NAND and AND cells to improve the overall reliability of the circuit. Table 7.10 lists the reliability improvement and subsequent performance overhead associated with selectively hardening the circuit nodes. While the probability of failure reduced by as much as 20%, the area penalty stayed below 5%, and the increase in leakage power never exceeded 1%.

As soft errors become a greater problem for combination logic, selective node hardening has been identified as a viable mitigation technique. The results in this chapter demonstrated that selective node hardening can be effective when integrated into the standard cell design methodology, The technique identified in this chapter, replacing two-input cells with four-input equivalent cells, reduced pulse width more efficiently than the higher-drive-strength or alternate-logic cells. This replacement provided up to 20% improvement in reliability with less than 1% leakage power overhead.

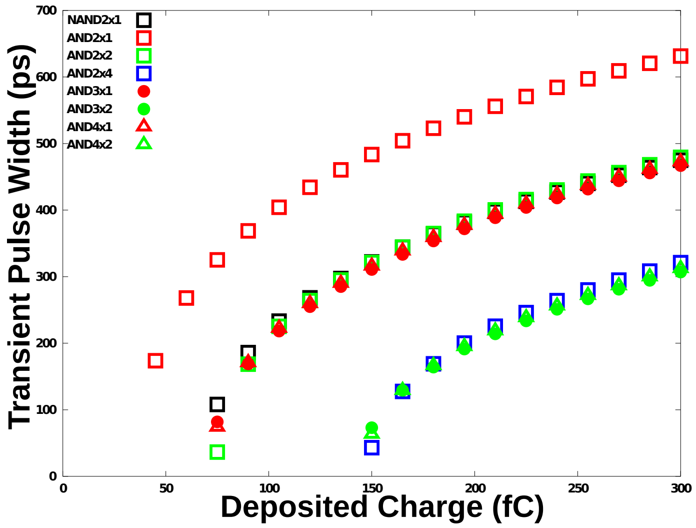


Figure 7.14: Pulse width vs. deposited charge for a two-input AND cell with multiple drive strengths ("and2x1", "and2x2", "and2x4"), three- and four-input AND cells functioning as a two-input AND cell ("and3x1", "and3x2", "and4x1", "and4x2"), and a two-input NAND cell ("nand2x1").

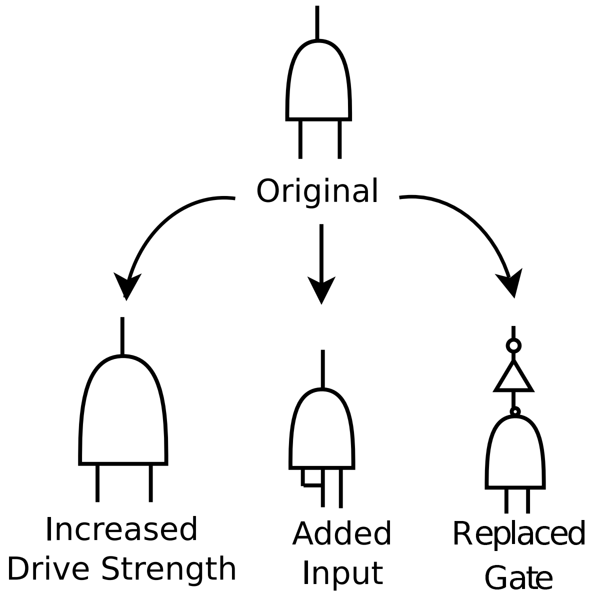


Figure 7.15: Selective node hardening techniques that use standard cells to replace a two-input AND cell.

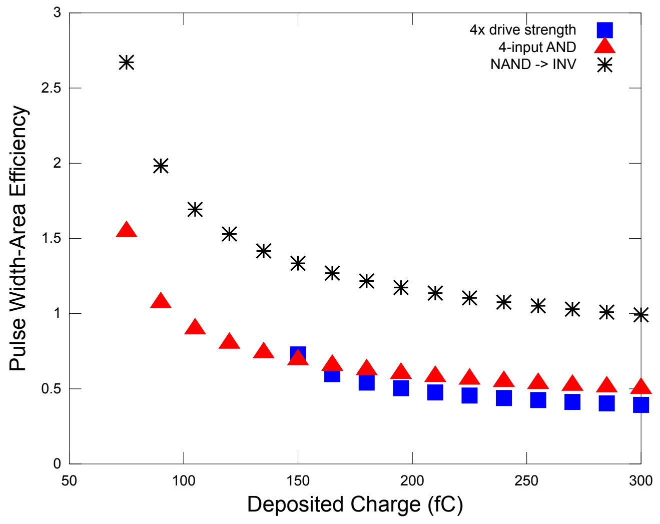


Figure 7.16: Pulse width efficiency (area) vs. deposited charge for a two-input AND cell with drive strength four times the original cell (“4x drive strength”), a four-input AND cell functioning as a two-input AND cell (“4-input AND”), and a two-input NAND cell connected to an inverter (“NAND -> INV”).

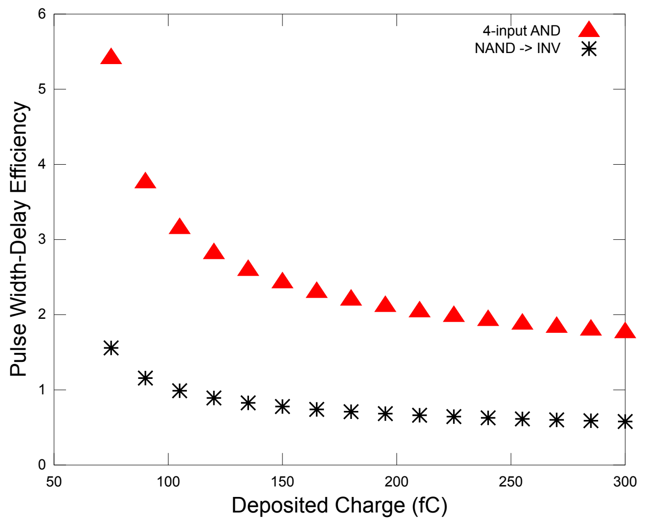


Figure 7.17: Pulse width efficiency (delay) vs. deposited charge for a four-input AND cell functioning as a two-input AND cell (“4-input AND”) and a two-input NAND cell connected to an inverter (“NAND -> INV”).

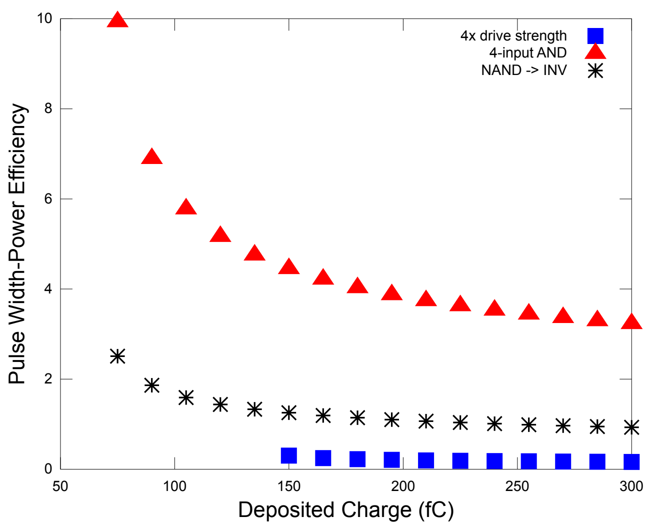


Figure 7.18: Pulse width efficiency (power) vs. deposited charge for a two-input AND cell with drive strength four times the original cell ("4x drive strength"), a four-input AND cell functioning as a two-input AND cell ("4-input AND"), and a two-input NAND cell connected to an inverter ("NAND -> INV").

Table 7.8: Percentage overheads in terms of area, power, and delay and Probability of Failure (POF) reduction of selective replacement of vulnerable nodes in the ISCAS85 benchmark circuit suite.

Circuit	**ΔArea (%)**	**ΔPower (%)**	**ΔPOF (%)**
C432	2.0	0.2	10.5
C499	2.3	0.1	4.8
C1908	3.8	0.3	11.3
C2670	2.7	0.2	12.2
C3540	1.9	0.2	13.4
C5315	4.9	0.6	21.2
C7552	4.1	0.4	15.7

Table 7.9: Low Area ($< 5\%$)

Circuit	**ΔPOF (%)**
C432	10.5
C499	4.8
C1908	11.3
C2670	12.2
C3540	13.4
C5315	21.2
C7552	15.7

Table 7.10: Low Power (< 1%)

Circuit	**ΔPOF (%)**
C432	10.5
C499	4.8
C1908	11.3
C2670	12.2
C3540	13.4
C5315	21.2
C7552	15.7

Table 7.11: Low Delay (< 1%)

Circuit	**ΔPOF (%)**
C432	24.2
C499	20.5
C1908	31.8
C2670	27.4
C3540	24.7
C5315	44.1
C7552	28.3

Chapter 8

Summary

In modern digital circuit design, the vulnerability of microelectronics towards faults increases as technology scales. In a subset of circumstances, faults result in an observable malfunction at the output. The likelihood of a fault reaching an output node is dependent upon the topology of the circuit. The goal of this work is to exploit the dependence on topology in order to improve the reliability of circuits during logic synthesis.

Logic synthesis tools use a combination of both heuristic and rule-based algorithms to map a Boolean network to a physical implementation. The use of these algorithms impact the topology of the circuit and subsequently the vulnerability of the circuit to soft errors. This research analyzed these algorithms to determine if opportunities to improve reliability can be exploited.

Modern EDA tools have standard optimizations for area, power, and delay constraints, but incorporating reliability information during synthesis remains a challenge. Additionally, the optimizations themselves affect reliability. The presented research analyzed the synthesized netlists for several logic circuits to determine their inherent sensitivity to radiation-induced transients. Depending upon the reduction and transformation of the design logic, the synthesis engine uses certain library cells and certain cell chains with a higher frequency. The placement of these cells along the logically sensitive paths influences the overall vulnerability of the design.

This vulnerability was captured through probabilistic graphical models to obtain the probability of propagation and latching through circuit-level and device-level simulation to obtain the probability of generation. These probabilities were combined to form a reliability metric.

The overall reliability metric that incorporates these vulnerability aspects can then be used to explore the design space of circuit for the most reliable implementation. This metric can aid in the identification of the impact of: (1) optimization algorithms, (2) design compiler settings, and (3) cell library characteristics on the reliability of a circuit. With this knowledge, an approach was developed to minimize circuit vulnerability based on cell selection.

Analysis from this dissertation has shown vulnerability trends in relation to area and timing optimizations. These trends were further investigated to develop an approach to minimize circuit vulnerability based on cell selection. Additionally, the electrical and latch-window masking factors were included in order to conduct a complete analysis of the vulnerability of the circuit. Additionally, reliability metrics were defined to allow for efficient exploration of reliable design implementations during the logic synthesis process.

8.1 Recommendations for Circuit Designers

The impact of logic synthesis on the reliability of digital integrated circuits has been shown in this dissertation. Based on the results of this dissertation, the designers of electronic design automation tools can improve reliability in the following ways:

1. **Target nodes with smallest logic depth.** The results from Chapter 5 showed that the nodes most likely to propagate an error have small logic depth.

2. **Separate the reliability problem by constraint.** Results from Chapter 6 showed that there are significant topological and cell usage differences between area- and timing constrained implementations.

3. **Replace common subcircuits to avoid an entire graph search.** In Section 7.2, the reliability of a circuit was shown to be efficiently reduced without identifying vulnerable nodes in advance.

4. **Replace vulnerable nodes with alternative cells based on reliability budget.** Each design constraint has a cell replacement strategy that reduces reliability while minimizing performance penalty (as shown in Section 7.3).

5. **Determine the reliability of each pattern tree.** The XOR study from Section 7.2 showed that the reliability can be improved by replacing the XOR pattern tree with a more reliable implementation.

8.2 Future Extensions

This research showed the effectiveness of reliability-aware synthesis at multiple stages in the synthesis process. The results from this study can aid in developing techniques to construct more reliable circuits that are independent of the common mitigation strategies that are currently implemented. This section explains how the research presented in this dissertation can be extended.

The identification of vulnerable nodes across a large network is currently an unsolved problem. The determination of circuit characteristics that impact reliability allows for the creation of tools that heuristically identify vulnerable nodes for circuits that are too large to calculate vulnerability using existing formal techniques. The results from this dissertation show that logic depth, capacitance, and cell adjacency can reasonably approximate vulnerability. However, the circuits would have to be physically implemented in order to guarantee that this approximation is accurate.

The findings in this dissertation have shown that the synthesis of circuits with area and delay constraints creates topological trends that can then be mapped to reliability. In future work, the trends related to physical layout can also be explored. For instance, in the case of multiple-node charge collection, the adjacency of cells influences the location of multiple faults. An analysis of cell placement algorithms could provide insight into the estimation and mitigation of these faults. Additionally, other constraints, e.g., power, can be analyzed to identify additional trends.

Finally, the replacement strategies developed in this dissertation proved effective in improving reliability while leveraging performance. This analysis was performed on two-input, one-output combinational logic cells. This work could be expanded to include cells with inputs greater than two (e.g., and-or-invert) and circuits with sequential elements.

REFERENCES

Aho, A. V. and Corasick, M. J. (1975). Efficient string matching: an aid to bibliographic search. *Communications of the ACM*, 18(6):333–340. 21

Aho, A. V. and Johnson, S. C. (1976). Optimal code generation for expression trees. *Journal of the ACM*, 23(3):488–501. 21

Albhin, J. R., Massengill, L. W., Bhuva, B. L., Narasimham, B., Gadlage, M. J., and Eaton, P. H. (2009). Single-event transient pulse quenching in advanced CMOS logic circuits. *IEEE Transactions on Nuclear Science*, 56(6):3050–3056. 38, 78

Alexander, D. R., Mavis, D. G., Brothers, C. P., and Chavez, J. R. (1996). Design issues for radiation tolerant microcircuits in space. In *IEEE Nuclear and Space Radiation Effects Conference Short Course*. 12

Alexandrescu, D., Costenaro, E., and Nicolaidis, M. (2011). A practical approach to single event transients analysis for highly complex designs. In *2011 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, pages 155–163. xi, 27, 28, 33, 36, 56, 57

Allen, A. O. (1990). *Probability, Statistics, and Queueing Theory: With computer science applications*. Academic Press. 29

Almukhaizim, S., Makris, Y., Yang, Y., and Veneris, A. (2006). Seamless intergration of SER in rewiring-based design space exploration. In *International Test Conference*, volume 2, page 833. Citeseer. 2, 53, 73

Anelli, G., Campbell, M., Delmastro, M., Faccio, F., Floria, S., Giraldo, A., Heijne, E., Jarron, P., Kloukinas, K., Marchioro, A., Moreira, P., and Snoeys, W. (1999). Radiation tolerant VLSI circuits in standard deep submicron CMOS technologies for the LHC experiments: Practical design aspects. *IEEE Transactions on Nuclear Science*, 46(6):1690–1696. 12

Asadi, G. and Tahoori, M. (2005). An analytical approach for soft error rate estimation in digital circuits. In *IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 2991–2994. 30, 87

Asadi, H. and Tahoori, M. B. (2010). Soft error modeling and remediation techniques in ASIC designs. *Microelectronics Journal*, 41(8):506–522. xi, 30, 31

Atkinson, N., Witulski, A., Holman, W., Ahlbin, J., Bhuva, B., and Massengill, L. (2011). Layout technique for single-event transient mitigation via pulse quenching. *IEEE Transactions on Nuclear Science*, 58(99):1–1. xi, 75, 77

Austin, T. M. (1999). DIVA: A reliable substrate for deep submicron microarchitecture design. In *32nd Annual International Symposium on Microarchitecture*, pages 196–207. IEEE. 13

Avizienis, A., Laprie, J.-C., Randell, B., and Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1(1):11–33. 1

Baumann, R. (2002). The impact of technology scaling on soft error rate performance and limits to the efficacy of error correction. In *International Electron Devices Meeting*, pages 329–332. IEEE. 1

Baumann, R. (2005). Radiation-induced soft errors in advanced semiconductor technologies. *IEEE Transactions on Device and Materials Reliability*, 5(3):305–316. v, 9

Bossen, D. and Hsiao, M. (1980). A system solution to the memory soft error problem. *IBM Journal of Research and Development*, 24(3):390–397. 13

Brayton, R. K., Rudell, R., Sangiovanni-Vincentelli, A., and Wang, A. R. (1987). MIS: A multiple-level logic optimization system. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 6(6):1062–1081. 2

Breuer, M. (1966). General survey of design automation of digital computers. *Proceedings of the IEEE*, pages 1708–1721. 1

Brglez, F. and Fujiwara, D. (1985). A neutral netlist of 10 combinational benchmark circuits and a target translator in Fortran. In *Proceedings of the International Symposium on Circuits and Systems*, pages 663–698. 40, 54, 87

Burnett, D., Lage, C., and Bormann, A. (1993). Soft-error-rate improvement in advanced BiCMOS SRAMs. In *31st Annual Proceedings., International Reliability Physics Symposium*, pages 156–160. 12

Calin, T. and Nicolaidis, M. (1996). Upset hardened memory design for submicron CMOS technology. *IEEE Transactions on Nuclear Science*, 43(6):2874–2878. 13

Cao, Y., Sato, T., Orshansky, M., Sylvester, D., and Hu, C. (2000). New paradigm of predictive MOSFET and interconnect modeling for early circuit simulation. In *Proceedings of the IEEE Custom Integrated Circuits Conference*, pages 201–204. 36

Carron, D. (2007). Silicon Chip 3D. v, 17

Chen, C. L. and Hsiao, M. Y. (1984). Error-correcting codes for semiconductor memory applications: A state-of-the-art review. *IBM Journal of Research and Development*, 28(2):124–134. 13

Constantinescu, C. (2002). Impact of deep submicron technology on dependability of VLSI circuits. In *Proceedings of the International Conference on Dependable Systems and Networks*, pages 205–209. 6

Dahlgren, P. (1995). A switch-level algorithm for simulation of transients in combinational logic. In *Proceedings of the Twenty-Fifth International Symposium on Fault-Tolerant Computing*, pages 207–216. 11

De Micheli, G. (1994). *Synthesis and optimization of digital circuits*. McGraw-Hill. v, vi, 2, 19, 20, 24

Detcheverry, C., Dachs, C., Lorfevre, E., Sudre, C., Bruguier, G., Palau, J. M., Gasiot, J., and Ecoffet, R. (1997). SEU critical charge and sensitive area in a submicron CMOS technology. *IEEE Transactions on Nuclear Science*, 44(6):2266–2273. 7

Dharchoudhury, A., Kang, S., Cha, H., and Patel, J. (1994). Fast timing simulation of transient faults in digital circuits. In *IEEE/ACM International Conference on Computer-Aided Design*, pages 719–726. 34

Dodd, P. E. and Massengill, L. W. (2003). Basic mechanisms and modeling of single-event upset in digital microelectronics. *IEEE Transactions on Nuclear Science*, 50(3):583–602. 1, 8, 13, 36, 44, 45, 85

Dodd, P. E., Sexton, F. W., Hash, G. L., Shaneyfelt, M. R., Draper, B. L., Farino, A. J., and Flores, R. S. (1996). Impact of technology trends on SEU in CMOS SRAMs. *IEEE Transactions on Nuclear Science*, 43(6):2797–2804. 7

Dodd, P. E., Shaneyfelt, A. R., Horn, K. M., Walsh, D. S., Hash, G. L., Hill, T. A., Draper, B. L., Schwank, J. R., Sexton, F. W., and Winokur, P. S. (2001). SEU-sensitive volumes in bulk and SOI SRAMs from first-principles calculations and experiments. *IEEE Transactions on Nuclear Science*, 48(6):1893–1903. 8

Fu, S., Mohsen, A. M., and May, T. C. (1985). Alpha-particle-induced charge collection measurements and the effectiveness of a novel p-well protection barrier on VLSI memories. *IEEE Transactions on Electron Devices*, 32(1):49–54. 12

Gadlage, M. J., Schrimpf, R. D., Narasimham, B., Pellish, J., Warren, K. M., Reed, R., Weller, R., Bhuva, B. L., Massengill, L. W., and Zhu, X. (2008). Assessing alpha particle-induced single event transient vulnerability in a 90-nm CMOS technology. *IEEE Electron Device Letters*, 29(6):638–640. vi, 26, 27

Hatchel, G. D. and Somenzi, F. (1996). *Logic synthesis and verification algorithms*. Springer. 20

Hazucha, P., Karnik, T., Walstra, S., Bloechel, B., Tschanz, J., Maiz, J., Soumyanath, K., Dermer, G., Narendra, S., De, V., and Borkar, S. (2003). Measurements and analysis of SER tolerant latch in a 90 nm dual-Vt CMOS process. In *Proceedings of the IEEE 2003 Custom Integrated Circuits Conference*, pages 617–620. 13

Jess, J. A. G. (2000). Designing electronic engines with electronic engines: 40 years of bootstrapping of a technology upon itself. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 19(12):1404–1427. 1

Keutzer, K. (1988). DAGON: technology binding and local optimization by DAG matching. In *Papers on Twenty-five Years of Electronic Design Automation*, pages 617–624. ACM. 21

Kinnison, J. D. (1998). Achieving reliable, affordable systems. In *IEEE Nuclear and Space Radiation Effects Conference Short Course*. 13

Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models*. MIT Press. 30

Krishnaswamy, S., Plaza, S., and Markov, I. (2007). Enhancing design robustness with reliability-aware resynthesis and logic simulation. In *IEEE/ACM International Conference on Computer-Aided Design*, pages 149–154, San Jose, California. IEEE Press. 2, 40, 53, 73

Lacoe, R. C., Osborn, J. V., Koga, R., Brown, S., and Mayer, D. C. (2000). Application of hardness-by-design methodology to radiation-tolerant ASIC technologies. *IEEE Transactions on Nuclear Science*, 47(6):2334–2341. 12

Lavagno, L., Martin, G., and Scheffer, L. (2006). *Electronic design automation for integrated circuits handbook - 2 Volume set*. CRC Press, Inc. v, 2

Leveugle, R., Calvez, A., and Maistri, P. (2009). Statistical fault injection: quantified error and confidence. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 502–506. 29

Limbrick, D. B., Mahatme, N. N., and Robinson, W. H. (2012a). Determining the efficacy of selective node hardening techniques using standard cells. In *13th European Conference on Radiation and Its Effects on Components and Systems*. 4

Limbrick, D. B., Mahatme, N. N., and Robinson, W. H. (2012b). Reliability-aware synthesis of combinational logic with minimal performance penalty. *IEEE Transactions on Nuclear Science, In Review*. 4

Limbrick, D. B. and Robinson, W. H. (2012). Characterizing single event transient pulse widths in an open-source cell library using SPICE. In *IEEE Workshop on Silicon Errors in Logic - System Effects (SELSE)*. In Print. 4

Limbrick, D. B., Yue, S., Robinson, W. H., and Bhuva, B. L. (2011). Impact of synthesis constraints on error propagation probability of digital circuits. In *2011 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, number 5, pages 103–111. 4, 87

Lin, T. T. Y. and Siewiorek, D. P. (1990). Error log analysis: statistical modeling and heuristic trend analysis. *IEEE Transactions on Reliability*, 39(4):419–432. 6

Lunardini, D., Narasimham, B., Ramachandran, V., Srinivasan, V., Schrimpf, R. D., and Robinson, W. H. (2004). A performance comparison between hardened-by-design and conventional-design standard cells. In *Workshop on Radiation Effects on Components and Systems (RADECS 2004)*, pages 1–5. 85

MacMillen, D., Butts, M., Camposano, R., Hill, D., and Williams, T. (2000). An industrial view of electronic design automation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 19(12):1428–1448. 1

Mahatme, N. N., Chatterjee, I., Patki, A., Limbrick, D. B., Schrimpf, R. D., Bhuva, B. L., and Robinson, W. H. (2010). An efficient technique to select logic nodes for single event transient reduction. In *15th European Conference on Radiation and Its Effects on Components and Systems*. 85

Massengill, L. W., Baranski, A. E., Van Nort, D. O., Meng, J., and Bhuva, B. L. (2000). Analysis of single-event effects in combinational logic-simulation of the AM2901 bitslice processor. *IEEE Transactions on Nuclear Science*, 47(6):2609–2615. v, 10, 37, 38

Mavis, D. G. and Alexander, D. R. (1997). Employing radiation hardness by design techniques with commercial integrated circuit processes. In *AIAA/IEEE Digital Avionics Systems Conference*, volume 1, pages 2.1–15–22 vol.1. 12

Mavis, D. G. and Eaton, P. H. (2002). Soft error rate mitigation techniques for modern microcircuits. In *40th Annual Reliability Physics Symposium Proceedings*, pages 216–225. 12, 36, 45

Messenger, G. (1982). Collection of charge on junction nodes from ion tracks. *IEEE Transactions on Nuclear Science*, 29(6):2024–2031. 34, 50

Mishchenko, A. and Chatterjee, S. (2006). DAG-aware AIG rewriting a fresh look at combinational logic synthesis. In *Proceedings of the 43rd Annual Design Automation Conference*, pages 532–535. 2, 73

Miskov-Zivanov, N. (2007). MARS-S: modeling and reduction of soft errors in sequential circuits. In *8th International Symposium on Quality Electronic Design*, pages 893–898. IEEE Computer Society. 37

Miskov-Zivanov, N. and Marculescu, D. (2006). MARS-C: modeling and reduction of soft errors in combinational circuits. In *Proceedings of the 43rd annual Design Automation Conference*, pages 767–772. 37

Mukherjee, S. (2008). *Architecture Design for Soft Errors*. Morgan Kaufmann. v, 6, 12, 29

Narasimham, B., Gadlage, M., Bhuva, B., Schrimpf, R., Massengill, L., Holman, W., Witulski, A., Reed, R., Weller, R., and Zhu, X. (2009). Characterization of neutron- and alpha-particle-Induced transients leading to soft errors in 90-nm CMOS technology. *IEEE Transactions on Device and Materials Reliability*, 9(2):325–333. 27

Ness, D. C. D., Hescott, C. C. J., and Lilja, D. D. J. (2007). Exploring subsets of standard cell libraries to exploit natural fault masking capabilities for reliable logic. In *Proceedings of the 17th ACM Great Lakes symposium on VLSI*, pages 208–211, Stresa-Lago Maggiore, Italy. ACM. 40

Nicolaidis, M. (1999). Time redundancy based soft-error tolerance to rescue nanometer technologies. In *Proceedings of the 17th IEEE VLSI Test Symposium*, pages 86–94. IEEE. 13

Nicolaidis, M. (2005). Design for Soft Error Mitigation. *IEEE Transactions on Device and Materials Reliability*, 5(3):405–418. 12, 53

Nicolaidis, M. (2011). *Soft Errors in Modern Electronic Systems*. Springer. vi, 44, 45

Parker, K. and McCluskey, E. (1975). Probabilistic treatment of general combinational networks. *IEEE Transactions on Computers*, 100(6):668–670. 30

Ramachandran, P., Kudva, P., Kellington, J., Schumann, J., and Sanda, P. (2008). Statistical fault injection. In *IEEE International Conference on Dependable Systems and Networks*, pages 122–127. IEEE. 29

Ramanarayanan, R., Degalahal, V. S., Krishnan, R., Kim, J., Narayanan, V., Xie, Y., Irwin, M. J., and Unlu, K. (2009). Modeling soft errors at the device and logic levels for combinational circuits. *IEEE Transactions on Dependable and Secure Computing*, 6(3):202–216. v, 11, 37, 38

Rao, R., Joshi, V., Blaauw, D., and Sylvester, D. (2009). Circuit optimization techniques to mitigate the effects of soft errors in combinational logic. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 15(1):5. ix, 73, 74

Rao, R. R., Chopra, K., Blaauw, D. T., Sylvester, D. M., and Member, S. (2007). Computing the soft error rate of a combinational logic circuit using parameterized descriptors. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(3):468–479. 37

Ray, J., Hoe, J., and Falsafi, B. (2001). Dual use of superscalar datapath for transient-fault detection and recovery. In *Proceedings of the 34th Annual ACM/IEEE International Symposium on Microarchitecture*, pages 214–224. IEEE Computer Society. 13

Reis, G., Chang, J., Vachharajani, N., Rangan, R., and August, D. (2005). SWIFT: Software implemented fault tolerance. In *Proceedings of the International Symposium on Code Generation and Optimization*, pages 243–254. IEEE Computer Society. 13

Rodgers, J. L. and Nicewander, A. W. (1988). Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66. 41

Rudell, R. L. (1989). *Logic synthesis for vlsi design*. PhD thesis. 2

Santarini, M. (2005). Cosmic radiation comes to ASIC and SOC design. *EDN*. 1

Shivakumar, P., Kistler, M., Keckler, S., Burger, D., and Alvisi, L. (2002). Modeling the effect of technology trends on the soft error rate of combinational logic. In *International Conference on Dependable Systems and Networks*, pages 389–398. IEEE. 1, 32, 88

Sosnowski, J. (1994). Transient fault tolerance in digital systems. *IEEE Micro*, 14(1):24–35. 13

Srinivasan, V., Sternberg, A. L., Duncan, A. R., Robinson, W. H., Bhuva, B. L., and Massengill, L. W. (2005). Single-event mitigation in combinational logic using targeted data path hardening. *IEEE Transactions on Nuclear Science*, 52(6):2516–2523. 37, 85

Srour, J. R. and McGarrity, J. M. (1988). Radiation effects on microelectronics in space. *Proceedings of the IEEE*, 76(11):1443–1469. 1, 85

Stine, J., Castellanos, I., Wood, M., Henson, J., Love, F., Davis, W., Franzon, P., Bucher, M., Basavarajaiah, S., Oh, J., and Others (2007). FreePDK: An open-source variation-aware design kit. In *IEEE International Conference on Microelectronic Systems Education*, pages 173–174. IEEE. vii, 36, 51, 52

Tosun, S., Mansouri, N., and Arvas, E. (2005). Reliability-centric high-level synthesis. In *Proceedings of the Design, Automation and Test in Europe*, pages 1258–1263 Vol. 2. 37

Warren, K., Sternberg, A., Weller, R., Baze, M., Massengill, L., Reed, R., Mendenhall, M., and Schrimpf, R. (2008). Integrating circuit level simulation and monte-carlo radiation transport code for single event upset analysis in seu hardened circuitry. *IEEE Transactions on Nuclear Science*, 55(6):2886–2894. 8, 27

Weaver, C., Emer, J., Mukherjee, S. S., and Reinhardt, S. K. (2004). Techniques to reduce the soft error rate of a high-performance microprocessor. In *Proceedings of the 31st Annual International Symposium on Computer Architecture*, pages 264–275. 13

Wo, Z. and Koren, I. (2005). Technology mapping for reliability enhancement in logic synthesis. In *Sixth International Symposium on Quality of Electronic Design*, pages 137–142. Published by the IEEE Computer Society. 74

Yang, S. (1991). Logic Synthesis and Optimization Benchmarks User Guide Version 3.0 (1991). Technical report, TR 1991-IWLS-UG-Saeyang, MCNC, Research Triangle Park, NC. 54

Zhang, B. and Wang, W. (2006). FASER: Fast analysis of soft error susceptibility for cell-based designs. In *7th International Symposium on Quality Electronic Design*, pages 6 pp.–760. 37, 40

Zhou, Q. and Mohanram, K. (2006). Gate sizing to radiation harden combinational logic. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 25(1):155–166. vi, 1, 34, 35, 36, 37, 50, 59, 85