Enhanced Design for Testability Circuitry for Test

Hui Jiang
Southern Methodist University, hui_j@outlook.com

Follow this and additional works at: https://scholar.smu.edu/engineering_electrical_etds

Part of the VLSI and Circuits, Embedded and Hardware Systems Commons

Recommended Citation
https://scholar.smu.edu/engineering_electrical_etds/45

This Dissertation is brought to you for free and open access by the Electrical Engineering at SMU Scholar. It has been accepted for inclusion in Electrical Engineering Theses and Dissertations by an authorized administrator of SMU Scholar. For more information, please visit http://digitalrepository.smu.edu.
ENHANCED DESIGN FOR TESTABILITY CIRCUITRY FOR TEST

Approved by:

Jennifer Dworak
Department of Electrical and Computer Engineering
Dissertation Committee Chairperson

Sukumaran Nair
Department of Electrical and Computer Engineering

Theodore Manikas
Department of Computer Science

Ping Gui
Department of Electrical and Computer Engineering

Gary Evans
Department of Electrical and Computer Engineering

Kundan Nepal
Department of Electrical and Computer Engineering
University of St. Thomas
ENHANCED DESIGN FOR TESTABILITY CIRCUITRY FOR TEST

A Dissertation Presented to the Graduate Faculty of the

Lyle School of Engineering

Southern Methodist University

in

Partial Fulfillment of the Requirements

for the degree of

Doctor of Philosophy

with a

Major in Computer Engineering

by

Hui Jiang

M.S., Computer Engineering, Southern Methodist University
B.S., Electronic Information Science, Shandong University, China

May 14, 2022
ACKNOWLEDGMENTS

This project is supported by NSF grant CCF-1061164 and CCF-1814928. I’m truly grateful for their financial support.

The first person I want to thank is Dr. Jennifer Dworak, who has been my advisor since 2014 when I was a Master student. Not only she guides me with her professional knowledge and patience for my academic progress, but also she stands by me when I have a bad time, and gives me advise like a big sister. Also, Dr. Manikas has been helping me since 2015 as my supervisor at the very beginning when I worked as a grader for his courses. From my courses to my academic research, he helps me with his academic experience and kindness. Moreover, I want to show my gratitude to Dr. Kundan Nepal (Chair of Electrical and Computer Engineering Department, University of St. Thomas) and Dr. Iris Bahar (Professor of Computer Science, Professor of Engineering, Brown University) for their wisdom and thorough academic advises. Further more, there are a lot more professors and people in my department, coworkers in my research team, and friends I am really grateful to.

Finally, my biggest gratitude goes to my families (my mother, father, elder sister and other relatives), who have been supporting me with love for decades.
<table>
<thead>
<tr>
<th>LIST OF CONTENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF FIGURES</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHAPTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Introduction</td>
</tr>
<tr>
<td>2. Previous Work</td>
</tr>
<tr>
<td>2.1. Fortuitous Detection with N-detect</td>
</tr>
<tr>
<td>2.2. Fault Cone Analysis</td>
</tr>
<tr>
<td>2.3. Transition Fault Detection</td>
</tr>
<tr>
<td>3. Detecting Defects Using a MISR Structure</td>
</tr>
<tr>
<td>4. Increasing Detections of the Otherwise Least Detected Stuck-at Faults during Scan Shift</td>
</tr>
<tr>
<td>4.1. Experimental Procedure</td>
</tr>
<tr>
<td>4.2. Experimental Results</td>
</tr>
<tr>
<td>4.3. Conclusion of N-detect of Stuck-at Faults</td>
</tr>
<tr>
<td>5. Further Increasing the Detection Number of Stuck-at Faults</td>
</tr>
<tr>
<td>5.1. Experimental Procedure</td>
</tr>
<tr>
<td>5.1.1. Flip-flop Selection for Maximum Cell-Aware Fault Coverage (Method 1) and Experimental Procedure</td>
</tr>
<tr>
<td>5.1.2. Flip-flop Selection for Maximum Stuck-At Fault Coverage (Method 2)</td>
</tr>
<tr>
<td>5.1.3. Flip-flop Selection from Fault Cone Results (Method 3 and 4)</td>
</tr>
<tr>
<td>5.1.4. Random Patterns Simulation</td>
</tr>
<tr>
<td>5.1.5. Summary Comparison of the Four Flip-Flop Selection Methods</td>
</tr>
<tr>
<td>5.2. Conclusion for Methods for Further Increasing the Number of Stuck-at Fault Detections</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1.</td>
<td>N-Detect Fortuitous Detection</td>
</tr>
<tr>
<td>3.1.</td>
<td>A Pattern Shifting in Scan Chain (Intermediate Patterns)</td>
</tr>
<tr>
<td>3.2.</td>
<td>Selected scan flip-flops are “shadowed” in a MISR so that a signature can be collected using intermediate shift data as test patterns.</td>
</tr>
<tr>
<td>4.1.</td>
<td>Experiment Flowchart</td>
</tr>
<tr>
<td>4.2.</td>
<td>Avg. Detection Number for Low-Detected Faults (Quadratic, Des56)</td>
</tr>
<tr>
<td>4.2.</td>
<td>Avg. Detection Number for Low-Detected Faults (Fm_rec, Colorconv)</td>
</tr>
<tr>
<td>4.2.</td>
<td>Avg. Detection Number for Low-Detected Faults (Fpu)</td>
</tr>
<tr>
<td>4.3.</td>
<td>Median of Detection Number for Low-detected Faults (Quadratic, Des56)</td>
</tr>
<tr>
<td>4.3.</td>
<td>Median of Detection Number for Low-detected Faults (Fm_rec, Colorconv)</td>
</tr>
<tr>
<td>4.3.</td>
<td>Median of Detection Number for Low-detected Faults (Fpu)</td>
</tr>
<tr>
<td>4.4.</td>
<td>Percent of Improved Low-Detected Stuck-at Faults (%)</td>
</tr>
<tr>
<td>5.1.</td>
<td>Experimental Flowchart</td>
</tr>
<tr>
<td>5.3.</td>
<td>Percent of Low-Detected Faults Gained Extra Detections with the MISR (Quadratic, Des56)</td>
</tr>
<tr>
<td>5.3.</td>
<td>Percent of Low-Detected Faults Gained Extra Detections with the MISR (Fm_rec, Colorconv)</td>
</tr>
<tr>
<td>5.3.</td>
<td>Percent of Low-Detected Faults Gained Extra Detections with the MISR (Fpu)</td>
</tr>
</tbody>
</table>
6.1. Per Flip-Flop UC Fault Coverage with 1%, 2%, 4%, 16% and 32% Flip-flop Vertical Lines ................................................................. 42

7.1. An Example of Transition fault ............................................................ 45
7.2. Transition Fault Detection Signal ....................................................... 47
7.3. Selected scan flip-flop in a scan chain is “Backed up” with a bypass flip-flop so that transition faults can be captured in the original scan chain without launch-on shifting the “normal” shifted or captured value. ............... 49
7.4. Transition Fault Scan Chain Enhancing DFT Structure Work Flow (1. Capture Cycle, 2. Cycle 1 After Capture) ........................................... 50
7.5. Transition Fault Scan Chain Enhancing DFT Structure Work Flow (5. Cycle 4 After Capture, 6. Cycle 5 After Capture) ............................... 52
7.6. Transition Fault Scan Chain Enhancing Detection Experimental Flow .......... 54
7.7. Intermediate Pattern Sampling Example (Chain Length 10) .................... 56
7.8. Transition Fault Coverage with Stuck-at ATPG Patterns, Intermediate Patterns and Selected Flip-flops ..................................................... 60
7.9. Percent of Flip-flops Required for Transition Faults Detected by Stuck-at Intermediate patterns ............................................................. 61
7.10. Transition Fault Coverage with varying overhead of flip-flop ................. 62
7.11. Transition Fault Coverage with Stuck-at ATPG Patterns, Intermediate Patterns, Selected Flip-flops, and Stuck-at Pattern Chain Test .............. 63
7.12. Transition Fault Coverage with varying overhead of flip-flop and Stuck-at Pattern Chain Test .............................................................. 64
7.13. Transition Fault Coverage with Stuck-at ATPG Patterns, Intermediate Patterns, Selected Flip-flops, and Stuck-at Pattern Chain Test .............. 65
7.14. Top-off Pattern for Commercial Tool Launch-off Shift Transition Fault Coverage with Selected Flip-Flops (Fm_rec, Colorconv) ....................... 67
7.14. Top-off Pattern for Commercial Tool Launch-off Shift Transition Fault Coverage with Selected Flip-Flops (Fpu)
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1. Characteristics of Circuits</td>
</tr>
<tr>
<td>4.2. Percentage of Flips-Flops in MISRs</td>
</tr>
<tr>
<td>5.1. Methods of Selection Flip-Flop Lists</td>
</tr>
<tr>
<td>7.1. Transition and stuck-at fault detection required for the two types of transition faults</td>
</tr>
<tr>
<td>7.2. Characteristics of Circuits</td>
</tr>
<tr>
<td>7.3. LoS ATPG Pattern Set Characteristics</td>
</tr>
</tbody>
</table>
More stringent defect detection requirements have led to the creation of new fault models, such as the cell-aware fault model, that attempts to model defects that might be missed by traditional test sets. Unfortunately, the resulting test sets can be long, and thus we have previously explored a DFT-based (Design for Testability-based) approach to reduce test time by harnessing scan shift cycles for cell-aware defect detection. This approach uses a MISR (Multiple Input Signature Register) structure to capture data on the functional inputs to selected scan flip-flops during the shifting procedure. The final signature in the MISR can then be compared to an expected signature.

While this DFT structure was shown to be very effective for detecting static cell-aware faults, other faults that are not explicitly modeled as cell-aware faults are also possible. The $n$-detect test approach was previously proposed to detect faults fortuitously by increasing the number of times that simpler faults (e.g. stuck-at faults) are detected. In this dissertation, we investigate the ability of our DFT MISR circuitry to provide multiple fault detections of the “hardest to detect” stuck-at faults during scan shift. We will show that significant additional stuck-at fault detections are possible in the circuits studied, even when only a subset of all scan chain flops are used for scan shift capture. We explore which flip-flops should be shadowed to increase the value of $n$ for the least detected stuck-at faults, and we then identify which circuit characteristics are most important for determining the cost of the MISR needed to achieve high values of $n$. For example, circuits that contain a few flip-flops with upstream fault cones that cover a large percentage of all faults in the circuit.
can often achieve high n-detect coverage fortuitously with a low-cost MISR. This allows a DFT engineer to predict the viability of this MISR-based approach early in the design cycle.

On the other hand, modern integrated circuits (ICs) must be tested for both static and delay defects. As a result, transition fault testing is an important component of modern testing for delay defects. Thus, to detect transition faults, we propose another DFT scan chain structure. Specifically, a Capture Control signal and a Shift Control signal allow the circuit to capture data within the original scan chain while still recovering the original shift pattern and the original captured results at the end of shifting. This dissertation will show that a significant number of transition faults can be detected during scan shift when a stuck-at ATPG pattern set is applied and sometimes shifted at-speed in a “Launch-on-Shift” (LoS) mode of operation. Furthermore, with selected top-off LoS patterns generated by a commercial tool, higher transition fault coverages are obtained than even with the commercial tool’s original launch-on shift transition fault pattern set.
Chapter 1
Introduction

Parts of this chapter were first presented in [1] and [2].

When chips are manufactured, it is easy for defects to occur in them. High defect coverage during circuit testing is required to help ensure that defective parts are not sold to customers. During automatic test pattern generation (ATPG), abstract models of defects, known as faults, are generally targeted deterministically. Historically, the most commonly used fault model has been the stuck-at fault. However, as new defect mechanisms appeared, and as the quality requirements became more stringent, the number of fault models that may be targeted has increased over time. Today, these fault models include bridges, opens, transition faults, and path delay faults, among others. While most of these fault models have focused on the interconnections between gates or standard cells in a design, more recently, the cell-aware fault model [3–9] has been proposed to better model the types of defects that may occur within standard cells. This model uses layout information from the standard cell to identify potential defect locations and analog analysis to determine defect behavior. The detection conditions are then abstracted to logic values at the inputs to the standard cell that can be targeted with a traditional ATPG tool.

While the inclusion of new fault models may improve defect coverage if those models are well-matched to the defects that actually occur, it is still possible for other defect behavior to remain unmodeled. For example, even when cell-aware faults are targeted, defects that occur between standard cells may be missed. Thus, another significant thrust of testing research has involved the probabilistic, as opposed to deterministic, detection of unmodeled defects. Some examples of purely probabilistic approaches include pseudo-random or weighted random testing [10]. Such approaches are commonly used for Logic Built-In Self Test (LBIST). Data may be applied to circuits and captured on each clock cycle (test-per-clock) [11] or they may be applied to scan chains such that a full scan pattern is shifted in before test data is
An alternative approach that combines aspects of deterministic and probabilistic techniques is n-detect. N-detect test sets increase the fortuitous detection of unmodeled defects by ensuring that other faults (e.g., stuck-at faults) are detected multiple times during test. The reasoning behind the n-detect approach arises from the realization that for the detection of any fault or defect, excitation (generating an error at the defect site) and observation (propagating the error to an observable output or scan cell) must be satisfied. The excitation requirements change from one type of fault or defect to another. However, the common requirement for detecting any fault or defect at a circuit site is the fact that the value at that site must be observed. Stuck-at faults are especially useful targets for n-detect because detecting all stuck-at faults at least \( n \) times guarantees that every circuit site is observed at least \( 2n \) times—\( n \) times when the site value is one and \( n \) times when the site value is zero. Thus, targeting stuck-at faults multiple times increases the chance of the excitation requirements of an unknown and unmodeled defect being fortuitously met while the site is observed—hopefully leading to detection of the unmodeled defect.

Unfortunately, all of these approaches may lead to long test pattern sets and long test times. For example, other researchers have previously shown that on average a 49% increase in pattern count was required when compared to a normal stuck-at test set when “cell-internal defects” and “traditional cell port faults” were targeted in ten industrial circuits [9]. (The same paper also showed that a 2.5% increase in defect coverage was possible if a test set of the same length as a stuck-at test set was optimized for cell-aware fault detection.) N-detect test sets also require a significant number of additional patterns—especially for high values of \( n \).

While many test compaction and compression approaches have been proposed to reduce test pattern counts and test data volume, alternative methods have used enhanced Design for Testability (DFT) circuitry to try to make faults “easier” to detect in some way. Common approaches include the use of test points to enhance the observability or controllability of lines in the circuit. Other approaches try to increase the flexibility of the flip-flops and scan chains in the design to control and capture data during test. For example, a test-per-clock approach was proposed in [11] that adds a multiplexer and XOR gate to every flip-flop in the design.
so that different chains may be in one of three modes during test: stimulus, compaction, or mission mode. While deterministic techniques and probabilistic techniques tend to result in long test sets, a goal of this dissertation is to achieve high N-detect coverage with scan chain enhancing circuitry that can fortuitously increase the number of fault detections without increasing the size of test set.

In [12], we investigated the ability of a selected set of shadow flops combined into a MISR (Multiple Input Signature Register) to detect cell-aware faults by capturing data during scan shift. Unlike more traditional fault models (which model faults on the inputs and outputs of gates), cell-aware faults model defects that may occur within a logic gate or standard cell. The detection of a cell-aware fault requires the detection of an appropriate stuck-at fault on the output of the gate potentially along with additional conditions that must be satisfied on the gate’s inputs. These additional conditions make the cell-aware fault more difficult to detect than the corresponding stuck-at fault—requiring additional patterns for full cell-aware coverage and leading to increased test time. Capturing data during shift allows a large increase in the number of effective testing clock cycles with no increase in the actual test time. Our experiments showed that a significant fraction of the detectable static cell-aware faults that would otherwise be missed by a stuck-at test set could be captured during scan shift—reducing the number of additional patterns needed to achieve full cell-aware coverage.

This dissertation extends this previous work and will explore how this previously proposed DFT approach targeted toward cell-aware fault detection can also improve the number of stuck-at fault detections with no increase in pattern count. In particular, we explore the ability of a MISR placed into the circuit to detect cell-aware faults during scan shift to also increase the value of \( n \) for faults in a stuck-at test set. We will show that even when the percentage of flip-flops shadowed in the MISR is very low to reduce area overhead, some stuck-at faults that may have been detected only once in the original test set achieve many more detections. Such an increase can help prevent test escapes, especially for unmodeled defects (such as shorts between standard cells), that are associated with locations containing stuck-at faults that are detected only once.

Because the original flip-flop selection method was focused on cell-aware fault detections, next we explore whether other approaches to selecting flip-flops are better suited for increas-
ing the number of detections of the least detected stuck-at faults. Increasing the detections of those faults is important because it is the corresponding locations that are more likely to contribute to test escapes and an unacceptably large defective part level. Thus, in addition to considering using the flip-flops selected for maximum cell-aware fault coverage in [12] in the MISR, we also propose three more flip-flop selection methods.

Furthermore, our previous work has shown that the trade-off between flip-flop overhead and fault coverage can vary significantly from circuit to circuit. Therefore, we also explore multiple ways of quickly determining whether or not a circuit is amenable to the proposed approach. Because the detection of the faults with the MISR relies on fortuitous detection, we explore the difference that arises when testability is estimated using the random pattern simulation functionality in a commercial tool instead of simulating the actual intermediate patterns. We will show why the easier to implement random pattern simulation serves as a good lower bound on the fault coverage that can be achieved.

Then, the ability of fault cone analysis to reveal fault testability and to indicate the hardware overhead needed for good fault coverage will be investigated. We will also show that fault cone analysis can be used to explain the different fault coverages achieved and hardware overheads required for the different circuits that were studied.

While all of the faults discussed so far have been static defects, modern integrated circuits (ICs) must be tested for both static and delay defects. Unfortunately, test sets for delay defects are generally considerably longer than those needed for static defects, and it is more difficult to obtain high fault coverage. This is especially problematic when a circuit must be tested in the field, such as on startup or shut-down, because aging tends to cause circuits to become slower over time—introducing the opportunity for new failures to appear. Thus, a means of testing for delay defects without long test times is also needed.

Unlike static faults, transition faults model a large lumped delay at a gate input or output. However, like a static cell-aware fault, a transition fault requires the detection of a stuck-at fault at the gate output with an additional condition that must be satisfied—albeit in the previous clock cycle. In circuits with scan chains, transition faults are often detected with the last shift of the chain, where the last shift launches the transition, and the final pattern in the chain is used to detect the stuck-at fault on the gate’s output. Intuitively,
if a MISR is used to capture scan chain data during shift, it could be possible to detect a transition fault during scan shift as well—provided that the capturing of the data occurs “at speed.”

However, using a MISR to detect delay defects during scan shift introduces an additional difficulty. In particular, if the delay from the circuit’s logic to the scan cell is very different than the delay from the circuit’s logic to the shadow flip-flop that captures data in the MISR, then the results of a delay-based test will be unreliable. Ideally, we would capture the test results in the original scan cells instead because it is the delay to those cells that will actually be seen in functional mode. However, capturing data in those cells will overwrite the pattern being shifted in.

Thus, in the third part of this dissertation, we will introduce an alternative structure to allow data to be captured in the original chain on scan shift to detect delay defects while ensuring that the values in the chain at the end of the shift procedure correspond to the pattern that is to be applied. We will show that a high percentage of the detectable transition faults can be fortuitously detected with this approach even when only a static stuck-at test set is used to fill the chains. Launch-on Shift (LoS) ATPG top-off patterns will also be added to detect any transition faults that still remain undetected for but for which a commercial tool could efficiently generate a test. The final fault coverages obtained are even higher than the commercial tool’s coverages due to fortuitous detections of faults that the commercial software found too complex. In addition, the minimum number of LoS ATPG patterns are selected as well to achieve only the commercial tool’s LoS fault coverage for a fair comparison.

The rest of this dissertation is organized as follows. Chapter 2 describes some previous work for n-detect fortuitous detection with scan chain enhanced DFT circuitry, other detection methods with fault cone analysis and transition fault detection. Chapter 3 reviews the DFT modifications described in [12] for detecting static cell-aware faults. Chapter 4 describes the experimental setup of stuck-at detection with n-detect, shows simulation results and explores conclusions from stuck-at fault n-detect results. Chapter 5 investigates additional flip-flop selection methods to improve the detection counts for stuck-at faults that originally have only a few detections. Then, Chapter 6 applies fault cone analysis to explain
the simulation results we obtained, as well as the effect of circuit characteristics on this scan chain enhancement. For transition fault detection, Chapter 7 proposes a new DFT design that allows extra captures in the original scan chain during scan shift without permanently overwriting the ATPG patterns being shifted in or the results from the previous test pattern that are being shifted out. The experimental procedure, simulation results, and conclusions are also shown in the chapter. Finally, Chapter 8 concludes the dissertation, and future work will be discussed.
Chapter 2

Previous Work

Parts of this chapter were first presented in [1] and [2].

2.1 Fortuitous Detection with N-detect

The difficulty of ensuring that all potential defects are well-modeled and targeted has led many researchers to explore n-detect testing [13–39]. Excitation (generating an error at the defect site) and observation (propagating the error to an observable output or scan cell) must be satisfied for any defect to be detected. While excitation requirements often vary from one type of defect to another, any defect that appears at a particular site must have its value propagated to an output. Thus, observation is a common detection requirement among many types of defects [14].

Stuck-at faults have relatively simple excitation requirements. In particular, to excite a fault P stuck-at 0, a logic 1 must be placed at site P. In contrast, to excite fault P stuck-at 1, a logic 0 must be present at site P. Other types of static faults, such as bridges and cell-aware faults generally require the detection of a stuck-at fault to occur, with some extra conditions that need to be simultaneously satisfied. For example, in the case of a dominating bridge, the extra condition corresponds to the logic value at the other bridged site. In the case of a cell-aware fault, extra conditions may correspond to extra logic values that must be set at the inputs to a standard cell.

As a result, the test space (i.e. set of input patterns) that detects a static defect is often a subset of the patterns that detect a stuck-at fault at the same site. This is shown in Figure 2.1, where the square represents all patterns that detect a stuck-at fault, and the circle represents the patterns that also excite and detect an unmodeled defect at that site. If multiple patterns (represented by the X’s) are chosen randomly to detect the stuck-at fault multiple times, the chance that at least one of them will fall in the circle and also detect the
unmodeled defect increases. The more patterns that detect the stuck-at fault, the higher the probability is. This is the fundamental reason why higher values of \( n \) can improve defect detection.

Many researchers have investigated different aspects of \( n \)-detect test sets [13–39]. The authors of [15] performed experiments on small physical circuits and found that no defects were missed when all stuck-at faults were detected at least 15 times and tests were applied at rated speed. The authors of [14] explained why \( n \)-detect test sets could be more effective due to the increasing number of site observations. The authors of [16] showed that the effectiveness of \( n \)-detect tests would increase if the layout were considered, and if the logic values in the physical neighborhood around a signal line were varied during additional detections.

This was followed by experiments on physically-aware \( n \)-detect test sets applied to industrial circuits in [17]. The authors of [18] also investigated ways of varying the test patterns by using different propagation paths for different detections of the fault.

Other papers have looked at the value of adding one more detection for various values of \( n \) and showed that additional detections for small \( n \) are much more important than additional detections for large values of \( n \) [19,20]. In fact, the value of additional detections decreases
exponentially as the value of $n$ increases [19]. Similarly, [21] looked at three definitions to measure the degree to which test quality began to saturate as the value of $n$ increased. The generation of test sets that target the stuck-at faults detected rarely by a one-detect test set $n$ times and target other faults until they are detected at least $m$ times ($1 \leq m < n$) for detection of four-way bridging faults was studied in [22].

Analysis of test set sizes for $n$-detect test sets and ways of reducing those sizes were explored in [23–31, 39]. For example, [23] reordered the test patterns so that the first $K$ patterns would increase unmodeled fault coverage (bridging faults and transition faults in the paper). This would allow the end of the pattern set to be truncated if necessary (e.g. when the entire set did not fit in the tester memory) while maximizing the quality of the remaining set.

An Integer Linear Programming (ILP) algorithm for minimizing test set size was proposed in [24]. Later, in [25], different from the ILP method, another heuristic method was also applied to reduce test set size. In [30], for $n$-detection of transition faults, stuck-at fault patterns were reordered for higher fault coverage of delay faults without increasing test set size by a factor of $n$. A test set generation method based on fault dominating relations was also proposed in [31] to reduce $n$-detect test set size. Moreover, an Embedded Multi-Detect ATPG Algorithm that utilizes the don’t-care bits to achieve additional multiple detections of faults (stuck-at and bridging in the paper) without increasing the size of the test set beyond the size of the original one-detect ATPG pattern test set size was described in [26]. With extra hardware overhead, logic implications for $n$-detect test set reduction was also investigated in [39]. However, [29] showed that ATPG tools trying to maximize the number of detections in a pattern could be biased against meeting some types of excitation conditions if those conditions were likely to prevent additional site observations.

Other papers have also explored the use of weighted pseudo-random BIST for $n$-detect [13] and the relaxation of $n$-detect test patterns to increase the number of unspecified don’t-care bits that can later be used for other applications such as low power or test compression [32]. The use of $n$-detect test sets for multiple-fault diagnosis was considered in [33, 34].

The first part of this dissertation applies a DFT enhancement that uses shadow flops collected into a MISR to detect cell-aware during scan shift [12] to the $n$-detect problem. We
will show that multiple additional detections of many of the least detected stuck-at faults are possible without any increase in the size of the test set.

2.2 Fault Cone Analysis

Fault cones have previously been studied in the context of multiple applications (e.g. [18, 38, 40–45]). For example, based on fault cones, [18] proposed a new method to obtain various distinct propagation paths for faults for \( n \)-detect. The authors of [40,41] used a fault cone technique to investigate internal structural information of the circuit to help identify failing scan cells for more accurate fault diagnosis. The authors of [42] introduced a method to test failures caused by replacing or cascading gates incorrectly in reversible circuits with fault cones. [43] investigated extra constraints during test generation for timing exceptions (to ensure the correctness of the pattern set) to improve test coverage decreased by timing exceptions, where fault cones helped in finding the hard timing exceptions. The authors of [44] graded the extent of strong fault-secureness with fault cone analysis (no fault-masking or path-sensitization if the fault cone sets are disjoint) for circuit robustness based on the detectability of faults. Similarly, also for robustness, in [45], an online fault-space pruning method for fault injection had been proposed. Analysis was performed by exploring the fault propagation to the next clock cycle state and fault masking with the circuit logic elements’ fault cones.

In the second part of this dissertation, the effect of four flip-flop selection methods and two types of patterns will be studied for the creation of the MISR in our DFT enhancement circuitry. Two of the flip-flop selection methods will use fault cones to reduce the time required to make the selection. Then, fault cone analysis will be further applied to study the circuit characteristics that influence whether a particular circuit is well-matched to our proposed approach [1]. This analysis will also apply to the results obtained in [12] regarding the number of flip-flops that had to be shadowed in the MISR for five different circuits.

2.3 Transition Fault Detection

The transition fault model has been one of the most widely used fault models to obtain high defect coverage. Because exciting a transition fault requires that the correct logic value be assigned to the fault site in a pre-conditioning pattern, the detection of a transition fault
is more difficult than the detection of the corresponding stuck-at fault [46]. In some cases, transition faults may be difficult or impossible to detect due to state assignments [47,48]. In addition, the authors of [49] stated that even when delay faults are testable with structural test patterns, they may not be detectable in the functional mode due to logic or timing constraints or both.

Various DFT methods have been proposed to improve transition fault coverage. The authors of [50] proposed a method that utilized on-die delay sensing and test point insertion for delay-fault testing. In [48], researchers reduced the number of untestable transition faults by enhancing a controller design with the addition of state transition pairs on invalid states.

However, other researchers have noted that high delay fault coverage of functionally unsensitizable faults may lead to yield loss and reduced designer productivity if they cause faults that would never fail in functional mode to be targeted and detected during test [51]. To address this, [52] applied test patterns in functional mode while attempting to optimize their tests with hazard-based transitions so that the delay faults detected would not be redundant under real operating conditions.

Some additional algorithmic approaches for transition test generation, simulation, and/or analysis include [53] and [54]. The authors of [53] provided several fault collapsing procedures for transition faults based on dominance relations. An efficient method was invented by the authors of [54] for transition delay fault coverage based on a multi-valued algebra, critical path tracing, and deductive fault simulation.

In addition to traditional transition fault models, small delay faults and cell transition faults have been proposed to improve delay coverage. The authors of [55] indicated that traditional fault models are insufficient for guaranteeing small delay defect detection. Similarly, in [56] an enhanced transition fault model was proposed to improve CMOS transition stuck-open fault coverage for stuck-open fault detection inside the CMOS cells.

As already noted, fault detection during scan shift is another DFT method that can help detect faults (e.g. [11], [57], [58]). More recently, [59] also targeted detection during scan shift with test point insertion; however, they were not for improving delay fault coverage. Thus, in the third part of this dissertation, a novel DFT circuit design is proposed to allow fortuitous transition fault detections to occur when a stuck-at test set is shifted through the
chains, with appropriate shifts occurring at-speed.
Chapter 3  
Detecting Defects Using a MISR Structure

Parts of this chapter were first presented in [1].

Much of the time and power expended during test are not directly used to detect defects. For example, a significant fraction of all testing clock cycles are generally not used for detecting defects when test sets are applied using scan chains. Instead, those clock cycles are devoted to shifting the next pattern into the chain while shifting out the previous pattern’s capture data. Moreover, during scan shift, the circuit logic is generally not isolated from the chains, which generates a significant amount of toggling of the internal circuit sites during shift and leads to extra power draw during shift.

Previously, researchers have tried multiple approaches to try to reduce the clock cycles and power wasted during shift. This includes attempts to create multiple shorter chains (which may be fed by an on-chip decompressor) and thus reduces the shifting clock cycles (e.g. EDT structure in [60] and Illinois Scan architecture in [61]). Similarly, the authors of [62] have applied this concept to IJTAG by having multiple chains shifting in parallel. Adjacent fill and other low power shift approaches have also been proposed to reduce the test power [63].

However, even with the optimizations above that reduce the length of the chains, some shift cycles are still needed—and that time and power expended during shift are still “wasted” from the perspective of detecting new, as yet undetected, defects. Thus, in our previous work, we devised a DFT approach to detect defects during scan shift. Essentially, the values in the chain in the middle of the shifting procedure each form an “intermediate pattern” that can be used for defect detection. One example of intermediate pattern generation is illustrated in Figure 3.1 with a five-bit long scan chain. The first line in the figure shows a state of capture. In this state, capture values “01010” are captured in five flip-flops, while a pattern “11011” will be shifted into the scan chain. Then, in the next clock cycle, the first bit of the
pattern “1” is shifted in. Partial previously captured values “0101” remain in the scan chain. At this time, we call this combination of the pattern being shifted in and the capture values in the chains being shifted out as one intermediate pattern. Similarly, the following three cycles also correspond to another three intermediate patterns generated from this shifting procedure. For a circuit with $m$ flip-flops in the scan chain, there are $m - 1$ intermediate shift patterns for each pattern shifted into the circuit under test (CUT).

To capture test results without overwriting the pattern being shifted in, some additional circuitry is needed to capture the results. In [12], we explored shadowing some flip-flops on the chains and using these shadow flops to capture data during scan shift to increase cell-aware fault detection. Furthermore, by collecting the shadow flops into a MISR, it is possible to obtain a single signature that combines test results over multiple scan shift cycles. An example of this DFT-based approach is shown in Figure 3.2 in the presence of an on-chip decompressor and compactor.

With this DFT approach, [12] achieved high cell-aware fault coverage by capturing data during scan shift and allowed the number of patterns needed to achieve detections of all

![Figure 3.1: A Pattern Shifting in Scan Chain (Intermediate Patterns)](image)
detectable static cell aware faults to be reduced. In fact, it was generally possible to detect most of the cell-aware faults using only a standard stuck-at fault ATPG test set that captured data during shift. Furthermore, for two circuits, all of the cell-aware faults that would have been missed by a stuck-at test set were detected by the intermediate patterns during scan shift—requiring no additional patterns to achieve full cell-aware fault coverage. Note that all of these detections of cell-aware faults by intermediate shift patterns corresponded to fortuitous detections. The patterns were not altered to deterministically increase the cell-aware detections. Only the selection of flip-flops shadowed in the MISR directly targeted the cell-aware faults. Specifically, to avoid the overhead involved in shadowing all flip-flops, the flip-flops were sorted in the order of detecting cell-aware faults from the most to the least. Then, only the flip-flops at the top of the list were included in the MISR.
Just as intermediate shift patterns are capable of detecting cell-aware faults fortuitously, it is reasonable to assume that they are also capable of detecting defects that do not necessarily follow the cell-aware fault model. Because previous research has shown that increasing the number of detections of the least detected stuck-at faults can provide good additional defect coverage.
Parts of this chapter were first presented in [1].

Based on the previously described DFT circuitry, experiments have been performed to explore how the circuitry can improve the detection count of stuck-at faults. As already stated, this is important because the extra detections may help prevent test escapes (i.e. defective parts that pass the tests), especially for faults originally detected less than 15 times [15].

4.1 Experimental Procedure

Experiments in this dissertation were performed on five circuits obtained from opencores.org [64]. Table 4.1 contains the characteristics of these circuits from smallest to largest. For each circuit, the number of flip-flops is listed in column 2, the number of stuck-at faults in column 3, the number of stuck-at ATPG patterns in column 4, and the number of intermediate patterns in column 5.

<table>
<thead>
<tr>
<th>Circuit</th>
<th># of Flip-Flops</th>
<th># of Stuck-At Faults</th>
<th># of ATPG Patterns</th>
<th># of Intermediate Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quadratic</td>
<td>120</td>
<td>8166</td>
<td>36</td>
<td>1044</td>
</tr>
<tr>
<td>Des56</td>
<td>193</td>
<td>13788</td>
<td>119</td>
<td>2856</td>
</tr>
<tr>
<td>Fm_rec</td>
<td>501</td>
<td>19888</td>
<td>406</td>
<td>10962</td>
</tr>
<tr>
<td>Colorconv</td>
<td>584</td>
<td>38518</td>
<td>98</td>
<td>3136</td>
</tr>
<tr>
<td>Fpu</td>
<td>5231</td>
<td>297358</td>
<td>538</td>
<td>17216</td>
</tr>
</tbody>
</table>
Figure 4.1 shows the overall procedure used for data collection. First, stuck-at ATPG test patterns were obtained using a commercial tool, and intermediate patterns were extracted using the procedure described in [12]. We assume that all flip-flops are initialized to zero before the first pattern is shifted in. All circuits studied contained on-chip decompressors, and the values actually shifted through the chains were the values used to obtain the intermediate shift patterns. The number of patterns simulated for each circuit is shown in Table 4.1.
The second step involves selecting the flip-flops that will be shadowed. A stuck-at fault that is not in the upstream cone of influence of a shadowed flip-flop cannot gain any new fault detections. Thus, an upper bound for increasing the value of \( n \) for a test set occurs when 100% of the flip-flops are shadowed. However, this overhead is likely to be prohibitively high for most circuits, and thus, experiments were also run where the flip-flops selected for shadowing corresponded to those that would have been chosen to achieve high static cell-aware coverage. Maximum cell-aware fault coverage can be obtained for each circuit using the percentage of flip-flops specified in Table 4.2. Note that this percentage is circuit dependent and appears to be a function of circuit size and circuit structure.

<table>
<thead>
<tr>
<th>Quadratic</th>
<th>Des56</th>
<th>Fm_rec</th>
<th>Colorconv</th>
<th>Fpu</th>
</tr>
</thead>
<tbody>
<tr>
<td>% of Flip-Flops</td>
<td>28.33</td>
<td>35.23</td>
<td>2.20</td>
<td>6.34</td>
</tr>
</tbody>
</table>

Because even this overhead may be too expensive for some circuits, experiments were also run for smaller overheads. To choose which flip-flops would be shadowed for the lower overhead experiments, we first sorted the flip-flops by the number of as yet undetected cell-aware faults that could be detected by each flip-flop from maximum to minimum. This list was then used to select flip-flops, in order, for overheads corresponding to approximately 1%, 2%, 4%, 8%, etc. up to the percentage shown in Table 4.2. Because we can only shadow an integer number of flip-flops, in each case, we selected a set of flip-flops that is larger than, but close to the percentage we desired (i.e. the number of flip-flops to be shadowed was rounded up).

Next, two different groups of simulations were done. First, an ATPG per-pattern fault simulation was done using the original ATPG test set. This provides the detection number for each fault in normal scan test and thus the initial value of \( n \) for each fault. This is shown in the path on the left of Figure 4.1.
Then, the additional detections that could be achieved by capturing data in the shadow flops during scan shift were obtained. In particular, the intermediate shift patterns were fault simulated while all flip-flops that would not be included in the MISR were masked. This ensured that only the shadowed flops could be used for fault detection. The sum of these two detection numbers for each fault corresponds to the total detection number achieved during test using scan shift.

Using R, we then filtered out the data for those faults that were detected 1 to 15 times by the original stuck-at fault test set. These are the faults most likely to contribute to test escapes [15,27,28,31,37–39] and for which increasing detection counts is most important. No additional work was done to determine if aliasing in the MISR would lead to some missed fault detections. This will be explored in future work. However, experiments performed in [12] showed very low aliasing for cell-aware faults for those simulated MISRs and selected capture patterns.

4.2 Experimental Results

Figure 4.2 shows the average detection numbers obtained when shadow flops and intermediate shift patterns are used for detecting the least detectable stuck-at faults for each of the OpenCores circuits. In each figure, the faults are grouped by the original number of stuck-at fault detections. Thus, the x-axis corresponds to the original detection number (1 to 15) for the original stuck-at fault ATPG patterns. The y-axis corresponds to the average number of detections for the faults in that group when some (or all) of the flip-flops are shadowed. Each line corresponds to a different shadow flip-flop overhead, where the overhead roughly doubles each time: 1%, 2%, 4%, etc. up to the overhead specified in Table 4.2, which corresponds to the shadow flop set needed to achieve the maximum possible cell-aware fault coverage using intermediate shift patterns. Note that the set of selected flip-flops with a higher overhead is always a superset of the set of selected flip-flops with a lower overhead. (In other words, the set of selected flip-flops with 4% overhead is a superset of the flip-flops selected for 2%, which is itself a superset of the flip-flops selected for 1%.) Finally, the top line corresponds to the number of fault detections that occur when all flops are shadowed. This corresponds to the maximum detection count possible for that test set under these conditions. Also, note
Figure 4.2: Avg. Detection Number for Low-Detected Faults (Quadratic, Des56)
Figure 4.2: Avg. Detection Number for Low-Detected Faults (Fm_rec, Colorconv)
that the y-axis scale is not linear to better show the impact of additional detections when the detection count is small. It is these additional detections that are most important.

For a given original detection number, as the flip-flop overhead is increased, the average (and median in Figure 4.3) detection numbers should either increase or stay the same. It should also not be lower than the original detection number.

Thus, this gives a measure of the magnitude of the number of additional detections for low-detected stuck-at faults. Although the graphs show variable behavior, it is clear that the number of additional detections for these faults is significant. Although not shown in the figures, if only the faults that have an improvement in detection counts are considered, then even faults that were detected only once in the original test set may easily be detected more than ten times on average for all the circuits. Ideally, this would mean that defects associated with those sites are highly likely to be detected during test and are unlikely to contribute to test escapes.
Figure 4.3: Median of Detection Number for Low-detected Faults (Quadratic, Des56)
Figure 4.3: Median of Detection Number for Low-detected Faults (Fm_rec, Colorconv)
Figure 4.3 shows the median detection number for the same experimental results. Increasing trends can be seen for all five circuits when more flip-flops are shadowed in the MISR. Note that dots with the same median numbers may hide behind the other dots with different flip-flop overheads.

As we can see from Figures 4.2 and 4.3, adding shadow flip-flops in a MISR can allow some of these faults to be detected many more times. However, the lines are not monotonically increasing. This is simply due to the fact that different faults with different detection requirements are involved. If many of the faults in an original detection count group are very difficult to detect at the selected flip-flops, they can actually cause the overall average detection count for that group of faults to be lower than the overall average for faults in a group with a smaller number of original detections. Recall that the original detection counts were based upon detections at all flip-flops. Thus, faults that were easier to detect with all flip-flops considered may be harder to detect when only a subset of the flip-flops are chosen.
The pseudo-random character of the intermediate patterns can also lead to randomness in the detection of the faults.

Figure 4.4: Percent of Improved Low-Detected Stuck-at Faults (%)

Figure 4.4 clearly illustrates the effect that adding more shadow flip-flops has on the number of low-detected faults that can achieve higher detection counts. The seven bars for a circuit cluster correspond to the shadow flip-flop overhead, i.e. 1%, 2%, 4%, 8%, 16%, 32%, 100%, as the legend shows. The height of each bar corresponds to the percentage of low-detected faults that show any increase in detection counts during scan shift by capturing test data using intermediate shift patterns.

For most of the circuits, at least 30% of the faults in the circuit were able to obtain additional detections even when only approximately 1% of the flip-flops were shadowed. (Recall that flip-flops that were more likely to detect more cell-aware faults were selected.
first.) The exception to this is the circuit Des56. This circuit appears to have an architecture where the fault cones are distributed relatively equally among the flip-flops. (Fault cones will be covered in detail in Chapters 5 and 6). Thus, covering most of the faults requires including most of the flops in the MISR.

Also note that even when all of the flip-flops were shadowed, some faults showed no improvement in detection counts for some circuits. Such faults are likely to be very difficult to detect—whether that is due to excitation requirements, observation requirements, or a combination of the two. Even with a relatively large number of intermediate shift patterns, they are not caught fortuitously and thus likely need to be targeted deterministically through additional ATPG effort if a higher value of $n$ is needed. Fortunately, because many other faults are detected fortuitously, the number of patterns needed for even a relatively high value of $n$ is likely to be much less than would be required with a traditional scan chain without a MISR.

4.3 Conclusion of N-detect of Stuck-at Faults

We have shown that this DFT approach used to achieve high cell-aware fault coverage by capturing data during scan shift can also significantly increase the number of detections of stuck-at faults. This is true even for many of the stuck-at faults that would otherwise be detected relatively few times by a stuck-at test set. Furthermore, the number of fortuitous detections for these faults is often large, indicating that they are probably less likely to contribute to test escapes than faults that do not obtain additional detections.

The number of detections and flip-flop overhead required to achieve good detection counts is circuit-dependent. In particular, circuits such as Des56 that appear to distribute the fault cones of the hard faults across more flip-flops require more flops to be shadowed. In contrast, circuits that have a few flip-flops that have a large portion of the circuit in their upstream cone of influence are likely to be more amenable to achieving additional fault detections with lower overhead.
Chapter 5
Further Increasing the Detection Number of Stuck-at Faults

Chapter 4 has shown that the approach introduced in [12] can significantly increase the number of times each stuck-at fault is detected, even for faults detected only a few times by a stuck-at test set. As previously discussed, the number of times a stuck-at fault is detected is important because each detection provides an additional chance of meeting the additional excitation requirements for detecting an unmodeled defect at a fault site above and beyond those needed for the detection of the stuck-at fault itself.

While Chapter 4 showed that the number of additional detections of stuck-at faults achievable with a MISR selected for the purpose of detecting cell-aware faults during scan shift was large, several questions remained to be answered. For example, Chapter 4 did not consider the possibility that even more detections could be achieved with a MISR designed to maximize the detections of stuck-at faults (instead of cell-aware faults) during shift. Moreover, the selection method was time consuming (runtime could be a month for the largest circuit among the five circuits), and it required explicit generation and simulation of intermediate patterns—requiring a test set to be available as well.

Thus, to explore better sets of flip-flops to be shadowed in the MISR and to estimate the percentages of improved stuck-at faults without long simulation time and intermediate pattern generation, three more flip-flop selection methods are studied in this chapter. To save fault simulation time, fault cone analysis, which reports the flip-flops to which faults propagate, was used for this selection. To avoid the intermediate pattern generation procedure, random patterns were applied instead.

5.1 Experimental Procedure

Experiments in this chapter were also run on the same five circuits obtained from Opencores.org [64]. Table 4.1 shows the characteristics of each circuit.
Figure 5.1 shows the overall procedure used for data collection. Applying the procedure with stuck-at ATPG patterns and intermediate patterns will be illustrated first as an example. Then, the procedure with random patterns will be introduced.

First, stuck-at ATPG test patterns were obtained using a commercial tool, and intermediate patterns were extracted using the procedure described in [12]. The number of patterns simulated for each circuit is shown in Table 4.1. Then, to select the set of flip-flops to be shadowed in the MISR in an efficient and effective manner, ordered lists of flip-flops were
generated with four methods. All four methods for flip-flop selection are listed in Table 5.1. For each method, flip-flops were sorted in the order of the number of faults covered by each flip-flop in the circuit from maximum to minimum. Method 1 will be used as an example to explain the experimental procedure. Then, simulation results with all four flip-flop selection methods and two types of patterns will be shown. More details regarding Methods 2, 3 and 4, and the other type of pattern will be explained later.

### Table 5.1: Methods of Selection Flip-Flop Lists

<table>
<thead>
<tr>
<th>Method</th>
<th>Fault List Reporting</th>
<th>Sorting Algorithm</th>
<th>Full Fault List</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Selected for Max Cell-Aware Fault Coverage</td>
<td>Simulated with all stuck-at ATPG and intermediate patterns</td>
<td>Greedy algorithm</td>
<td>Cell-aware faults can be detected by cell-aware ATPG patterns but missed by stuck-at ATPG patterns</td>
</tr>
<tr>
<td>2. Selected for Max Stuck-At Fault Coverage</td>
<td>Simulated with all stuck-at ATPG and intermediate patterns</td>
<td>Greedy algorithm</td>
<td>Stuck-at faults can be detected by stuck-at ATPG patterns</td>
</tr>
<tr>
<td>3. Selected from Fault Cone Report Simply</td>
<td>Fault cone report</td>
<td>Simply sorting</td>
<td>N/A</td>
</tr>
<tr>
<td>4. Selected from Fault Cone Report with Greedy Algorithm</td>
<td>Fault cone report</td>
<td>Greedy algorithm</td>
<td>All unique UC faults reported to D pins of all FFs</td>
</tr>
</tbody>
</table>

#### 5.1.1 Flip-flop Selection for Maximum Cell-Aware Fault Coverage (Method 1) and Experimental Procedure

Method 1 corresponds to the flip-flop selection method introduced in [12]. With Method 1, flip-flops were selected for maximum coverage of detectable cell-aware faults that would not be fortuitously detected by stuck-at ATPG test patterns applied in the normal way. The
following greedy algorithm was used to select the flip-flops. First, through fault simulation of stuck-at ATPG patterns and the corresponding intermediate patterns, lists of cell-aware faults detectable at each flip-flop were obtained. Then, the flip-flops were sorted by the number of cell-aware faults they could detect from maximum to minimum. The flip-flop at the top of the list with the highest cell-aware fault detections would be selected to be shadowed in the MISR. After that, faults covered by the selected flip-flop would be removed from the fault list. The remaining flip-flops would be resorted in the order of number of cell-aware faults they could detect that were still included in the fault list. The process would continue until the desired coverage was achieved or all flip-flops are selected. If the fault list became empty before the required number of flip-flops were selected, then the remaining flip-flops were selected in order from the last available sorted list.) Thus, an upper bound for hardware overhead occurs when 100% of the flip-flops are shadowed.

Although Method 1 selects the flip-flops based upon cell-aware fault detections, we are still interested in the ability of the MISR to detect stuck-at faults multiple times. As a result, all stuck-at ATPG patterns generated by the commercial tool were simulated with the entire scan chain being used for fault detection. This provided the normal fault detection number for all of the stuck-at faults not just the least detected faults). Then, the intermediate patterns (no stuck-at ATPG patterns included) were also simulated with flip-flops shadowed in the MISR, while all flip-flops that would not be included in the MISR were masked. This ensured that only the shadowed flops could be used for fault detection. The sum of these two detection numbers for each fault corresponds to the total detection number achieved during test using scan shift with the MISR. The experiments were run with the same hardware overheads for all circuits, i.e. 1%, 2%, 4%, 8%, 16%, 32% and 100% (the number of flip-flops to be shadowed was rounded up). The number of extra detections of each of these faults when the MISR was used was calculated. Then, the percentage of the hard-to-detect faults whose detection counts increased was calculated for each MISR hardware overhead. Once again, aliasing in the MISR was not considered.

Figure 5.3 shows simulation results for the five different circuits. Each sub-figure (a), (b), (c), (d), and (e) of Figure 5.3 corresponds to a different circuit. The x-axis corresponds to the hardware overhead of the MISR, and the different bars with corresponds to the different
Figure 5.3: Percent of Low-Detected Faults Gained Extra Detections with the MISR (Quadratic, Des56)
Figure 5.3: Percent of Low-Detected Faults Gained Extra Detections with the MISR (Fm_rec, Colorconv)
Figure 5.3: Percent of Low-Detected Faults Gained Extra Detections with the MISR (Fpu)

methods of selecting and analysing the flip-flops to be included in the MISR. The data for Method 1 as described above corresponds to the first bar on the left of each overhead cluster. As can be seen in Figure 5.3, with Method 1, at least some of the low-detected faults are detected more times for all of the hardware overheads. Obviously, as the hardware overhead increases, the number of faults whose detection count increase goes up as well. However, even with only 1% hardware overhead, the number of faults whose detection counts increase is significant.

5.1.2 Flip-flop Selection for Maximum Stuck-At Fault Coverage (Method 2)

Method 1 selected the flip-flops based on cell-aware fault coverage. In Method 2, we investigated the effect of targeting the stuck-at faults instead. Thus, similar to Method 1, Method 2 used the greedy algorithm to iteratively select flip-flops. However, the selection was based on stuck-at fault coverage. Because the goal was to increase the number of times that each stuck-at was detected, the full stuck-at fault list was targeted with this method.
The corresponding results for Method 2 with intermediate patterns correspond to bar 3 of each hardware overhead in Figure 5.3. The figure shows that the percentage of low-detected stuck-at faults detected with the shadow flops is generally higher than Method 1 when stuck-at faults are targeted in the flip-flop selection method when at least 2% hardware overhead is used. However, this is not always true for a hardware overhead of 1%. This is likely due to the fact that all stuck-at faults were targeted during flip-flop selection. Because Method 1 targeted cell-aware faults not detected by the ATPG test set, the targeted cell-aware faults were more likely to be present at hard-to-observe circuit locations. These correspond to the same locations that are likely to have few stuck-at fault detections. Thus when the allowed overhead of the MISR is small, it is important to target the least detected stuck-at faults if the goal is to achieve better stuck-at \( n \)-detect fault coverage of the least detected faults.

5.1.3 Flip-flop Selection from Fault Cone Results (Method 3 and 4)

Even though Methods 1 and 2 can improve the detection number of the low-detected stuck-at faults by a large amount, the simulation time required for these two methods is considerable. To quickly estimate the effect of the MISR without spending simulation time as with Method 1 and 2, fault cone methods (Method 3 and Method 4) were proposed. Fault cone methods generate fault lists based on circuit structure. Faults reported to the D pin of a certain flip-flop means that the D pin of this flip-flop is in the fault cone of the faults reported. In another words, it may be possible for these faults to be propagated to this flip-flop and to be captured in this flip-flop when simulating with patterns. Thus, fault cone results show the possibility of flip-flops detecting faults, and how hard a fault is to detect can be roughly based on the number of flip-flops that the faults can propagate to (the fewer flip-flops, the harder). According to our fault cone report of the five experimental circuits [64], six fault classes were reported, i.e. ATPG_Untestable (AU), DET_Implication (DI), Tied (TI), UNControlled (UC), UNUsed (UU), Blocked (BL) faults. Based on the definition of these fault classes, we considered only UC faults to be potentially detected with pattern stimulus. Thus, we considered the UC fault number as the detection metric of the fault cone method for each flip-flop.

Method 3 sorted the flip-flops based on number of UC faults reported to each flip-flop,
regardless of overlap among the fault lists of the flip-flops. In contrast, Method 4 sorted flip-flops according to number of uncovered UC faults in the full UC fault list of the circuit from maximum to minimum and used the greedy algorithm from Methods 1 and 2 for selection. This meant the faults already covered by selected flip-flops would be removed from the target UC fault list.

Although the fault cone method was employed to avoid simulation while selecting the flip-flops for Methods 3 and 4, to determine the real effect of these methods, simulation was done to evaluate their effectiveness. Results with intermediate patterns are shown as bar 5 of each hardware overhead sample cluster in Figure 5.3 for Method 3, and bar 7 for Method 4. Both Methods 3 and 4 yield lower values for the percentage of improved faults than Method 2. However, the fault cone method is much quicker. Because it uses the greedy algorithm that considers which stuck-at faults have already been detected and removes them from the fault list, the results are better for Method 4 than for Method 3 for most of the circuits (i.e. except for the circuit Colorconv). The results for Colorconv may be due to the fact that fault cone analysis can be a less accurate way to estimate fault detections versus simulating with a pattern stimulus.

5.1.4 Random Patterns Simulation

While fault cone flip-flop methods (Method 3 and 4) did not require simulation time when selecting flip-flops, our analysis in the last section still used simulation of intermediate patterns from the actual test set to determine their effectiveness for improving detection counts. To obtain such an analysis without the need to already have an ATPG test set and generate the intermediate patterns, here we consider the simulation of random patterns instead.

The same number of random patterns as intermediate patterns were simulated with the same selected flip-flops for each method, and the results were processed with the same procedure as described in the previous section. Simulation results in bars 6 and 8 of each flip-flop selection cluster show the results with random patterns. The percentage of improved faults is usually slightly lower than the simulation results obtained with intermediate patterns (bars 5 and 7), except for Fpu (where larger differences are shown). Thus, with fault cone
selection methods (Method 3 and 4) and random patterns, the user can quickly estimate the effect of this MISR method.

The flip-flops selected with Methods 1 and 2 were also simulated with random patterns to further compare the effect of simulating intermediate versus random patterns. Each simulation result with random patterns can be seen in bars 2 and 4 of each flip-flop sample cluster in Figure 5.3. When comparing effect of the two types of patterns, a larger percent of low-detected faults gained extra detections when simulated with intermediate patterns than random patterns most of the time, and the results were very close in the other cases when the percentages were not larger, except in the case of Fpu.

This is reasonable when we consider that the PI (Primary Input) settings of the previous stuck-at ATPG patterns were kept the same when shifting a new pattern into the scan chain, i.e. the PI settings of intermediate patterns are the same as those of actual stuck-at ATPG patterns. However, the PI settings for random patterns were truly random. Even though the intermediate patterns are random-like due to the fact that each intermediate pattern is a combination of part of the captured result values that are being shifted out and part of the new pattern being shifted in, keeping the same ATPG PI settings make intermediate patterns more capable of detecting faults than random patterns. Thus, with random patterns, the testability of faults with the inclusion of the MISR structure can be shown before generating intermediate patterns for a new circuit, and a higher percent of improved low-detected faults can usually be expected with later intermediate pattern simulation.

5.1.5 Summary Comparison of the Four Flip-Flop Selection Methods

In Figure 5.3, all circuits with every flip-flop selection method and both types of simulated patterns can see faults obtain extra detections to various degrees. Among all of the flip-flop selection methods, Method 2 was able to help all circuits obtain the largest percent of improved faults for all flip-flop overheads larger than 1%, and for some circuits (e.g. Quadratic, Des56, Colorconv) this was true even with 1% overhead. This may be due to:

1. Using an alternative model for fault detection analysis during flip-flop selection (e.g. cell-aware faults for Method 1) can lead to less effective results than using the targeted fault. (e.g. Here the targeted faults are stuck-at faults.)
2. The fault cone results in Method 3 and 4 only consider circuit structure in flip-flop selection, but no pattern stimulus. The fault lists reported by fault cone may not be entirely accurate. Our other previous experiments showed that the detection number of faults detected with pattern simulation were always fewer than the detection number reported based on fault cone.

In Fpu, not only the primary input values of the intermediate patterns, but other circuit or fault characteristics can matter when trying to improve fault detections. In Figure 5.3(e), a larger difference can be seen for the percentage of improved faults with intermediate patterns compared with random patterns than any other circuit. Moreover, for Fpu, even with 100% of the flip-flops shadowed in the MISR, not all faults can gain extra detections. In contrast, detection counts of almost all faults can be increased in the same simulation case for the other circuits. This is consistent with the results for cell-aware faults reported in [12] for this circuit. Stuck-at ATPG patterns and intermediate patterns could not get all cell-aware faults of Fpu covered, so that a non-negligible number of cell-aware ATPG top-off patterns needed to be added to guarantee full cell-aware fault coverage. One possible reason could be due to the character of the fault detection requirements in Fpu. In particular, there may simply be more faults with difficult excitation or observation requirements that must be satisfied for fault detection to occur. This would make fortuitous detection with the “somewhat random” intermediate patterns much harder and would also make good assignments to the primary inputs potentially much more important.

As for Fm_rec and Des56, their simulation results are two extremes. Fm_rec can improve the detection number of most stuck-at faults with a small percent of flip-flops included in the MISR (after reaching 2% overhead in Figure 5.3 (c)), while Des56 requires a large overhead (Figure 5.3 (b)). This was also true for cell-aware fault detection in [12]. Only 2.2% of the flip-flops were required to be shadowed in the MISR for maximum cell-aware fault coverage; however, the highest percentage, 35.23%, was required for Des56. Further fault cone analysis will be done in next chapter to explain this phenomenon.
5.2 Conclusion for Methods for Further Increasing the Number of Stuck-at Fault Detections

In this chapter, we ran simulation experiments with four flip-flop selection methods and two types of simulation patterns for extra detections of low-detected stuck-at faults. Our experimental results showed that selecting the flip-flops from a ranked list sorted based on the maximum stuck-at fault coverage obtained at each flop can highly improve the percentage of low-detected stuck-at faults that gain extra detections with the MISR structure. The detection count increases were higher than that achieved with the other three methods (including Method 1 introduced in [12]). The larger percentage of improved faults is significant. This analysis is important because the more detections of stuck-at faults, the more likely a test is to detect other untargeted faults or unmodeled defects. Also, a lower bound of the expected benefit to be achieved by adding a MISR for scan shift detection appeared to be obtained by using random pattern simulation. Thus, random pattern simulation can be run as a quick test before generating the real intermediate patterns.
From the results obtained in [12], we noticed that Des56 and Fm_rec were two extremes with respect to the percentages of flip-flops that were required to be shadowed in the MISR to obtain maximum cell-aware fault coverage [12]. Similar observations could be made for the data involving extra detections of stuck-at faults, as shown in Figure 5.3 (b) and Figure 5.3 (c) in Chapter 5. In this chapter, we explore how fault cone analysis can reveal characteristics of the circuits regarding the relationship between flip-flops and fault detection. We will show that these circuit characteristics have a large impact on the amount of flip-flop overhead required to achieve the high coverage with a MISR that captures test data during scan shift.

Figure 6.1 shows the fault cone analysis results. For each circuit, faults were reported to the D pin of each flip-flop by a commercial software tool. As described previously, among the six types of fault classes reported, only UC faults are likely to be detectable with pattern stimulus (although the detection is not guaranteed because fault detection still requires simultaneous excitation and observation by the patterns, and this is not guaranteed by the commercial tool when it reports fault cone data). Therefore, when analyzing fault cone results, only UC faults were counted. Intuitively, when more faults are identified as UC faults for a flip-flop, more faults have that flip-flop in their downstream cones of influence, and thus more fault detections are likely possible at that flip-flop. In contrast, a flip-flop with a small number of UC faults is likely to have few faults that can propagate to it.

In Figure 6.1, the x-axis is the flip-flop percentage where the flip-flops are sorted in the order of the percentage of UC faults reported to the flip-flop from maximum to minimum. Dots were plotted based on the UC fault data for each flip-flop.

The values plotted on the y-axis show the percentage of all faults that are in the fault cone of each flip-flop. In other words, each dot corresponds to the number of faults reported as UC for a flip-flop divided by the number of faults in the full UC fault list. The larger
Figure 6.1: Per Flip-Flop UC Fault Coverage with 1%, 2%, 4%, 16% and 32% Flip-flop Vertical Lines

In Figure 6.1, we can see that, for Fm_rec, only a small percentage of flip-flops contribute to most of the fault detection. Each flip-flop in the top 3.19% of the flip-flops has a large number of faults in its fault cone. That could explain why Fm_rec can get a large number of fault detections during scan shift with only a very small percentage of flip-flops shadowed in the MISR. In contrast, the highest fault cone percentages for Des56 are around 5%. The fault cones appear to be more evenly distributed among many flip-flops. This helps explain why Des56 requires a MISR with many shadow flops included to get good fault coverage across the circuit during scan shift.

This type of analysis can be very useful to a designer trying to decide whether to insert the proposed MISR structure into a design. Specifically, if a few flip-flops have fault cones that together cover most of the faults, then the proposed approach may be a relatively low-
cost way to improve coverage, reduce test time, and increase quality. On the other hand, if the faults are distributed relatively evenly into fault cones with limited overlap such that most of the flip-flops would need to be shadowed, a large MISR will likely be necessary to get high coverage.
Chapter 7
Transition Fault Detection by Enhancing Scan Chain with MISR and Backup Flip-Flops

Parts of this chapter were first presented in [2].

The scan chain enhancements previously explored in this dissertation only considered static faults. In this chapter, a new DFT circuit design is introduced to allow fortuitous transition fault detections when only stuck-at test sets are applied.

7.1 Transition Fault Detection During Scan shift

The majority of the test time in scan-based circuits is devoted to scan shift instead of capture—especially when chains are long (e.g. [12], Chapter 4.) As a result, only a very small percentage of all testing clock cycles are used to capture defective behavior in such designs. To allow these shift cycles to be used for defect detection, modifications to the traditional scan chain have been proposed. In particular, a scan chain enhancing method that used a MISR structure to capture test data during scan shift achieved high cell-aware coverage for static defects in [12]. Similarly, high $n$-detect coverage of stuck-at faults was explored in Chapter 4.

These approaches were highly successful because they allow the number of “effective” patterns applied to increase by a factor approximately equal to the number of scan shift cycles if test data is captured on every shift cycle. Thus, even a relatively short pattern set can be supplemented by pseudo-random “intermediate” patterns that are orders of magnitude larger than the original test set, and each of these intermediate patterns can potentially detect additional faults and defects.

Due to the success of these previously studied approaches, it is reasonable to consider applying “intermediate” scan shift patterns to transition fault detection as well. However, an important complication arises due to the fact that transition faults are modeling extra delays. Technically, transition faults are assumed to model large lumped delay defects, and
as a result the path taken through the circuit (and the slack of the associated path) is assumed to be inconsequential. However, in reality, the slack of the path taken often does matter when detecting extra delays in real circuits, and thus ideally delay defects should be detected in the functional flip-flops—even during scan shift—instead of shadow flip-flops. At the same time, it is necessary to allow the desired ATPG pattern to be present in the chain at the end of scan shift and to shift out the actual values captured by the original ATPG-generated test pattern.

Because [12] and Chapter 4 aimed to detect static defects only, capturing test data in shadow flip-flops collected into a MISR, instead of in the original functional flip-flops in the scan chain, did not affect the detection of those defects. Because this is no longer true in the case of added delays, a modification to the intermediate pattern-based DFT structures proposed in prior work is needed for the approach to be extended to delay faults.

Figure 7.1: An Example of Transition fault

Figure 7.1 shows one example of a transition fault (in this case, a slow-to-rise transition fault). To detect this fault, a 0-to-1 transition is launched on the faulty input pin of the AND gate, while the other pin must be set to 1 in the second clock cycle to allow point $P$ to be observed at the output. Thus, in the second clock cycle (observation pattern), the detection requirements for point $P$ slow-to-rise are identical to the detection conditions for a stuck-at 0 fault at point $P$. The difference arises in the first clock cycle (pre-conditioning
pattern) where \( P \) must be set to the appropriate value to launch the transition. Table 7.1 summarizes the corresponding transitions and stuck-at fault detections required for the two types of transition faults.

Table 7.1: Transition and stuck-at fault detection required for the two types of transition faults

<table>
<thead>
<tr>
<th>Transition Fault Type</th>
<th>Required Transition Type</th>
<th>Corresponding Stuck-at Fault Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slow to Rise</td>
<td>0 to 1</td>
<td>Stuck-at 0</td>
</tr>
<tr>
<td>Slow to Fall</td>
<td>1 to 0</td>
<td>Stuck-at 1</td>
</tr>
</tbody>
</table>

There are two main ways to apply a transition test pattern set in a scan chain environment. The first is Launch-on Capture (LoC). The other one is Launch-on Shift (LoS). As shown in Figure 7.2, for Launch-on Capture, the scan enable signal goes low for rising clock edges. The first rising clock edge finishes shifting the pre-conditioning pattern into the chain. The circuit response values captured in the flip-flops after the application of the shifted in pre-conditioning pattern form the second (observation) pattern. The differences between the pre-conditioning and observation patterns cause transitions to be launched into the circuit. Then, in the next clock cycle, results must be captured at-speed to detect transition faults.

The Launch-on-Shift (LoS) approach uses the last shift of the scan chain to launch transitions. The pattern in the chain before the last shift is the pre-conditioning pattern, while the last shift generates the observation pattern. In this case, the Scan Enable signal must go low within an at-speed clock period so that the results of the test can be captured at-speed.

When intermediate shift patterns are used to detect transition faults during scan shift, the approach is similar to the standard LoS method because the shifting of the chain is used to launch transitions into the circuit. It is the capturing of the results that is difficult and that requires updated circuitry.
7.2 Circuitry Structure for Transition Fault Fortuitous Detection with Intermediate Patterns

As already noted, when intermediate shift patterns are used to launch transitions, capturing test data in the original flip-flops (instead of only in a MISR) is preferable so that the delay of the detection path during test matches the delay of the path used in functional mode. However, capturing data in the original flip-flops during scan shift will overwrite the previous test results being shifted out or the new pattern being shifted in—making it impossible to apply the original ATPG test set. To address this issue, a new DFT structure is proposed in Figure 7.3. Figure 7.3 shows DFT circuitry designed for transition fault detection with intermediate shift patterns. In this figure, FF1 to FF5, along with their preceding multiplexers, are shown in black and form five MUX-D scan flip-flops. These scan flip-flops form the original scan chain.

In this figure, FF4 is selected to capture test data during scan shift while the other flip-flops will only capture test data once the full ATPG-generated pattern has been shifted in. To accomplish this additional capturing of test data in the original scan flip-flop while preserving the ability to successfully shift in each ATPG pattern, the circuit elements shown in red are added to the original chain. In particular, a Backup FF is added in parallel with FF4 to backup the values that are supposed to be shifted into FF4. This FF and the other added elements in red work with the rest of the DFT logic as described below.
7.2.1 Description of the Added Circuitry

When we wish to apply a launch-on shift (LoS) transition test using an intermediate pattern during scan shift, we need to allow FF4 to capture data from the circuit’s logic while the other flip-flops on the scan chain capture data from the previous flip-flop in the chain. At the same time, the Backup FF should capture the data that would have been shifted into FF4.

Thus, three control signals from another logic block, Signal Control, are added (Figure 7.3). Note that only one Signal Control module is needed for all of the scan chain flip-flops that are selected to capture test data simultaneously with intermediate shift patterns—even when those flip-flops are in different scan chains Capture Control is used to determine whether SE will enable the shifting of data into FF4. When Capture Control is equal to 1, the SE signal will determine whether FF4 captures data from the circuit’s functional logic or from the previous flip-flop in the chain. When Capture Control is equal to 0, the capture of test data into FF4 from the circuit’s logic with an intermediate shift pattern is enabled even as the other flip-flops in the chain continue to shift normally.

After test data is captured into FF4 with an intermediate shift pattern, two things must happen. 1) The value in FF4 must be fed into a MISR so that a single signature can be used to identify whether any of the values captured by the intermediate shift patterns were incorrect. 2) The value which would have been shifted into FF4 under normal shifting conditions must be shifted into FF5 instead of the value that was captured in FF4 from the circuit’s logic.

Thus, to shift the correct value into FF5, a multiplexer after FF4 is used to control which value is shifted into FF5. The select input of this multiplexer is set by Shift Control from the Signal Control logic block. Note that the MISR Enable signal is generated by the Signal Control module as well to ensure that the MISR value will be updated only when a test result from an intermediate shift pattern is in the corresponding scan flip-flop (here FF4)—as opposed to all clock cycles.
7.2.2 Example of Operation with the Proposed Enhancement

Figure 7.5 shows an example of the proposed structure in operation. The example begins as data is captured in all of the scan flip-flops when a regular ATPG test pattern is applied. In this example, an intermediate shift pattern will be used to detect defects, and \( FF_4 \) will capture test data after two bits of the next ATPG pattern are shifted in.

1. Cycle of normal capture: The \( SE \) signal is set to 0, and all flip-flops on the scan chain capture values from the circuit’s logic. These values are specified as \( C_1 \) to \( C_5 \), depending on the flip-flop in which the capture occurs.

2. First cycle after normal capture: The first bit of the next ATPG pattern is shifted in with the \( SE \) signal set to 1. The same value \( C_3 \) (from \( FF_3 \)) is shifted into \( FF_4 \) and into the \textit{Backup FF}. \textit{Capture Control} is set to 1; \textit{Shift Control} and \textit{MISR Enable} are set to 0.

3. Second cycle after normal capture: The second bit of the next ATPG pattern is shifted into the chain. The value \( C_2 \) is shifted into \( FF_4 \) and the \textit{Backup FF}. The values of
Figure 7.4: Transition Fault Scan Chain Enhancing DFT Structure Work Flow (1. Capture Cycle, 2. Cycle 1 After Capture)
Figure 7.5: Transition Fault Scan Chain Enhancing DFT Structure Work Flow (3. Cycle 2 After Capture, 4. Cycle 3 After Capture)
Figure 7.5: Transition Fault Scan Chain Enhancing DFT Structure Work Flow (5. Cycle 4 After Capture, 6. Cycle 5 After Capture)
SE, Capture Control, Shift Control and MISR Enable remain the same as in last clock cycle. This cycle launches the transitions that will be used for detecting transition faults with the intermediate shift pattern.

4. Third cycle after normal capture: The third bit of the next ATPG pattern is shifted into the scan chain. The previously captured value $C_1$ is saved in the Backup FF. Because we want to capture test data ($M_1$) from the circuit logic in response to the transitions launched on the last shift cycle in $FF_4$, the value of Capture Control is set to 0. To match the delay of the circuit in normal operation, the time between the start of the “2nd cycle after capture” and the start of the “3rd cycle after capture” should be equal to one clock period. Thus, the value of Capture Control must transition from 1 to 0 in less than 1 clock. Shift Control and MISR Enable remain unchanged.

5. Fourth cycle after normal capture: The fourth bit of the next ATPG pattern is shifted into the scan chain. The MISR is enabled by setting MISR Enable to 1 so that the value captured in $FF_4$ can be used to update the signature in the MISR. Shift Control is set to 1, so that the value $C_1$ from the Backup FF is shifted into $FF_5$. Capture Control is set to 1 so that both $FF_4$ and the Backup FF will receive the first bit of the new ATPG test pattern: $S_1$.

6. Fifth cycle after normal capture: The last bit of the next ATPG pattern is shifted in. The value shifted into $FF_5$ is received from $FF_4$ with Shift Control as 0. The signature in the MISR does not update as MISR Enable is disabled (set to 0). Both $FF_4$ and the Backup FF capture the value $S_2$ of the ATPG test pattern. At this point, the entire ATPG pattern has been shifted in, and the process can begin again.

7.3 Experimental Procedure of Scan Chain Enhancing Transition Fault Detection

To determine the ability of the proposed approach to fortuitously detect transition faults with stuck-at test patterns, the following procedure was followed, as shown in Figure 7.6.

1. **Stuck-at Fault ATPG Pattern Generation**: Stuck-at ATPG patterns are generated for each circuit. In each case, an on-chip decompressor was used to feed the
circuit’s scan chains. The values loaded in the scan chains, primary input (PI) values, primary output (PO) values, and good circuit simulation capture values for each ATPG pattern are obtained.

2. **Intermediate Pattern Generation**: Stuck-at intermediate patterns are generated for each shift cycle. Each intermediate pattern is a combination of the ATPG values being shifted in and the good circuit simulation values being shifted out from the previous pattern.
3. **Stuck-at ATPG Pattern Transition Fault Detection**: The last shift of a regular stuck-at ATPG pattern into the scan chain can be used to detect transition faults without requiring the backing up of any flip-flop data or the enabling of the added MISR. In particular, because the last shift of the stuck-at pattern naturally launches transitions into the circuit, transition faults can be detected along with stuck-at faults by capturing data at the circuit’s outputs and scan flip-flops as is normally done for a stuck-at test—provided that the capturing can be done at-speed.

To determine the corresponding transition fault coverage, transition information from the last shift and stuck-at fault detection information for each stuck-at ATPG pattern are reported for every fault site using a commercial tool. If a transition fault was excited with the correct transition, and if the corresponding stuck-at fault was detected with the observation pattern, the transition fault is considered detected by the last shift of the ATPG pattern.

Using the final stuck-at test pattern for transition detection is advantageous because stuck-at fault coverage is usually very high, and even stuck-at faults that are detected only once can potentially contribute to fortuitous transition fault coverage. However, some transition faults will still remain undetected; it is these undetected faults that we will target for detection by intermediate patterns during scan shift. Flip-flops will be selected for the additional Back-up and MISR circuitry to improve the detection of these missed faults.

4. **Selecting Shifting Clock Cycles to Perform Extra Captures**: Because we are capturing test data in the original flip-flops when an intermediate shift pattern is applied (e.g. the value \( M1 \) is captured in the regular flip-flop in Figure 7.5), we must determine those values and include them in our intermediate pattern fault coverage analysis if at-speed capture for intermediate patterns is performed on every shifting clock cycle. This will make our analysis more complex. To avoid this complexity, in this dissertation, we restrict intermediate pattern fault coverage analysis to only those clock cycles in which values such as \( M1 \) do not appear in the scan chain.

Figure 7.7 shows one example of intermediate pattern sampling to avoid this complex-
ity. Each numbered square corresponds to a shift clock cycle for a chain of length 10. In this example, ten clock cycles are needed to fully shift a stuck-at ATPG test pattern into the chain.

Figure 7.7: Intermediate Pattern Sampling Example (Chain Length 10)

In our experiments, the first intermediate pattern pair used for fortuitous detection of transition faults corresponds to intermediate patterns 1 and 2 (i.e. shifting clock cycles 1 and 2). The transitions in the circuit logic are launched by the second shifting clock cycle, and the intermediate pattern in the chain after the second shift is the pattern that is used to observe the stuck-at faults that correspond to each transition fault. Then, the “extra” capture of the test result occurs with shift cycle 3—where the selected flip-flops capture the circuit’s test results, and the backup flip-flops preserve the values that would have been shifted into those selected flip-flops under normal conditions. Thus, the time between the launching of transitions with shift cycle two and the capturing of test results with shift cycle three must be “at speed.”

Because the data captured in the selected flip-flops on shift cycle 3 correspond to data generated by the circuit logic, they are not available a priori from simple analysis of the original stuck-at ATPG patterns in the test set. Instead, good circuit simulation must be done to obtain those values. While that data must be obtained eventually to
determine the appropriate MISR signature, if we have not yet finalized which flip-flops will be selected for the extra captures, then there are many possible versions of shift cycle 3 that could be used and analyzed for estimating transition fault coverage. To avoid this complexity, the next intermediate pattern pair used for fortuitous detection of transition faults corresponds to shift cycles 4 and 5.

If we continue in this way, the last intermediate pattern pair would correspond to shift cycles 7 and 8, which would launch the next set of transitions. Intermediate pattern 8 would correspond to the observation pattern, and clock cycle 9 would correspond to the next capture of test results and backing up of test data. Unfortunately, if we do this, then we cannot use intermediate pattern pairs 9 and 10 to detect transition faults for the “Stuck-at ATPG Pattern Detection” from Step (3) because some of the values in the chain on cycle 9 come from the circuit logic instead of only the shifted data.

As a result, because “Stuck-at ATPG Pattern Detection” from Step (3) is usually more valuable for fortuitously detecting transition faults, in our experiments we do not use intermediate patterns 7 and 8 for fortuitous transition fault detection. (Ending on a true stuck-at test pattern is more beneficial because the ATPG tool has tried to guarantee at least minimal detections of even the most difficult-to-detect stuck-at faults with the final stuck-at fault test patterns.)

5. **Flip-Flop Selection for Backup and Extra Capture**: To determine which flip-flops should be selected for backup and extra captures, stuck-at fault coverage data from the observation patterns (e.g. 2, 5, and 10 in Figure 7.7) are used. For faults detected by intermediate patterns, flip-flops will be selected with the same greedy algorithm used before, based on stuck-at detection to reduce runtime. Flip-flops are sorted in the order of detected stuck-at faults from highest to lowest, and the top flip-flop is selected. Stuck-at faults detected by that flip-flop are removed from consideration, and the process repeats. The same sorting and selecting steps are repeated until all faults have been detected or all flip-flops have been selected. At the end, the selected flip-flops form a flip-flop list that could potentially detect detectable transition faults. Because transition launching conditions are not used in the analysis, the selection pro-
procedure is slightly more general than would otherwise be the case. (Future work will look at the differences in flip-flop selections that may occur with other variations of this procedure.)

6. **Transition Fault Detection with Selected FFs**: Transition fault simulation is performed with the selected flip-flops to obtain an accurate value for transition fault coverage. This step also allows for analyzing fault-coverage at different hardware overhead allowances when fewer flip-flops are selected.

7. **Top-off Transition Fault Pattern Generation**: While stuck-at ATPG patterns and intermediate patterns detected a high percentage of transition faults, some transition faults can still remain undetected. To achieve maximum transition fault coverage, launch-on shift ATPG patterns are added as top-off patterns. In fact, we will show that the fault coverage with top-off patterns may be even slightly higher than that of a commercially generated ATPG pattern set due to some faults that the commercial tool considers too costly to detect being fortuitously detected with our approach.

In addition, because the number of of top-off patterns for maximum transition fault coverage can be high for some circuits (especially when each top-off pattern can only detect a small percentage of the undetected transition faults), we also consider test sets that only add as many top-off patterns as are necessary to ensure that the final fault coverage is not less than a commercial tool’s launch-on shift transition fault coverage.

7.4 Experimental Results of Transition Fault Detection with Intermediate Patterns and Flip-Flops Shadowed

Using these steps we performed experiments on five circuits obtained from Opencores.org [64]. Table 7.2 shows the circuit and test set characteristics (column 2 to column 4) as well as the intermediate patterns and sampled intermediate patterns generated by our method (columns 5 and 6). Table 7.3 indicates the number of launch-on shift patterns and corresponding fault coverages obtained with commercial software tool.

Figure 7.8 shows fortuitous transition fault coverage with stuck-at ATPG patterns and intermediate patterns. For each circuit, the first bar (green) shows the launch-on shift
Table 7.2: Characteristics of Circuits

<table>
<thead>
<tr>
<th>Circuit</th>
<th># of Flip-Flops</th>
<th># of Transition Faults</th>
<th># of SA ATPG Patterns</th>
<th># of Intermediate Patterns</th>
<th># of Sampled Intermediate Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quadratic</td>
<td>120</td>
<td>8166</td>
<td>36</td>
<td>1044</td>
<td>324</td>
</tr>
<tr>
<td>Des56</td>
<td>193</td>
<td>13788</td>
<td>119</td>
<td>2856</td>
<td>833</td>
</tr>
<tr>
<td>Fm_rec</td>
<td>501</td>
<td>19888</td>
<td>406</td>
<td>10962</td>
<td>3248</td>
</tr>
<tr>
<td>Colorconv</td>
<td>584</td>
<td>38518</td>
<td>98</td>
<td>3136</td>
<td>980</td>
</tr>
<tr>
<td>Fpu</td>
<td>5231</td>
<td>297358</td>
<td>538</td>
<td>17216</td>
<td>5380</td>
</tr>
</tbody>
</table>

Table 7.3: LoS ATPG Pattern Set Characteristics

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Quadratic</th>
<th>Des56</th>
<th>Fm_rec</th>
<th>Colorconv</th>
<th>Fpu</th>
</tr>
</thead>
<tbody>
<tr>
<td># of LoS Transition Fault ATPG Patterns</td>
<td>64</td>
<td>233</td>
<td>681</td>
<td>148</td>
<td>628</td>
</tr>
<tr>
<td>LoS Transition Fault Coverage (%)</td>
<td>79.08</td>
<td>94.72</td>
<td>89.69</td>
<td>74.31</td>
<td>87.07</td>
</tr>
</tbody>
</table>

transition fault coverage obtained by a commercial ATPG tool for a dedicated launch-on-shift test set. The second bar (yellow) shows the fortuitous fault coverage achieved with the stuck-at ATPG and intermediate patterns when all flip-flops can capture test data on intermediate patterns. Most of the transition faults are covered. Finally the third bar (blue) shows the fault coverage when the flip-flop selection algorithm is used to shadow only a subset of flip-flops into the MISR. Even though stuck-at fault detection was used for flip-flop selection, the transition coverage remains almost the same.

Figure 7.9 shows the percent of flip-flops shadowed in the MISR to maximize fortuitous detection of transition faults by stuck-at and intermediate patterns. In some cases, these numbers could be further reduced if faults that could be detected by a chain test were not included.

We also ran experiments to see how putting an upper bound on the percentage of flip-flops selected impacted fault-coverage. Using a greedy algorithm (as in Step 5), we picked flip-flops up to a certain budget by prioritizing the flip-flops that detect the most faults first. Figure 7.10 shows the corresponding transition fault coverage. From the figure, we
can see that even with a low budget of 1%, significant fault-coverage is possible. For circuit Quadratic, we can get the maximum fault coverage with only 19% of the flip-flops. Thus, Quadratic was excluded from the 20% and 40% experiments in Figure 7.10. Similarly circuit Fm_rec also gets to maximum fault coverage with just 23% of the flip-flops selected. The column for that circuit is excluded from the 40% experiment. Note that all the flip-flop percentages shown in this figure are generally far lower than the flip-flop percentages required to detect all transition faults that could be detected by the intermediate patterns (Figure 7.9). The hardware overhead reduction is up to 97.45%. The flip-flop percentages used in the experiments here range from 1% to 40%, while the required FF percentages range from 19.17% to 98.45%.

Even higher fault coverage values may be obtained with less overhead if the possible detection of faults during the chain test were considered (11% to 17% of faults may be detectable by chain test for the circuits studied). The last two bars of each circuit cluster in
Figure 7.11 shows transition fault coverage with stuck-at ATPG pattern chain test detections. For all five circuits, fault coverages are very close to commercial tool launch-on shift fault coverage, with stuck-at ATPG patterns, intermediate patterns, selected flip-flops and stuck-at pattern chain test.

Next, with stuck-at ATPG pattern chain test detections added, flip-flops were re-selected. The percentages of flip-flops required to detect as many transition faults as possible with intermediate patterns remain the same. Figure 7.12 shows transition fault coverage with various flip-flop overheads. From the results, for circuit \textit{Quadratic}, even with only 1\% flip-flop overhead, transition fault coverage is very close to commercial launch-off shift fault coverage. For other circuits, high fault coverage can also be obtained with a certain percent of flip-flop overhead.

Launch-on shift ATPG top-off patterns were added to detect any transition faults that still remain undetected. The last bar in Figure 7.13 shows the final fault coverage with commercial tool launch-on shift top-off patterns. From the figure, higher transition fault coverage can be seen compared with even commercial tool launch-on shift transition fault
Figure 7.4 shows the number of top-off patterns required to 1) achieve the maximum possible fault coverage with our method and 2) to obtain a fault coverage at least as high as that achieved by the commercially generated LoS test set. Data were collected for different flip-flop overheads.

Take Quadratic for example. The left part of Figure 7.14 (a) shows the number of patterns applied as a percent of the commercial ATPG launch-on shift pattern count required for maximum fault coverage with various flip-flop overheads. In this case, all faults that were originally detected by the commercial tool launch-on shift pattern set, the last shift of the stuck-at patterns and/or stuck-at intermediate patterns were targeted. Generally, the x-axis lists percentages of flip-flop overhead, i.e. 1%, 2%, 5%, 10%, 20% and 40% top selected flip-flops (until the percentage of required flip-flops in Figure 7.9 for each circuit) were included. For Quadratic, the flip-flop percentages on the x-axis are 1%, 2%, 5%, 10%
and the maximum percent of flip-flops required for the previously identified intermediate pattern detectable transition faults. The left y-axis shows the number of patterns to be applied as a fraction of the commercially generated LoS test set pattern count. Each bar corresponds to the sum of the number of stuck-at patterns and the number of commercial tool launch-on shift patterns needed for each scenario divided by the number of patterns in the commercial tool launch-on shift test set. The blue part of each bar corresponds to the number of stuck-at ATPG patterns as a fraction of the number of LoS patterns in the LoS pattern set. The top-off parts (orange bars on the top of blue bars) are the percentages of launch-on shift top-off patterns that are added to achieve the maximum transition fault coverage with each flip-flop overhead percentage. The right y-axis, which corresponds to the grey line, shows the overall transition fault coverage achieved with the stuck-at ATPG patterns and commercial tool top-off launch-on shift patterns. Given that all of the launch-on shift detectable transition faults are detected and some other faults are fortuitously detected
with the stuck-at patterns shifted at speed (ATPG patterns or intermediate patterns), the final fault coverage with top-off launch-on shift patterns can be higher than the commercial tool’s launch-on shift transition fault coverage. As can be seen in Table 7.3, this can be true even with a flip-flop overhead lower than the maximum overhead in Figure 7.9.

However, the high fault coverage can require a considerable percentage of top-off launch-on shift patterns. Thus, the percent of top-off launch-on shift patterns required to just achieve the commercial tool launch-on shift transition fault simulation coverage was also studied. The the bars in right hand side part of Figure 7.14 indicate the smallest percentage of top-off launch-on shift patterns required to achieve (no less) transition fault coverage than that obtained by the commercial tool’s launch-on shift ATPG test set. The grey line shows the actual fault coverage (corresponding to the right y-axis) for each case with various percentages of flip-flop overhead. One thing need to notice is that, the numbers of transition faults detected by each top-off launch-on shift pattern added may not be able to attain a final coverage identical to the commercial tool’s launch-on shift fault coverage. The extra
fault coverage may be higher for some scenarios and a little bit lower for other scenarios. This means the fault coverage in the right hand side may not be monotonically increasing. The single example occurs with Fm_rec, where the LoS fault coverage is 89.69%. For the cases with 2% and 5% flip-flop overhead, the first fault coverage that is no less than the target fault coverage corresponds to 89.70% and 89.71%, respectively. However, for the 10% flip-flop case, it is possible to achieve exactly 89.69% coverage. So the fault coverage shown decreases from the 5% flip-flop case to 10% case in this graph.

7.5 Conclusion of Transition Fault Detection by Enhancing Scan Chain with MISR and Backup Flip-Flops

In this chapter, the experimental results above show that significant transition fault coverage can be achieved using a combination of ATPG stuck-at patterns shifted at-speed,
Figure 7.14: Top-off Pattern for Commercial Tool Launch-off Shift Transition Fault Coverage with Selected Flip-Flops (Quadratic, Des56)
Figure 7.14: Top-off Pattern for Commercial Tool Launch-off Shift Transition Fault Coverage with Selected Flip-Flops (Fm_rec, Colorconv)
Figure 7.14: Top-off Pattern for Commercial Tool Launch-off Shift Transition Fault Coverage with Selected Flip-Flops (Fpu)
intermediate patterns, and selective flip-flop shadowing in the MISR, even with a small percent of flip-flop overhead used. With stuck-at ATPG pattern chain test detection added, higher transition fault coverage numbers were obtained. Furthermore, to achieve maximum coverage, launch-on shift ATPG top-off patterns were added. However, considerable top-off launch-on shift patterns were required. To fairly compare the pattern counts, the percent of top-off launch-on shift patterns required for commercial tool launch-on shift fault coverage were also indicated. With that lower fault coverage targeted, the percent of required top-off launch-on shift patterns was reduced.

Similar to [12], and in Chapter 4 and 5 in this dissertation, the percentages of flip-flops required for intermediate pattern detectable transition faults are higher for circuits Des56 and Fpu than the other circuits. Regarding the transition fault coverage with the selected flip-flops, it is also true that Fm_rec can easily obtain high transition fault coverage with a small percent of flip-flop overhead (Figure 7.12).

The same fault cone analysis in Chapter 6 can explain these phenomena. Given that detection of the corresponding stuck-at fault is one of the two conditions required for transition fault detection, the fault cone analysis in Chapter 6 still applies. Specifically, that analysis indicated whether each stuck-at fault could potentially propagate to each flip-flop in the circuit. When the flip-flops that have the highest percentage of fault cones reported can only detect a small percent of all faults, then few stuck-at faults are able to propagate to each flip-flop. Then many flip-flops will be needed to cover most of the stuck-at (or transition) faults in the circuit.

For percent of top-off launch-on shift patterns required, overall, the percentages reduce when the percentages of flip-flop overhead increases for both maximum fault coverage and commercial tool launch-on shift fault coverage.

Generally, for transition fault detections with this proposed scan chain enhanced DFT circuitry and a certain fault coverage goal, it is a trade-off between flip-flop overhead and the number of top-off patterns needed. Circuit characteristics can be explored by using fault cone analysis, as illustrated in Chapter 6. Circuits that show fault cone characteristics similar to Fm_rec may be especially well-matched to this scan chain enhancing DFT approach, while pattern sets can also be reduced for the other circuits with a certain flip-flop overhead.
Moreover, given the difficulty of transition fault detection, stuck-at ATPG patterns may not be the ideal pattern set to use. Instead, if a transition fault ATPG pattern set is used, transition faults can be deterministically detected with the ATPG patterns. With the addition of this scan chain enhancing DFT circuitry, the intermediate patterns (generated from transition fault ATPG patterns) can still detect transition faults fortuitously. As a result, pattern count reduction could likely be seen. Smaller percentages of required flip-flops are also possible while still achieving either maximum transition fault coverage or matching the ATPG test set’s launch-on shift transition fault coverage as well. The top-off patterns would not be necessary for undetected transition faults (transition faults not detected by patterns applied with this procedure but detectable with transition fault ATPG patterns), because the transition fault ATPG patterns would have been able to detect all of them.
8.1 Dissertation Conclusion

A previously proposed DFT enhancement that consists of shadow flops collected into a MISR has been further evaluated. Although the design was first intended to fortuitously detect cell-aware faults during scan shift, this dissertation has shown that it is also useful for stuck-at fault \( n \)-detect. Specifically, it helps a considerable percentage of the otherwise low-detected stuck-at faults gain extra detections. For some circuits, this can be accomplished even with only 1\% of the flip-flops shadowed in the MISR. Our analysis has also shown that even better \( n \)-detect results can often be achieved if the flip-flops selected for inclusion in the MISR were chosen based on their contribution to the maximum stuck-at fault coverage. These extra detections of stuck-at faults are essential due to the fact that these low-detected faults can easily contribute to test escapes. Improving the detection counts of low detected faults increases the opportunity for fortuitous detection of any untargeted faults or unmodeled defects associated with these stuck-at faults that might otherwise remain undetected.

Given that the fault simulation process can be time-consuming when used for flip-flop selection, and because the intermediate pattern generation procedure requires that an ATPG test set already be available, using fault cone analysis and random patterns for flip-flop selection was considered. With these means, a baseline of the percentage of stuck-at faults that obtained extra detections could be reported quickly. DFT engineers could use this estimation of the effect of applying this structure to decide whether a circuit would benefit sufficiently from this enhancement. Even better performance could be expected when the real intermediate patterns are generated and used for flip-flop selection while targeting maximum detections of stuck-at faults.

Next, an alternative MISR-based approach targeted toward delay defects was proposed.
It was shown that this design is able to detect a large percentage of the transition faults when only a stuck-at fault ATPG pattern set is applied, provided that shifting the chain could be done “at speed” at the appropriate times. When stuck-at ATPG chain test detections and transition fault launch-on shift top-off patterns were added, a slightly higher fault coverage is achieved compared with the commercial tool’s reported launch-on shift transition fault coverage.

Circuit characteristics for the five circuits were analyzed using fault cone results reported by a commercial tool. These results provide insight to help explain the variations in overhead requirements, stuck-at fault n-detect results, and transition fault detection results for the five circuits studied in this dissertation. This type of analysis can be used by a DFT engineer to identify which circuits are most amenable to the proposed approaches at a reasonable cost.

In conclusion, better test quality can be expected with smaller test sets and test data volume with these two scan chain enhancements, especially for some of the circuit/pattern characteristics analyzed in this dissertation.

8.2 Future Work

In the future, additional types of faults will be studied for detection with the proposed DFT enhancements and other similar DFT structures. For example, even though a high percentage of transition faults can be detected with the second scan chain enhancement described in this dissertation, the detection of cell-aware delay faults has not been explored. Applying the proposed design to cell-aware delay faults is promising and needed because the test sets for these faults can be prohibitively large. Thus, future research will focus on the detection of cell-aware delay faults with the proposed DFT circuitry.

In addition, new flip-flop selection algorithms that are different from the greedy algorithm will be studied to try to further reduce the hardware overhead required by the proposed designs. It is well known that even though a greedy algorithm may produce the “best” solution at each step, it may not lead to the best solution overall.
BIBLIOGRAPHY


77


