1-1-2012

Experimental Study Of Fault Cones And Fault Aliasing

Vedanth Bilagi
Portland State University

Experimental Study Of Fault Cones And Fault Aliasing

by

Vedanth Bilagi

A thesis submitted in partial fulfillment of the
requirements for the degree of

Master of Science
in
Electrical and Computer Engineering

Thesis Committee:

W. Robert Daasch, Chair
C. Glenn Shirley
Branimir Pejcinovic

Portland State University
2012.

Abstract

The test of digital integrated circuits compares the test pattern results for the device under test (DUT) to the expected test pattern results of a standard reference. The standard response is typically obtained from simulations. The test pattern and response are created and evaluated assuming ideal test conditions. The standard response is normally stored within automated test equipment (ATE). However the use of ATE is the major contributor to the test cost. This thesis explores an alternative strategy to the standard response.

As an alternative to the stored standard response, the response is estimated by fault tolerant technique. The purpose of the fault tolerant technique is to eliminate the need of standard response and enable online/real-time testing. Fault tolerant techniques use redundancy and majority voting to estimate the standard response.

Redundancy in the circuit leads to fault aliasing. Fault aliasing misleads the majority voter in estimating the standard response. The statistics and phenomenon of aliasing are analyzed for benchmark circuits. The impact of fault aliasing on test with respect to coverage, test escape and over-kill is analyzed. The results show that aliasing can be detected with additional test vectors and get 100% fault coverage.

Acknowledgements

I would like to take this opportunity to express my gratitude to many wonderful people who have encouraged me during my graduate study at Portland State University. First of all, this research would not have be successful without the help and support of my academic advisor Dr. Robert Daasch. His outstanding knowledge and research involved in digital design and test helped me not only achieve the goal but also improve my critical thinking and problem solving skills. Also, I am very thankful to Glenn Shirley for brainstorming and making the ideas more crisp and clear.

Also, I would like to express my gratitude to Professor Branimir Pejcinovic member of my M.S. thesis defense committee. I would also like to thank the staff in the Electrical and Computer Engineering Department for their assistance throughout my graduate study. The people in the ECE department made me a much better engineer.

Thanks to the ICDT (Integrated Circuit Design and Test) lab colleagues Kapil, Vivek, Rohan, Chaitrali and Satoshi for the support and encouragements. Thanks to Andrew for discussing and sharing his work on fault simulation of benchmark circuit.

Finally, I personally would like to thank my parents and my sisters for their constant support during my school life in America. I would not be what I am today without the support and intelligence of the wonderful people around me.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Digital circuit testing involves applying test patterns and observing the circuits responses to the applied patterns. The tester compares the observed response to a test pattern with the reference response and declares a chip defective upon mismatch. Test engineers usually obtain the expected response through fault-free simulation of the circuit for the corresponding test pattern [1]. The test pattern and response are created and evaluated assuming ideal test conditions. The expected response is normally stored within automated test equipment (ATE), avoiding the need to emulate the reference unit. However, the use of ATE is the major contributor to the test cost. This thesis explores alternative strategies to the comparison with standard response.

For reliability reasons it is required to test the circuit often, especially where a failure cannot be afforded. For example, the ICs that go into spacecrafts, satellites and other sophisticated and high stake machines. The need for a reference response unit mitigates the possibility of online/real-time testing.

1.1 Motivation

The need for reference response [1] increases the cost and complexity of the test and has become the bottle-neck at reducing the test cost. An alternative technique for IC testing that substitutes the stored response is a fairly viable option. The elimination of the need of stored response opens up other possible strategies for online/real-time testing.

Today's integrated circuits are hierarchical and modular. In ICs, the same circuit modules are used multiple times as part of different operation or replicating the same operation multiple times. For example, a 32-bit adder is used in arithmetical logical unit, program counter and also in table indices. IC's may have N copies of the same 32-bit adder or N 32-bit adders, slightly different in architecture but functionally identical. In either case there are N copies of functionally identical 32-bit adder. The redundancy in the IC's gives an opportunity to explore the ways to use the available redundancy in the circuit in order to eliminate the need for reference response in IC test.

The real-time testing of ICs gives way for design architectures where redundant circuits will be used only when required. The confined use of redundant circuits drastically reduces the power consumption, which is a major concern in most fault tolerant systems [2]. If a circuit fails at some point of time, the IC can be reprogrammed to use the other good circuit.

1.2 Contribution

The purpose of this thesis is to explore an alternative strategy for IC test that reduces automated test equipment's cost and enables real-time/online testing. In this thesis a fault tolerant technique is used for IC test. The fault tolerant technique replaces the stored response used in typical IC test methodology with redundancy and majority vote. The proposed fault tolerant technique raises the issue of fault aliasing because of the missing reference response. This thesis describes experiments and collects data to analyze the effect and probability of fault aliasing.

This thesis explores and uncovers the importance of fault cones, introduced in Chapter 2 and describes their impact on fault tolerant technique of digital IC testing. Fault cones provide a means to understand how different single-stuck-at faults decrease the test efficiency. Fault cones show the limitations of fault coverage, when viewed from a fault tolerance perspective. All the experiments and data use ISCAS-85 benchmark circuits to characterize and model the functional unit blocks of a digital IC.

In summary, following are the contributions of this thesis.

- Introduced a feasible IC test technique that replaces the stored response with redundancy and majority voting.
- Analyzed the effect fault aliasing on IC test.
- Determined the fault cones and probability of aliasing for bench mark circuits.
- Determined the fault coverage statistics of random test vectors for benchmark circuits.

1.3 Organization of this thesis

This thesis is divided into five chapters. Chapter 1 outlines the motivation behind the thesis and the contributions of the thesis. Chapter 2 gives a brief background of the problem and introduce the strategy adopted in the thesis. Chapter 3 describes in detail the experimental setup and procedure. Chapter 4 analyzes the experimental results. The effect of fault aliasing is analyzed in detail in Chapter 4.

Chapter 5 concludes the thesis based on the statistical data from the benchmark circuits. Recommendations for possible future work are listed in Chapter 5.

Chapter 2

Background: IC testing and fault tolerance

The cost of IC test increases with the complexity of the circuit. The Figure 2.1 shows the trends of silicon fabrication and test cost per transistor over the years [3]. The x axis marks the year. The cost per transistor on the y axis is in log scale. Although the test cost per transistor has not increased, the manufactured cost per transistor has decreased exponentially over the years. The test cost per transistor has remained almost constant over the years. The concern is that the silicon fabrication cost may cross over the test cost per transistor.

Keeping the test cost low is important in order to keep the IC cost low. Test cost depends on two factors. First, time required to test an IC, i.e test time. Higher the test time, higher is the test cost. Second, cost of the testing equipment.

2.1 The concept of standard reference

The test of digital integrated circuits compares the test pattern results of the Device Under Test (DUT) to the test results of a standard reference. The standard reference is the response of a completely defect free unit. The standard reference response is typically generated from design data, not a physical unit.

The standard reference response is stored in Automated Test Equipment (ATE). The Automated Test Equipment (ATE) sets the necessary test conditions on the Device Under Test (DUT), applies the input test pattern, collects the output result. The ATE compares the reference response with the response obtained from the

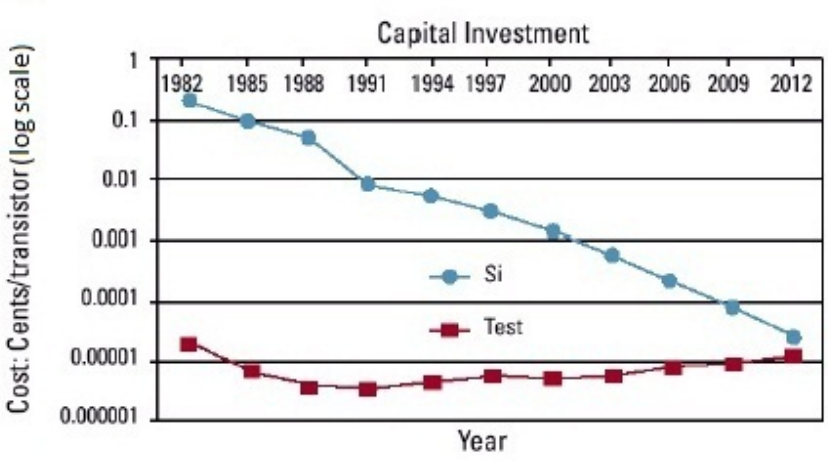

Figure 2.1: Silicon fabrication vs test cost.

DUT and declares a chip defective upon mismatch.

2.1.1 The comparison with standard reference

The input test pattern is applied to the circuit under test and the result is compared with the stored response in the ATE. The circuit under test is tested for exactly the same binary response as the reference response. The Figure 2.2 shows the concept of comparison with reference response [4].

A test procedure needs the test conditions such as temperature, supply voltage and frequency to be accurately controlled. The standard reference data is obtained by setting the test conditions in simulation. The same test conditions must be maintained during the test. The automated test equipment uses simulation data as the reference response during the test. ATE provide the expected working environment for the DUT.

The ATE is expensive and the ATE cost increases with number of tester pin counts and clock frequency. ATE cost has become the bottle-neck for reducing the test

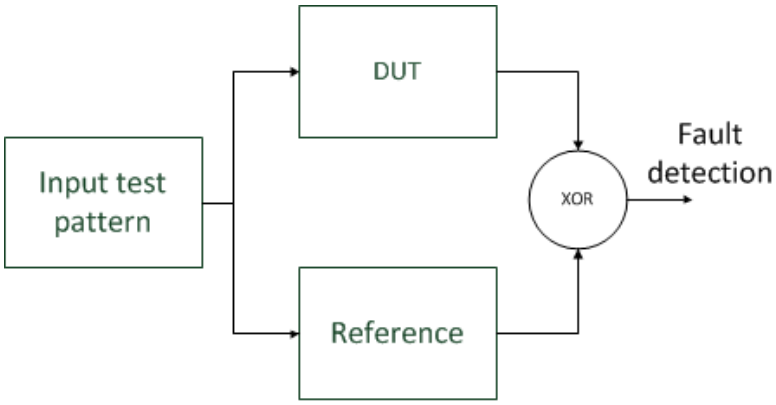

Figure 2.2: The concept of standard reference

cost. Reducing the test cost for ICs is the current major goal for the semiconductor industry and hence the interest of many researchers.

2.2 Introduction to fault tolerance

Fault tolerance is the art and science of building systems that continue to operate satisfactorily in the presence of faults [5]. Fault tolerant circuits provide increased reliability by sustaining correct circuit operation in the presence of faults. Fault tolerance is a technique to mask faults if any, and deliver correct result.

In principle, fault tolerant systems have two characteristics, fault detection and fault correction. Fault detection technique used in fault tolerant systems is analyzed and applied in IC test. The scope of this thesis is confined to fault detection part of fault tolerance systems.

The basic characteristics of fault tolerance are [6]:

- Minimize the incidence of single point of failure in the device under test.
- Fault isolation to the failing component. For example, in the Figure 2.3 the

result of the three blocks A, B and C are independent of each other. A fault in block A will affect only the result of block A.

- Fault containment to prevent propagation of the failure.

Fault-tolerant systems are typically based on the concept of redundancy. The redundant components addresses the fundamental characteristic of fault-tolerance in three ways [6]:

Replication: Providing multiple identical instances of the same system or subsystem, directing tasks or requests to all of them in parallel, and choosing the correct result on the basis of a quorum.

Redundancy: Providing multiple identical instances of the same system and switching to one of the remaining instances in case of a failure.

Diversity: Providing multiple different implementations of the same specification, and using them like replicated systems to cope with errors in a specific implementation.

A lock-step fault tolerant technique uses replicated elements operating concurrently. At any time, all the copies of each element should be in the same state. All the copies are given the same input and clocked at the same time. For the same inputs given to each copy, the same outputs are expected. The outputs of all the copies are compared using a voting circuit [6].

A technique with two copies of DUT is termed Dual Modular Redundancy (DMR). For the same inputs given to each copy of the DUT, the result of the two DUTs

Copy A	Copy B	Copy C	TMR Decision
R1	R1	R1	All copies are fault free
R1	R1	R2	Copy C is faulty
R1	R2	R2	Copy A is faulty
R1	R2	R3	No decision taken

Table 2.1: Typical TMR decision table

may be same or different. Any discrepancy in the result of the two DUT copies indicate that there is fault in at least one of the DUT copies. Same result from both the DUT copies indicate both copies are identical. The voting circuit can then only detect a mismatch in the two DUTs. Decision on faulty and fault free DUT and recovery of correct result relies on other methods.

A technique with three copies of each DUT is termed Triple Modular Redundancy (TMR). The voting circuit can determine the faulty DUT when a two-to-one vote is observed. In a two-to-one vote, the voting circuit can output the correct result, and discard the erroneous version. The case when all the three copies of the DUT give different result, no decision can be made. The Table 2.1 lists the possible results and decision chart of a typical TMR fault tolerant system. R1, R2 and R3 are three different results given by identical copies of the DUT. If all the three copies give the same result the voter declares all the three DUT copies as fault free.

The Figure 2.3 depicts a simple triple module redundancy fault tolerant system. The topology uses three identical blocks evaluating the same data input. The inputs and output bus represents multi-bit data. The output of these three identical block is fed to a voter. The voter block checks if there is any error in the output

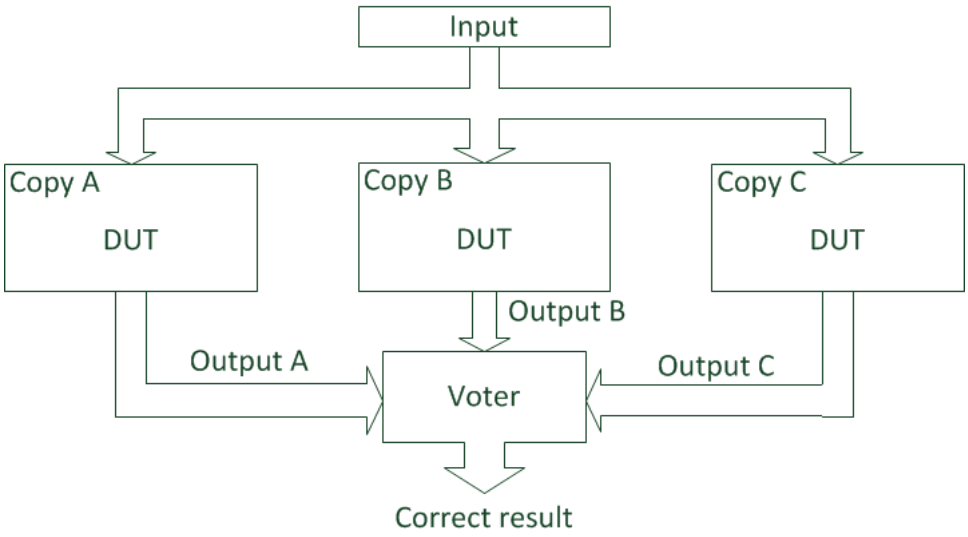

Figure 2.3: Block diagram of a simple TMR system

of any of the block. If there is a fault in one of the DUT the voter detects the faulty DUT copy based on majority vote. This is a single fault detection technique. Aside from being the first approach to hardware fault tolerance, majority voting based fault tolerant circuits are simple to implement [7].

Stroud [7] describes the majority voting and error correction with a TMR example. For IC test error detection is required. This thesis utilizes the error detection technique of majority voting used in fault tolerant systems.

2.3 Fault tolerance technique applied to IC test

Section 2.2 showed how fault tolerant systems are implemented in general. This section exploits and uses the fault tolerance technique to detect stuck-at faults in digital integrated circuits.

ICs are designed with proper structure, modularity and hierarchy. Most basic blocks are used repeatedly in order to implement larger circuit blocks. In many cases there are multiple identical blocks performing same operation. Also, there may be blocks with exactly same functionality, but implemented with different architectures. Logic blocks that are functionally same can be used to make a fault tolerant TMR system described in section 2.2. The fault detection property of the fault tolerant technique is used to test these logic circuits.

The Figure 2.4(a) shows a simple fault tolerant circuit which has the error correction and detection block. There are three copies of the same circuit whose outputs go to a majority voter/error detection and correction block. The error detection block in this case is basically a majority voter. Majority rule is a decision rule that selects alternatives which have a majority, i.e. more than half the votes. For example, if any two of three circuits give the same output, those two circuits are considered good. The assumption here is, only one of the three circuits can be faulty. A fault in any one of the three circuit may change the output result, which will differ from the output result of other two circuits.

The Figure 2.4(b) depicts IC test strategy using the error detection technique used in fault tolerant systems. The Figure 2.4(b) is obtained by replacing error detection and correction in the Figure 2.4(a) by the majority voter.

The faulty circuit can still can be used for the majority vote. For any fault in the circuit, the circuit gives erroneous result only when the fault is sensitized and is observable. For the inputs which do not sensitize and propagate the fault, the circuit behaves as a good circuit supporting the majority vote. The test strategy

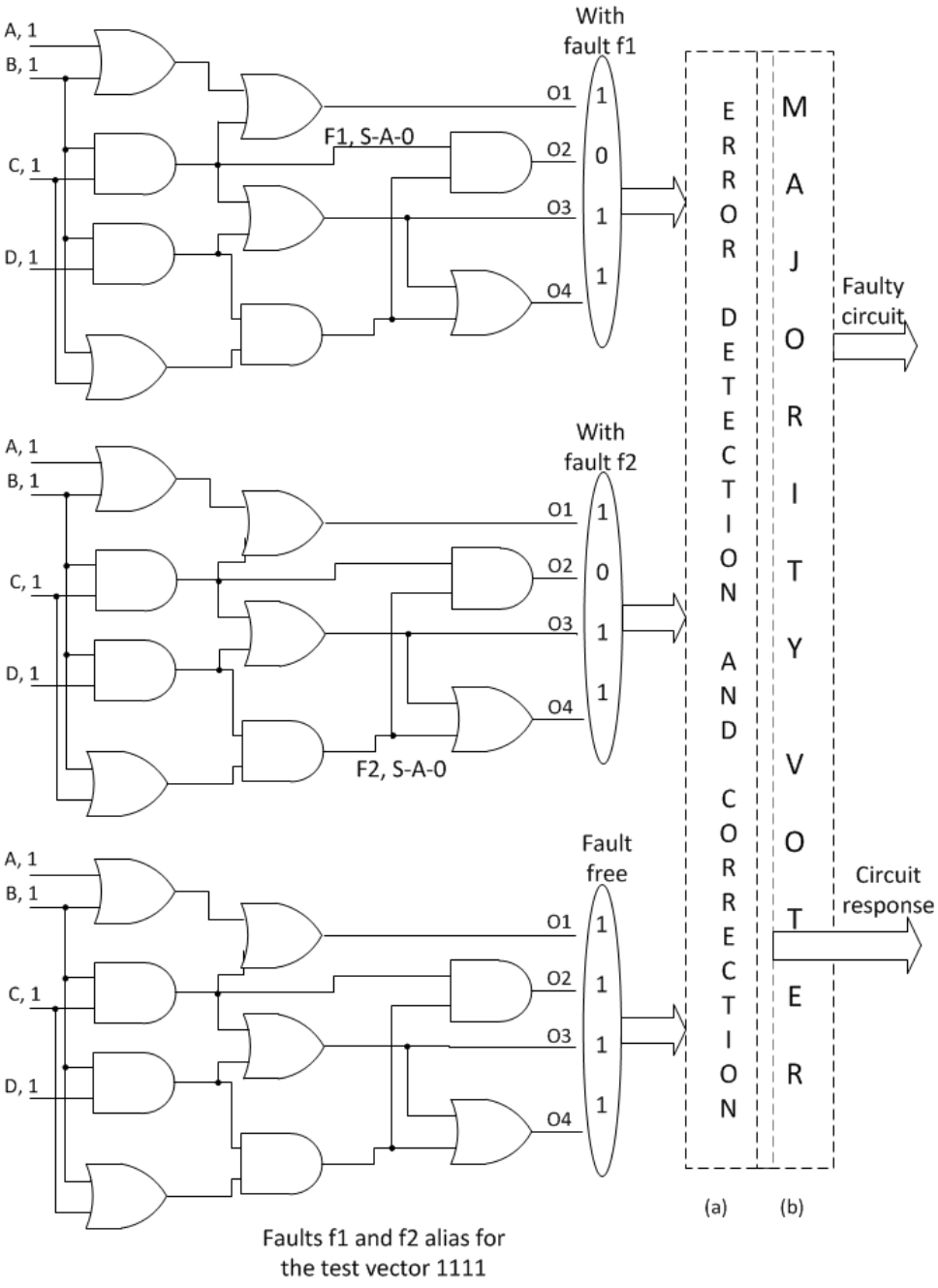

Figure 2.4: Concept of fault tolerance(a) applied for IC test(b)

uses even the faulty circuit for the majority voting. Using a faulty circuit for majority vote might lead to fault aliasing which is discussed in detail in chapters 2, 3 and 4.

2.4 Terminologies

2.4.1 Functionally identical blocks

Integrated circuits are structured and modular. In the design of integrated circuits, the same modules are used multiple times as part of different operation or replicating the same operation multiple times. For example, an arithmetic logic unit may require 32-bit adders. The N 32-bit adders may be implemented with different architectures, but the functionality of all the N adders remain the same. The N adders can be used for parallel processing the data or the adder may be dedicated for a particular task. Circuits with identical pin-outs and functionality are called functionally identical blocks.

In integrated circuits, each module or block performs a specific function such as addition, multiplication, division etc. A circuit block that performs a specific function and gives the result is called Functional Unit Block (FUB). Two functional blocks are functionally identical if they have same number of input and output pins and perform exactly the same function. i.e. for any given input, the two blocks should give binary equivalent results. The blocks that meet these two criteria of same number of input and output pins and same functionality can be grouped as functionally identically blocks.

During the test all the functionally identical blocks are grouped to form a N-modular redundancy system. Where N is the number of circuits which are functionally identical. The N blocks are tested for faults with the majority vote discussed in section 2.3.

2.4.2 Stuck-at faults, testability

In the stuck-at model, a faulty gate input is modeled as a stuck-at-zero (S-A-0) or a stuck-at-one (S-A-1). A S-A-0 indicates that the node is permanently tied to ground GND, and S-A-1 indicates that the node is permanently tied to positive supply VDD. These faults most frequently occur due to gate oxide shorts (the nMOS gate to GND or the pMOS gate to VDD) or metal-to-metal shorts [8]. The faults discussed in this thesis refer to stuck-at faults.

Each node can be in one of the three states, S-A-0, S-A-1 or fault-free. A circuit with n nodes can have possibly $3^n - 1$ stuck-at node combinations. A considerable simplification occurs because of the defect densities usually observed in integrated circuits, multiple defects in a circuit are very unlikely. So, only single stuck-at faults are modeled. Single-stuck-at means only one fault exist in the circuit. An n-node circuit can have at most $2n$ possible single stuck-at faults[9].

To detect a stuck-at fault, the fault needs to be sensitized and observed. Sensitizing the fault is setting the inputs to the circuit such that the logical value at the faulty node is inverse of the fault. For example a S-A-1 node should have an input zero and vice versa.

The controllability of an internal circuit node within a chip is a measure of the ease of setting the node to a one or zero state. In order to sensitize a fault the fault node should be controllable. The internal circuits nodes are not accessible directly. The observability of a particular circuit node is the degree to which you can observe that node at the outputs of an integrated circuit. The sensitized fault need to be propagated to the output pin. To test a node for S-A-0 or S-A-1 faults, the fault should be sensitized and propagated. The fault must be both controllable and observable. The testability of a node is the product of controllability and observability of the fault[9].

2.4.3 Fault cones

A fault cone of a fault is the set of output ports the fault can possibly affect. For different test vectors, faults may be detected at different output ports. The set of output ports where a fault can be possibly detected for the exhaustive test set is the fault cone of the fault. Each fault defines a circuit cone which starts at the fault site and terminates at the primary outputs driven by the fault node [10]. The fault cones discussed in this thesis refers to the fault cones with an exhaustive set of input test vectors.

Consider the example circuit in Figure 2.5. The circuit has four inputs A, B, C and D and four output ports O1, O2, O3 and O4. Two possible stuck-at faults are marked on nodes f1 and f2. The S-A-1 fault on node f1 is detectable at all the four output ports. The S-A-1 at node f2 is detectable only at nodes O3 and O4. The fault cone size of f1 and f2 are four and two respectively.

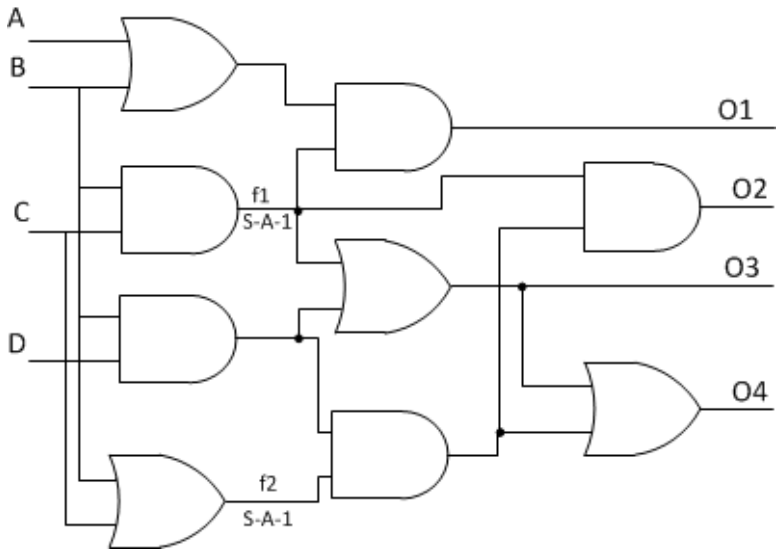

Figure 2.5: Faults and fault cones

Each fault in the circuit has a unique fault cone. Two faults which are detectable at one or more output ports in common have over lapping fault cone with exhaustive set of input test vectors. Different faults may affect the same output port. Faults cones of the faults affecting the same output port overlap. In the Figure 2.5 the fault cones of the S-A-1 faults on nodes f1 and f2 overlap each other.

The Figure 2.6 shows three faults in a FUB and their fault cones. The FUB has 12 output ports. The Figure depicts that for any applied test vector fault A can produce an error at O2 - O6 ports. Fault B can produce an error at ports O3 - O10. Output ports O3 - O6 are affected by both faults A and B. The fault cones of A and B overlap. Fault cones of faults A and B overlap each other. Fault cones of faults B and C overlap each other. Fault cones of faults A and C do not overlap each other. Hence, faults A and B could alias, faults B and C could alias, but faults A and C cannot alias.

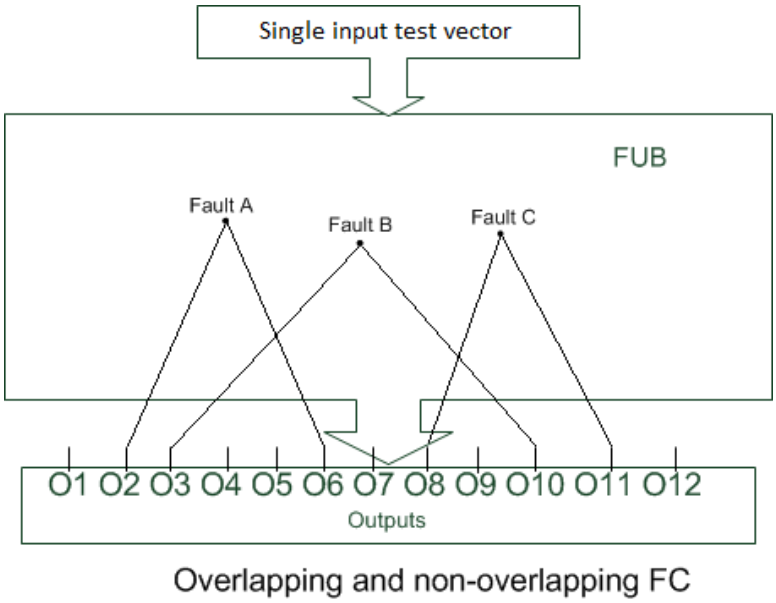

Figure 2.6: Overlapping and non-overlapping fault cones

Fault cones may vary in size from 0 to M, where M is the total number of output ports. If the fault cone of a fault is 0, then the fault is undetectable. If the fault cone of a fault is M, then the fault is highly detectable. Higher the fault cone of a fault, higher the observability of the fault.

A fault cone is a set of all the output ports the fault can possibly affect for any test vector. The fault cones for S-A-0 and S-A-1 fault on the same node may be different. A test set that propagates the fault through all the possible output ports is required in order to get the fault cone. The conventional fault coverage stops testing for a fault once it is detected at one of the output ports. Therefore, the test set that gives the data on fault cones is larger than the full fault coverage test set.

What does the fault cone depend on? Intuitively, its the location of the fault in the circuit and the fan-out of the faulty node. The fan-out in this case is not just

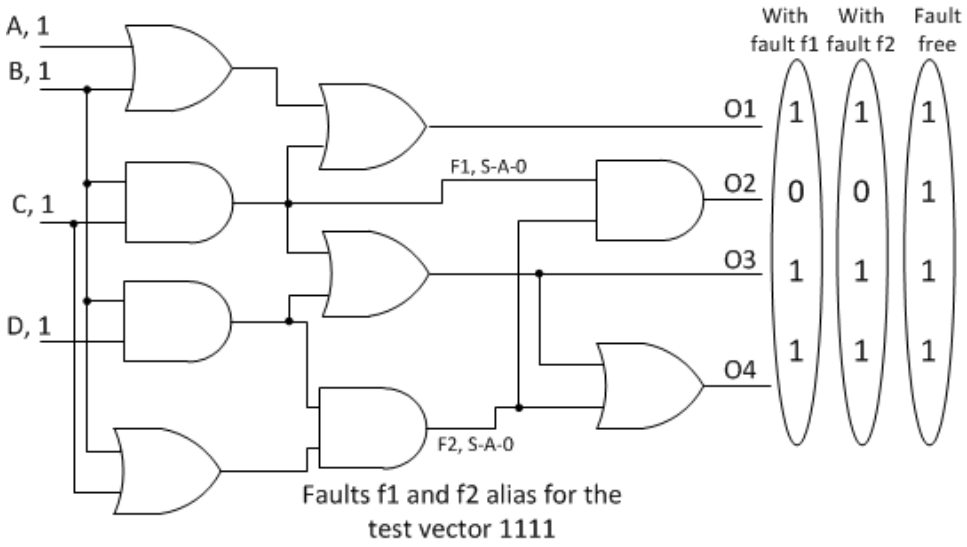

Figure 2.7: Example for single vector aliasing

the fan-out of the faulty node but also its branches down to the output ports.

2.4.4 Fault aliasing

Two identical circuit blocks may produce the same error at the outputs. When the test response of the two identical circuit blocks are compared, the response of both the blocks producing the same error at the output will be same. The faults in the two identical blocks producing the same output error is the working definition of fault aliasing.

The Figure 2.7 shows two faults aliased for a given test vector. The circuit has four inputs A, B, C and D and four outputs O1, O2, O3 and O4. For the test vector 1, 1, 1 and 1 at the inputs A, B, C and D respectively both the single stuck at faults f1 and f2 produce the same output. The two faults make the output O2 toggle from 1 to 0. Thus faults f1 and f2 alias each other.

In a TMR system, if two faults in two of the three circuits produce same error at the respective outputs for a given input, the two faulty circuits will have the majority vote. The two faults alias each other forming majority vote. The fault free circuit giving the correct result will be in minority. Thus, aliasing makes the majority vote decision wrong. This thesis explores the fault aliasing and analyze the statistics and effect of fault aliasing on fault coverage.

In the lock-step fault tolerance technique, if two of the three circuits have faults that produce the same output error for a given input, the two faulty circuit together vote out the other good circuit as a faulty circuit [11]. Fault aliasing makes the fault free circuit fail and passes the faulty circuits.

Fault cones have the statistics of aliasing based on the percentage of overlap of fault cones. In Figure 2.6, the fault cones of FaultA and FaultB overlap and hence can alias. Fault cones of faults A and C never overlap and hence they can never alias. The statistics of fault cones is used to estimate the probability of aliasing.

Aliasing may occur at different levels. In the conventional built-in-self-test (BIST) where the outputs are compressed before comparing, aliasing can occur due to the compaction process. This thesis focuses on fault aliasing at the output ports of the circuit blocks. This is before compaction, if any.

In Figure 2.8, the S-A-0 fault *F1* in Copy A and the S-A-0 fault *F2* in Copy B give exactly the same Output Error Pattern (OEP) of 1,0,1,1. The fault free circuit give the correct result of 1,1,1,1. The S-A-0 fault *F2* in Copy B and the S-A-0 fault *F3* in Copy B are equivalent faults.

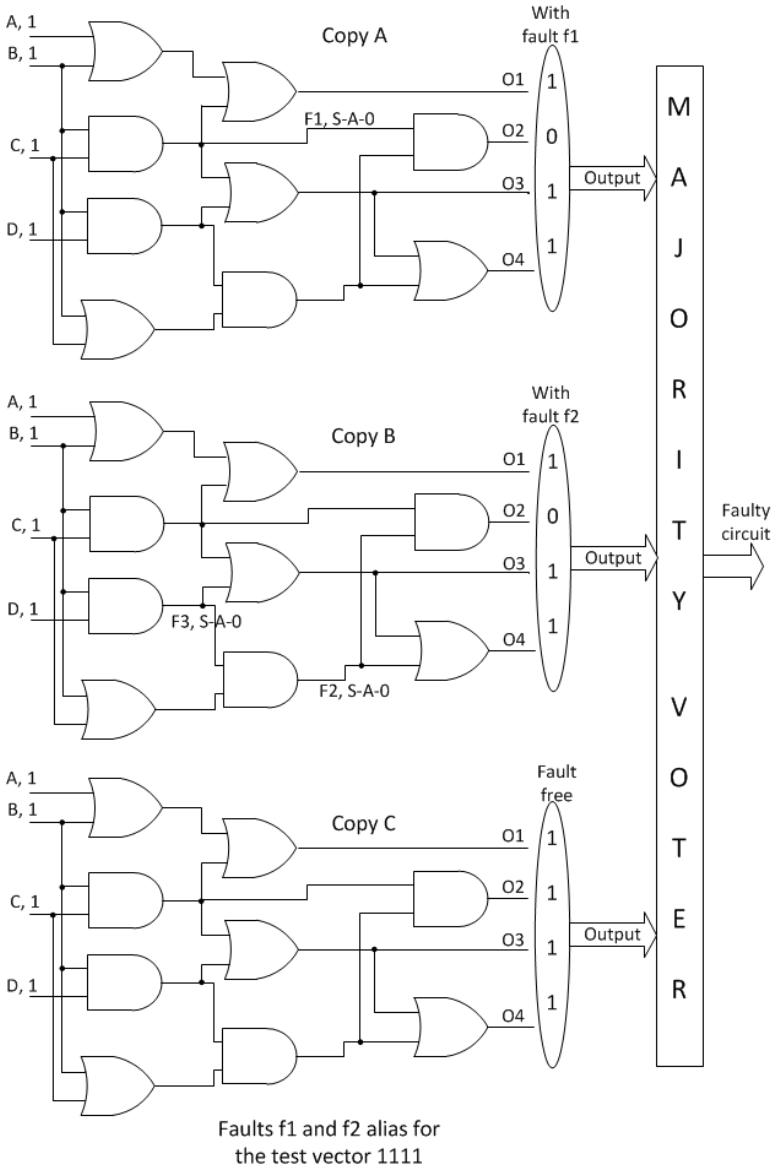

Figure 2.8: Sample TMR depicting aliasing faults

Fault aliasing is not same as equivalent faults [11]. Two faults of a Boolean circuit are called equivalent if they transform the circuit such that the two faulty circuits have identical output functions. Equivalent faults are also called indistinguishable and have exactly same set of tests [12].

Recent publications [13] analyze the yield and product quality required for a desired fault tolerance level in a TMR. The upper and lower bounds for the yield of a TMR system with m single stuck-at faults are derived in [13]. This thesis gives the probability of failure of a TMR system and analyze the relation between fault cones and fault aliasing. This thesis explains which faults are prone to aliasing and why. The ISCAS-85 benchmark circuits are used to analyze the impact of fault aliasing on a TMR system.

2.4.5 ISCAS-85 benchmark circuits

The ISCAS-85 benchmark circuits are a set of well-defined, gate level netlists of common building blocks. These benchmark circuits are part of real projects of different companies. The ISCAS-85 benchmark circuits are widely used in digital integrated circuit research work, in the area of design verification, test generation, clock distribution, power consumption and timing analysis. The gate level netlist is converted transistor level netlist in a format readable by IRSIM and Faultsim [14].

2.4.6 Fault simulation

Fault simulation result shows what happens to the response of circuit when a fault is introduced. In a production test, only the package pins, the Primary Inputs (PIs) and Primary Outputs (POs) are accessible. To test an IC, a series of sets

of input patterns are generated that will detect any faults in the circuit. The simulator used for fault simulation is called fault simulator [15].

Fault simulation can be broadly classified in to two categories described below, based on the input test pattern [16].

Deterministic test vector fault simulation: A set of highly engineered and tailored test vectors are used to simulate the circuit and detect the faults. In order to detect a fault, the fault needs to be sensitized first and then propagate the error due to the fault to one of the output pins. The test pattern is engineered to sensitize and propagate the fault. If a particular fault is undetected, the circuit netlist is traced in order to get a vector that is capable of sensitizing and propagating the fault through the output [16].

Random test vector fault simulation: As an alternative to deterministic fault simulation, instead of trying to simulate and pin every possible fault, use probabilistic fault simulation. Use of random test vectors gives sufficiently high fault coverage (95+%) for reasonable number of test vectors or for equal number of test vectors as in deterministic test pattern. A node is not stuck if the node can be toggled. i.e. change from a 0 to 1 or vice versa. A toggle test checks which nodes toggle as a result of applying test vectors and gives a statistical estimate of vector quality , a measure of faults detected per test vector. A random test vector has equal probability for 0 and 1 for each test vector. Therefore a random test vector is perfect for the toggle test. Testing for nodes toggling simply requires a single logic simulation that is much faster than complete fault simulation [15].

Chapter 3

Experiments and data

In this thesis fault tolerant technique is used for IC test. The fault tolerant technique replaces the need for reference response in typical IC test methodology with redundancy and majority vote. The proposed fault tolerant technique raises the issue of fault aliasing. This thesis sets experiments, collects data to analyze the affect and probability of fault aliasing.

This thesis explores and uncovers the importance of fault cones and their impact on fault tolerant technique of digital IC testing. Fault cones provide a means to understand how different single-stuck-at faults decrease the test efficiency. Fault cones show the limitations of fault coverage, when viewed with a fault tolerance perspective. Fault simulation data of ISCAS-85 benchmark circuits are used to study fault cones and fault aliasing. This chapter describes the experiments and the data from the experiments with ISCAS-85 circuits.

3.1 Experimental circuits

The ISCAS-85 circuits have well-defined, high level structures and functions based on common building blocks such as multiplexers, ALUs, and decoders [17]. The Table 3.1 gives the statistics of the ISCAS-85 circuits [18], [17].

The ISCAS-85 benchmark has a set of 10 combinational circuits. Combinational circuits gives the flexibility to use independent individual random test vectors for testing. Thus, combinational circuits ease the study of fault cones. In this thesis

Circuit	Function	Inputs	Outputs	Gates	fanin	fanout
c432	Interrupt controller	36	7	160	2.10	2.65
c499	32-bit SEC	41	32	202	2.02	4.34
c880	8-bit ALU	60	26	383	1.90	3.50
c1355	32-bit SEC	41	32	546	1.95	2.97
c1908	16-bit SEC/DED	33	25	880	1.70	2.58
c2670	12-bit ALU and cont	233	140	1193	1.64	2.74
c3540	8-bit ALU	50	22	1669	1.76	3.15
c5315	9-bit ALU	178	123	2406	1.90	3.51
c6288	16x16 multiplier	32	32	2406	1.99	2.64
c7552	32-bit comparator	207	108	3512	1.75	2.95

Table 3.1: ISCAS-85 circuit statistics

the three largest circuits from the list of ISCAS-85 benchmark circuits are used. The three largest benchmark circuits provide adequate circuit size to experiment. The benchmark circuits consists of both array and random logic [17]. The benchmarks used for the experiments are briefly described.

c5315, 9-bit ALU:

Statistics: 178 inputs; 123 outputs; 2406 gates

The c5315 benchmark is an ALU that performs arithmetic and logic operations simultaneously on two 9-bit input data words, and also computes the parity of the results. The c5315 benchmark has relatively low testability among the ISCAS-85s [18][19].

c6288, 16x16 Multiplier:

Statistics: 32 inputs; 32 outputs; 2406 gates

The c6288 benchmark is a 16x16 multiplier which multiplies two words of 16 bit and gives a 32 bit result. The circuit consist of 240 full-adders and a half adder arranged in 15x16 matrix. The testability of this is very high

compared to all other ISCAS-85 circuits [18].

c7552, 34-bit adder and magnitude comparator with parity checking:

Statistics: 207 inputs; 108 outputs; 3512 gates

The c7552 benchmark circuit contains a 34-bit adder, a 34-bit magnitude comparator using another 34-bit adder, and a parity checker. Each of the input buses is fed by a set of 2:1 multiplexers controlled by the *Sel* input. This is the largest of the ISCAS-85 circuits [18].

3.2 Fault Simulation procedure

Fault simulation is analysis of the operation of a circuit under various fault conditions. In this thesis only single-stuck-at faults are considered. IRSIM and Faultsim are used for fault simulation of the ISCAS-85 circuits. IRSIM and Faultsim are free and open source tools initially developed at Stanford University[14]. The source code of Faultsim is modified to meet the requirement of multiple detects of the same fault.

IRSIM is a tool for simulating digital circuits. It is a 'switch-level' simulator; that is, it treats transistors as ideal switches. IRSIM uses transistor level netlist to simulate. The transistors are modeled as switches, ignoring most of the higher-order and analog properties of the devices and instead treating each device as an ideal on-or-off connection. The simulator considers three important aspects of digital circuit behavior:

- Transistor state (on or off)
- Logic value (high or low)

- Transition events (from one logic value to another)

Faultsim is a fault simulator that runs in the environment of IRSIM. Faultsim seeds each and every potentially faulty node with a fault and re-simulates the circuit. The simulation result with a seeded fault is compared with the simulation result of the fault-free circuit.

The Figure 3.1 describes a pseudo code for running IRSIM and Faultsim. The pseudo code fault simulates the circuit for N random test vectors and stores the data in *v(i).csv*. The circuit is simulated for all the possible single stuck-at faults in the circuit. The command *relax r* sets all the circuit nodes to a random binary value. The circuit is simulated till the output is settled and then compared with the fault free simulation result.

The fault simulator gives the input to the circuit and simulate. The result of the fault free circuit for the applies test vector is stored. The fault simulator then injects a stuck-at fault at a node in the circuit. In other words, the fault simulator toggles the logic value of fault node. The circuit is simulated with the fault. This injected fault circuit simulation result is XORed with the fault free circuit simulation result. The difference in the XOR gives the output error pattern of the fault.

The random test vectors discussed in Section 2.4.6 are used for fault simulation. There are two main reasons for using random test vectors. First one being the elimination of expensive ATPGs which gives the input test vector and compares the result of the DUT with the stored response. Second, no reference response data is required to evaluate the results in fault-tolerance technique for IC test. The use

```
for (i=1:N)
   {
    irsim scmos03.prm c5315.sim;  # Load the model .prm and the netlist .sim

    relax r;  # Release all nodes of the circuit including the inputs to take random
              values

     s 100;   # Simulate the circuit with random input for 100ns (till all nodes
              settle)

     faultism c5315.in v(i).csv;  # fault simulate the circuit with the existing
                                  random input and store the result in v(i).csv
    }
```

Figure 3.1: Pseudo code for running IRSIM and Faultsim

of random test vectors gives consistent results for fault coverage and fault aliasing as shown in Chapter 4.

3.3 Fault simulation data

The Faultsim data consist of each possible fault and its effect on each output of the circuit, for every test vector applied. The data set from Faultsim consist of five columns shown in Table 3.2. First column, Result-D/F represent whether the fault is detected(Detect) or not(Fail). A fault is termed detected if the effect of the fault is seen at at-least one of the output port. The second column, fault-type is the type of fault S-A-0 or S-A-1. Third column, fault-node represents the node where fault exists. Fourth column, TV no. is the test vector index. Different test vectors are applied to the circuit at different times and then the circuit is simulated with the fault. Each value in TV/time column i.e time represents a random test vector. The fifth column gives the output pin at which the fault is detected.

Result-D/F	**Fault-type**	**Fault-node**	**TV no.**	**O/P-affected**
Detected	0	G690	99.00	G2548
Not-detected	1	G690		
Detected	0	G1170	99.00	G6150
Detected	0	G1170	99.00	G6123
Not-detected	1	G1170		
Not-detected	0	G1146		
Detected	1	G1146	99.00	G6200
Detected	0	G1074	99.00	G6150
Detected	0	G1074	99.00	G6123
Detected	0	G1074	99.00	G5971
Detected	0	G1074	99.00	G5672
Detected	0	G1074	99.00	G5308
Not-detected	1	G1074		
Not-detected	0	G1050		
Not-detected	1	G1050		
Detected	0	G1183	99.00	G6150
Detected	0	G1183	99.00	G6123
Not-detected	1	G1183		
Not-detected	0	G1294		
Detected	1	G1294	99.00	G6150
Detected	1	G1294	99.00	G6123
Detected	1	G1294	99.00	G5971
Detected	1	G1294	99.00	G5672
Detected	1	G1294	99.00	G5308
Not-detected	0	G975		
Detected	1	G975	99.00	G4241

Table 3.2: Sample simulation output from Faultsim

Fault	Fault node	Output error pattern
0	G690	G2548
0	**G1170**	**G6150**, **G6123**
1	G1146	G6200
0	G1074	G6150, G6123, G5971, G5672, G5308
0	**G1183**	**G6150**, **G6123**
1	G975	G4241

Table 3.3: OEP of faults for a single random test vector TV1

The Table 3.2 is a part of the fault simulation data set of c7552 benchmark circuit. Each node in the circuit is denoted prefix 'G' followed by node number. In the sample output file in Table 3.2, S-A-0 fault on node G690 is detected at nodes G2548 and S-A-1 on node G690 is not detected using the test vector no. 99. S-A-0 on node G1170 is detected at two outputs; nodes G6150 and G6123, S-A-1 remain undetected. Similarly, S-A-0 at node G1146 is undetected and S-A-1 is detected at node G6200 . Also, S-A-0 at node G1074 is detected at output nodes G6150, G6123, G5971, G5672 and G5308, S-A-1 remain undetected. At node G1050 both S-A-0 and S-A-1 are not detectable with the given random test vector no. 99.

3.3.1 Faultsim output analysis and per vector aliasing

The output of the fault simulator is analyzed to derive statistics of fault aliasing and fault cones. The data is reorganized to get the fault cone for the applied test vector. The processed data contain all the details of a detected fault. Faults that are not detected are deleted from the data. The processed data appears as below.

The sample data in Table 3.3 represents the outputs affected by faults for the given test vector *TV1*. The list of outputs effected by a fault is called output error pattern (OEP). For example a S-A-0 at G690 affects the output G2548, S-A-0 at

Fault	Fault node	Output error pattern
0	G690	G6123
0	**G1170**	**G6150**, **G6123**
1	G1146	G6200, G6150
0	G1074	G6150, G6123, G5672, G5308
1	G1106	G6200
0	**G1183**	**G6150**, **G5672**
1	G975	G4237

Table 3.4: OEP of faults for a random test vector TV2

G1170 affects the outputs G6150 and G6123. A S-A-0 at G1183 also affects the outputs G6150 and G6123. The two faults S-A-0 at G1170 and S-A-0 at G1183 has the same output error pattern. Now the two faults are indistinguishable. The two faults G6150 and G123 are aliased. The Table 3.4 represents the fault simulation data for another random test vector *TV2*.

For the second test vector also the S-A-0 at G1170 affects the output ports G6150 and G6123, but S-A-0 at G1183 affects the output ports G6150 and G5672. The output error patterns of the two faults now differ and the faults are now distinguishable. The aliasing of the two faults seen by the first test vector is resolved by the second test vector. The aliasing seen within an applied test vector is per vector aliasing. Applying a second test vector may resolve the aliasing seen by the first test vector.

The Table 3.6 describes the condition for fault aliasing per vector. The symbol X denotes an undetected fault, E_i denotes the error pattern of the detected fault. For the two faults to alias, faults have to satisfy two conditions. First, both the faults must be detected by the same test vector. Second, both the detected faults

Faults	Output Error Pattern
73	G4241
6	G4241, G4591, G4946, G5308
1	G4241, G5308, G5971, G6123
1	G4241, G5308, G5971, G6123, G6150, G6160, G6180, G6190, G6210, G6240, G6250, G6260, G6270, G6170, G6280
84	G4591
1	G4591, G4946
17	G4591, G4946, G5308
1	G4591, G5672, G6123, G6150, G6160
1	G4591, G5672, G6123, G6150, G6170, G6180, G6200, G6210, G6250, G6260, G6270, G6280
43	G4946
22	G4946, G5308
1	G4946, G5308, G5971, G6123, G6180, G6220, G6230, G6240, G6250
1	G4946, G5308, G6123, G6170, G6180, G6190, G6200
20	G5308
145	G5308, G5672, G5971, G6123, G6150
1	G5308, G6123, G6150, G6160, G6190, G6230, G6240, G6250, G6260
1	G5308, G6123, G6180, G6190, G6200
1	G545
15	G5672

Table 3.5: Faultsim output analyzed for number of faults aliased and their OEP for one test vector

Fault A	Fault B	Alias result
X	X	Undetected
E_0	E_0	Aliased
X	E_0	Not aliased
E_0	X	Not aliased
E_1	E_2	Not aliased

Table 3.6: Condition for fault aliasing

should have same output error pattern.

The data in the Table 3.5 is derived from the fault simulation data by Faultsim. The outputs affected by each fault are compiled. This data is for one test vector. The first column gives the number of faults that have the output error pattern tabled in second column. The data shows that, there are 73 faults which affect only output $G4241$ for the given test vector. There are six faults that that have the error pattern $G4241, G4591, G4946, G5308$.

The pseudo code in Figure 3.2 summarizes when a fault is said to be aliased. The OEP of each fault is compared with OEP of all other faults in the circuit. If two faults have the same OEP, the two faults are aliased.

3.3.2 Multi-Vector aliasing

Each fault detected by a test vector have a specific output error pattern. A fault may have different output error pattern for different test vectors. The Table 3.5 list a sample data that shows fault aliasing for a single test vector. The first column represents the number of faults that have the same output error pattern. If more than one fault have the same output error pattern, the faults are said to be aliased. When a second test vector is applied, many of these faults which are

```
for (i = 1 : TotalFaults)     # Do for all faults
        {
        if f(i) = detect        # If fault detected
                {
                for (j = 1 : TotalFaults)
                        {
                        if OEP(i) = OEP(j)    # Is OEP of any two faults same?
                                result(i) = aliased;
                        end if;
                        }
                }
        end if;
        }
```

Figure 3.2: Pseudo code for detecting fault aliasing

aliased with the first test vector generate a different error pattern. Faults that generate a different error pattern are no more aliased. Aliasing seen even after applying multiple test vectors is called N-vector aliasing.

The fault simulation data of ISCAS-85 benchmarks show that, the faults that alias even after multiple test vectors are applied, have a fault cone of 1. In other words, the faults which have OEP of one bit for each and every test vector are prone to aliasing. Faults with fault cone of one effect the same output bit for each detect, hence alias with other faults effecting the same output bit.

3.4 Vector dependent and independent equivalence

Fault equivalence: Two faults of a boolean circuit are called equivalent if they transform the circuit such that the two faulty circuits have identical output functions. Equivalent faults are also called indistinguishable and have exactly same set of tests [12].

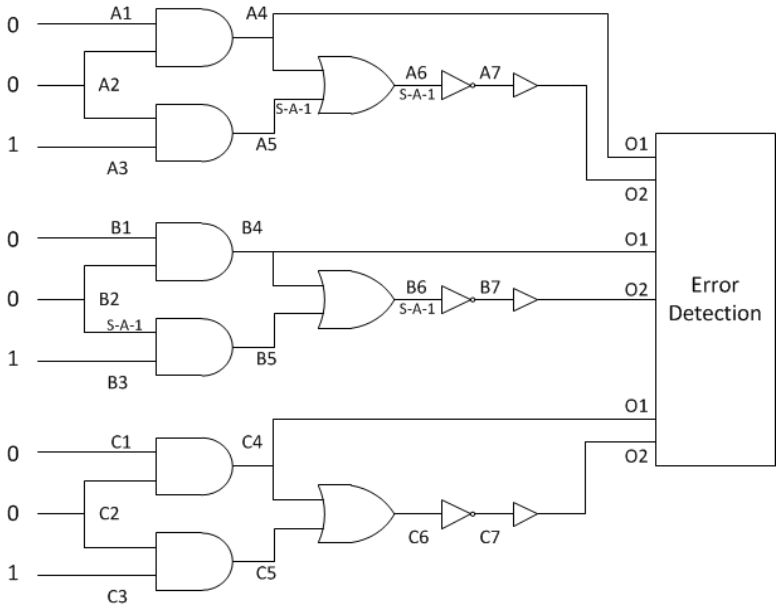

Figure 3.3: Circuit depicting vector dependent and independent equivalence

The conventional fault simulation merge all the equivalent faults into one fault [20]. These faults are equivalent irrespective of the test vector applied. For example a S-A-0 and a S-A-1 faults respectively at the input and output of an inverter are equivalent irrespective of the input test vector. Also, a S-A-0 fault at the input and output of an AND gate are equivalent irrespective of the applied input test vector. Such faults are vector independent equivalent faults.

Fault equivalence of nodes, with the condition of logical value on other nodes are vector dependent equivalent faults. Such faults lead to aliasing, which can be resolved applying different test vectors.

The Figure 3.3 shows a TMR module. There are three copies of the same circuit. In the figure a S-A-1 at nodes A5 and A6 are equivalent independent of the applied test vector. It is true for all the faults that will have identical output functions. The S-A-1 faults at nodes B2 and B6 are equivalent and have the same output error

pattern for the applies vector (0,0,1). The two faults at nodes B2 and B6 affect only the output O2. Output O1 is unaffected by either of the faults. If another vector (1,0,1) is now applied, the S-A-1 fault at B2 affects both the outputs O1 and O2, but the S-A-1 fault at B6 affect only the output O2. Now the two faults have different error pattern and hence the two faults are not aliased.

Chapter 4

Results and Inferences

This chapter summarizes the data and results from the experiments on the three largest ISCAS-85 benchmark circuits discussed in Chapter 3. The statistics of fault coverage, fault aliasing and fault cones are presented.

The data from the benchmark circuits can be used to generalize the fault cone and aliasing statistics for other digital circuits. The three circuits represents both array logic and random logic circuits [18]. The differences in the statistical data between the circuits, if any can be explained by the structure and functionality of the circuit.

4.1 Fault coverage with random test vectors

Fault coverage for a set of test vectors is the fraction of all possible faults detected by the set of test vectors. The Figure 4.1 shows the fault coverage for the three benchmark circuits for different sets of 50 random test vectors. The x-axis is the consecutive number of random test vectors applied and the y-axis is the cumulative fault coverage with applied test vectors.

The Table 4.1 lists the fault coverage of the three circuits for up to 1000 random test vectors. The fault coverage in the table is the cumulative fault coverage for increasing number of random test vectors.

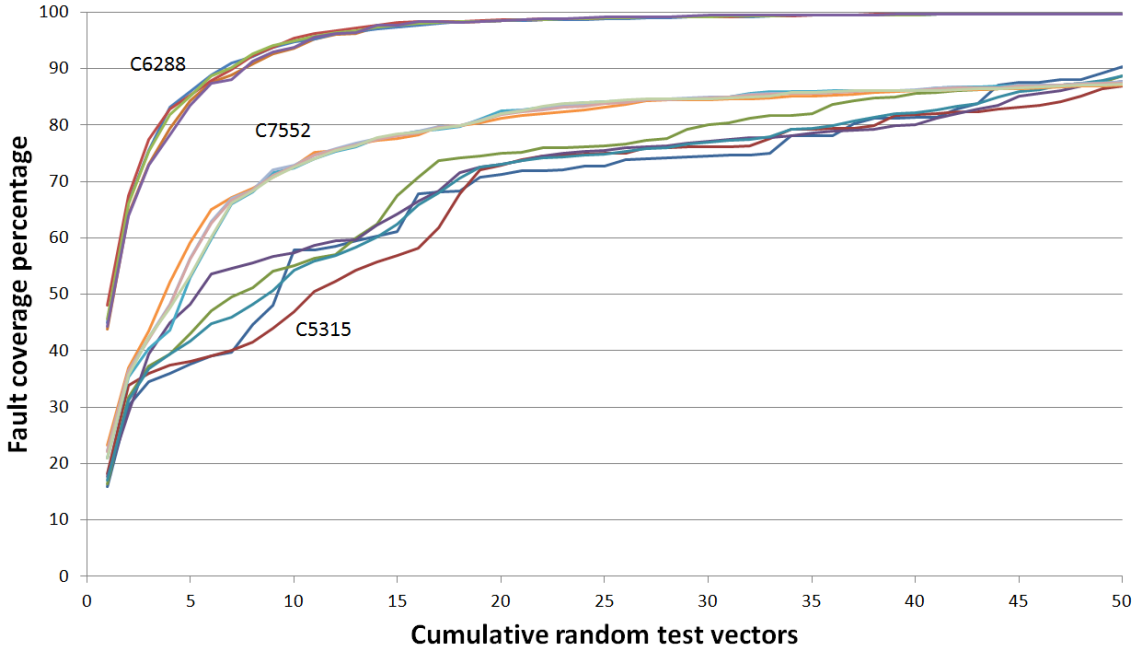

Figure 4.1: Fault coverage for ISCAS-85 circuits with random test vectors

The fault coverage for the circuits c6288 and c7552 are consistent for different random test patterns of 50 test vectors. The circuit c6288 is a 16x16 multiplier. The circuit c7552 is a 34-bit adder and magnitude comparator with parity checking. Both the circuits are array logic. Among the three benchmark circuits considered in this thesis, the circuit c6288 has the highest testability and hence c6288 has high fault coverage of 100% [19]. For the three circuits c5315, c6288 and c7552, with test pattern of 50 random test vectors, the average fault coverage achieved is 88.48%, 99.59% and 87.44% respectively. The Figure 4.1 shows that the fault coverage for each circuit is consistent for different set of 50 random test vectors.

Among the three benchmark circuits considered in this thesis, the benchmark circuit c6288 has highest testability and c5315 has the least testability [17]. The fault simulation data show that for circuits with high testability, fault coverage is

No. of TV	FC for c5315 (%)	FC for c6288 (%)	FC for c7552 (%)
10	57.92	93.62	72.36
20	71.27	98.52	82.52
50	90.36	99.58	87.64
100	95.61	99.78	92.36
200	97.33	99.96	94.90
300	98.47	100.00	96.47
400	98.63	100.00	97.63
500	99.12	100.00	98.21
600	99.36	100.00	98.49
700	100.00	100.00	98.71
800	100.00	100.00	98.92
900	100.00	100.00	99.17
1000	100.00	100.00	99.29

Table 4.1: Fault coverage for three ISCAS-85 circuits up to 1k random test vector

consistent with different sets of random test vectors. The variation in the fault coverage plots below 70% coverage for different sets of random test vectors for the circuit c6288 is 6% lower compared to the variation for c5315.

The fault coverage plots for the circuit c5315 for different sets of random test vectors vary considerably compared to the fault coverage plots of c6288. This variation indicates the need of specific test vectors that have high fault coverage. The circuits whose testability is low need specific test vectors that detect the faults in the circuit. If that specific test vector come as part of random test vector set, more faults are covered than other random test vectors.

The precise relation between the testability of a circuit and fault coverage is given in [19]. Higher the testability of a circuit, higher the fault coverage.

Benchmark	No. of ATPG TV	No. of random TV
c5315	37	700
c6288	12	300
c7552	73	>1000

Table 4.2: Test pattern lengths of ATPG and random TVs for full fault coverage

4.1.1 Comparison of ATPG and random test vectors

In general, fault coverage with random test vectors efficiently detects 90-95% of all possible faults. The remaining 5-10% faults are hard to detect with random test vectors and deterministic approach is preferable [12]. Even so, the deterministic test patterns also may not always give 100% fault coverage efficiently. For the circuit C6288 Figure 4.2 shows a comparison of fault coverage from deterministic test pattern generated from ATPG and random test pattern. The ATPG pattern uses only 12 test vectors for 100% coverage where as with random test vectors fault coverage of 98.52% is achieved with 20 test vectors.

The Table 4.2 gives the test pattern length from ATPG and random test vectors for 100% fault coverage for the three circuits discussed in this thesis. The high testability of the benchmark circuit c6288 is because the c6288 benchmark circuit is a structured multiplier which has 32(16x2) inputs and 32 outputs. The cumulative fault coverage with both ATPG generated patterns and random test patterns for c6288 is shown in Figure 4.2. From the data in tables 4.1 and 4.2, the fault coverage of random test vectors is on par with the ATPG test vectors till initial 90-95% of fault coverage.

The ATPG generated patterns are engineered to detect faults with minimum test vectors. Thus for a given fault coverage, ATPG generated test vectors is always

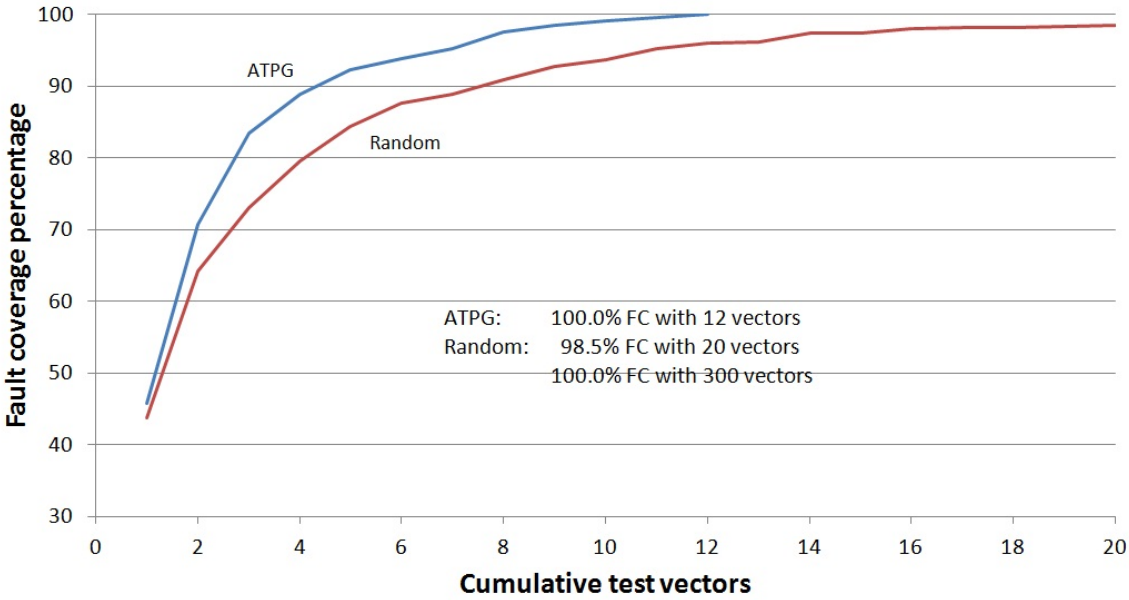

Figure 4.2: Fault coverage for ISCAS-85 C6200 circuit with random test vector and ATPG

less than the number of random test vectors required for the same fault coverage.

4.2 Aliasing for single test vector

The Figure 4.3 shows the number of faults giving the same output error pattern for a test vector. In Figure 4.3 the x-axis is the output error patterns generated by the faults in the circuit. Each vertical bar in the graph shows the number of faults that produce the same output error pattern. The data from the fault simulation shows, more than 90% of detected faults were aliased for a given test vector. The number of unique output error patterns produced is less than the total number of faults in the circuit.

A corollary, a fault affects different outputs for different test vectors. The full set of outputs affected by a fault defines the fault cone of that particular fault. Fault

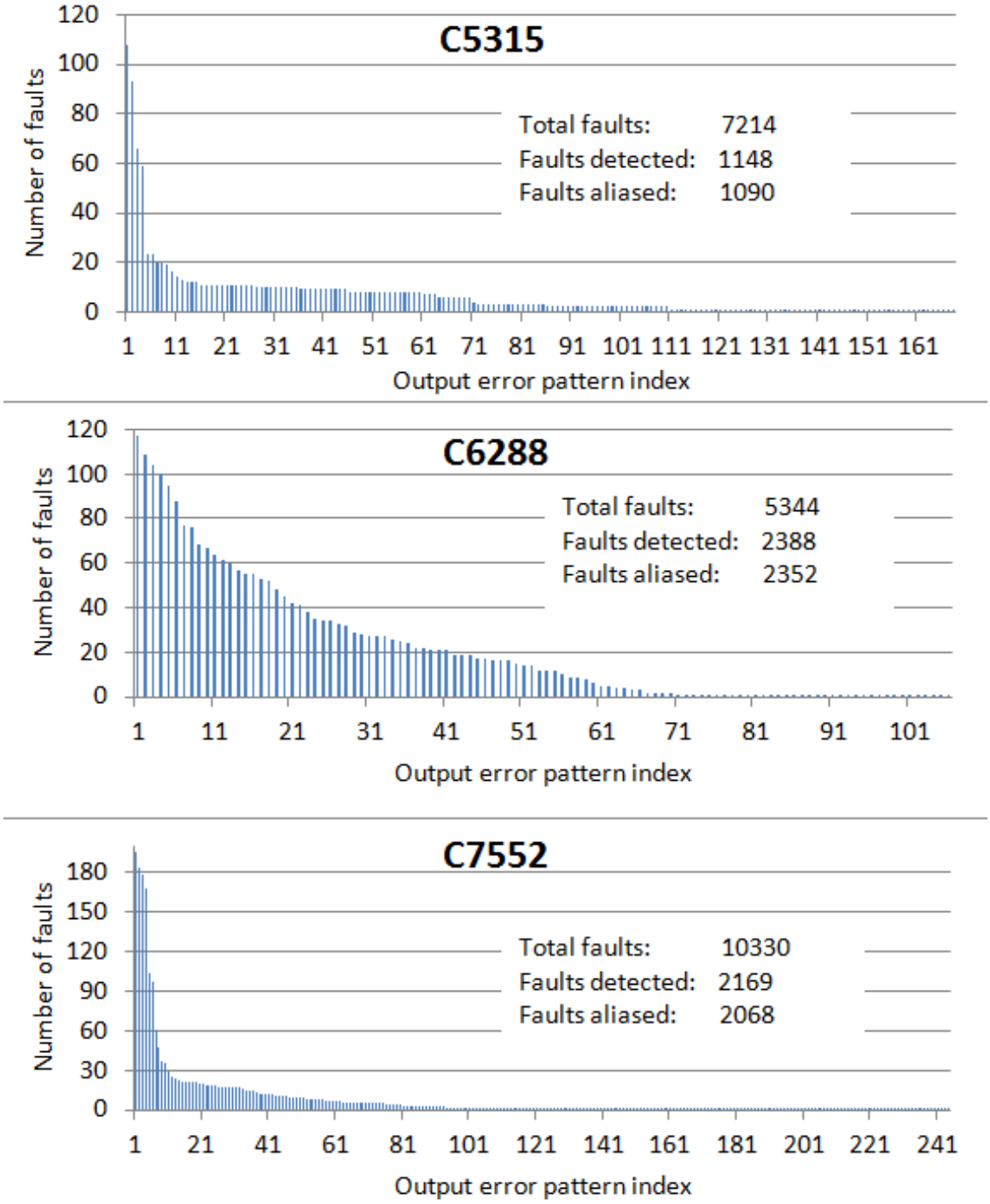

Figure 4.3: Output error pattern aliasing for a single test vector for the three benchmark circuits

cones also show how susceptible are the outputs to faults in the circuit. More the fault cones overlap at a particular output, more susceptible the output is. This increased susceptibility leads to aliasing, producing identical output error patterns. i.e. aliasing occurs when different faults produce a same output error pattern.

4.3 Aliasing: Full fault coverage and beyond

The data from Figure 4.3 show that more than 90% of the faults are associated with bar heights greater than one, that is, aliased for a single input test vector. When another random test vector is applied, some of the faults produce a different output error pattern, increasing the numbers of bars and reducing the height of the bars in the Figure 4.3. Thus reducing fault aliasing. This reduction in fault aliasing with increasing number of random test vectors can be generalized for all circuits.

Consider an example where two identical functional unit blocks FUBA and FUBB have different independent stuck-at faults FaultA and FaultB respectively. Both faults produce the same output error pattern OEPX, for a given random test vector TV1. These faults alias each other. The same two faults FaultA and FaultB produce different output error patterns OEPA and OEPB respectively, for another random test vector TV2. Now, FaultA and FaultB are distinguishable and are not aliased. Here the first test vector aliased the faults and the second test vector resolved the aliased faults. This example shows that different independent faults may produce same error pattern for some test vectors, not necessarily for all test vectors.

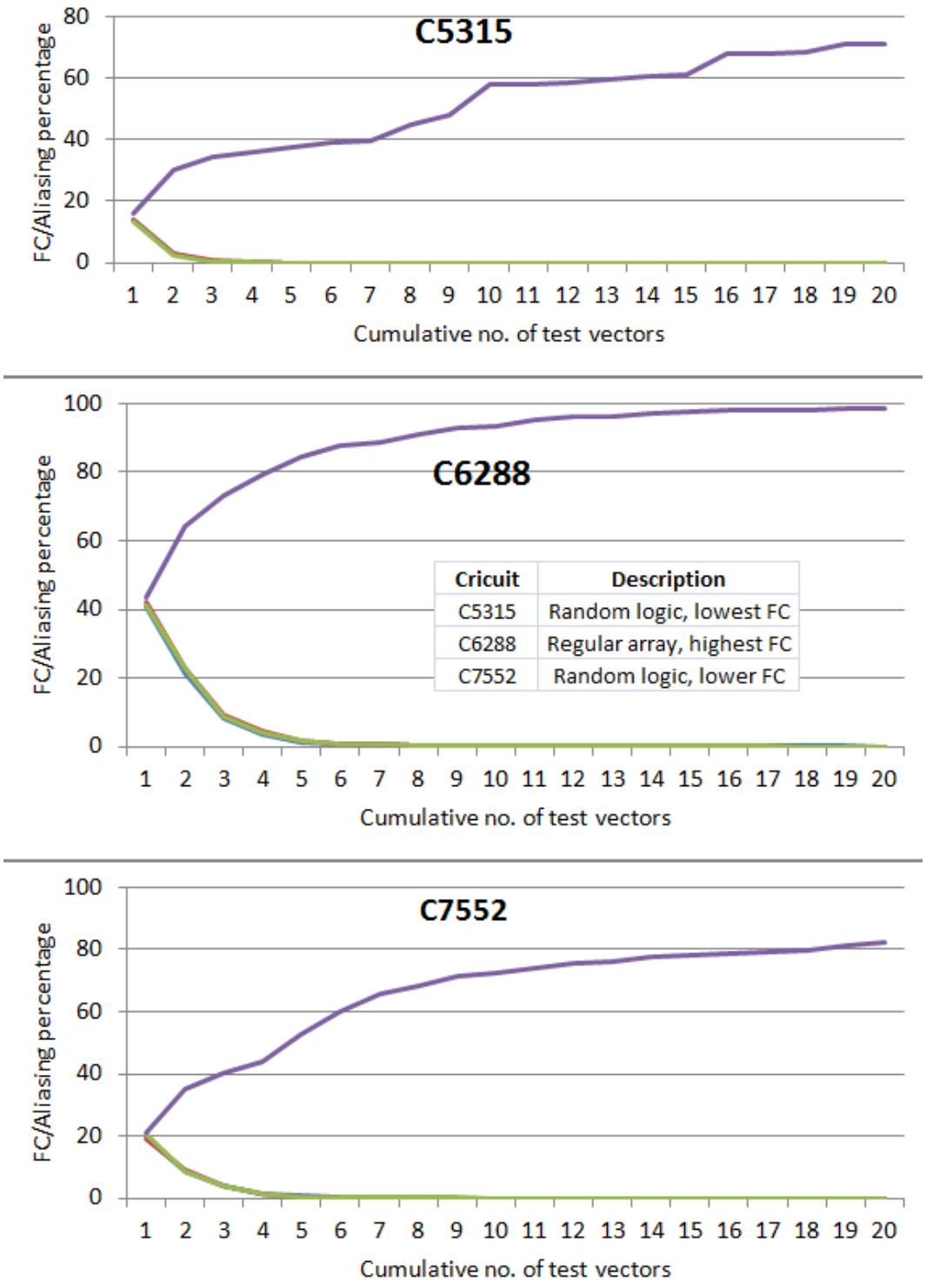

Figure 4.4: Fault coverage and fault aliasing

The Figure 4.4 is a plot of the fraction of faults in Figure 4.3 for which bars have height greater than one verses cumulative number of test vectors. The y axis is the FC/aliasing percentage. x axis is the cumulative test vector. For the first test vector, Aliasing is the sum of all the bar heights greater than one divided by the total number of possible faults. The Table 3.6 is used to categorize fault alias.

The data for C6288 circuit show that approximately each random test vector covers about 44% of total possible single-stuck-at faults. From the data in Figure 4.4, 48% of these faults are aliased. So, $0.44 * 0.48 = 0.21$, that is 21% of the over all possible faults in the circuit. The graph shows the aliasing percentage goes to zero much faster than the undetected faults.

4.4 Summary

The application of new random test vectors resolve aliasing by producing different output error patterns for aliased faults. With every new test vector applied, number of faults aliased decreases. Thus, if a TMR has two faulty units, test escape and overkill can be controlled as aliasing reduces to zero with increasing number of test vectors before the test pattern reaches 100% fault coverage. In a TMR system, if two modules are faulty, it is possible to detect that there are two faults. If all the three modules are faulty and alias then there will be a test escape.

The Table 4.3 summarizes the alias and the number of faults in the TMR. R_i represents a response of the DUT. Column one *case* represents the cases with the responses of the three modules. Column two *TN no.* represents the test vector index applied to the three modules. Columns 3,4 and 5 are the responses of the three copies A, B and C respectively. Column five represents the information

Case	TV no.	Copy A	Copy B	Copy C	No. of faults in the TMR
1	1	R_1	R_1	R_1	No information
2	1	R_1	R_2	R_2	One or two faults, Alias
	2	R_1	R_1	R_2	Two faults, Alias resolved
3	1	R_1	R_2	R_3	Two faults

Table 4.3: Table summarizing alias and the number of faults in a TMR

obtained from the test. The column five in Table 4.3 is different from the Table 2.1 in the sense that the Table 4.3 shows how many modules in a TMR are faulty. The Table 3.6 is used as the condition for fault aliasing.

In the Table 4.3, case one represents a scenario where all the modules give the same response for a given input test vector. If the three modules continue to give the same response for all the applied test vectors, then there are no detectable faults in the TMR. The case two represents scenario where two faults in copies B and C are aliased for the first test vector. When the response of two modules match and the other one differ, there are two possibilities. First, copies B and C may be faulty and the faults alias. Second, copy A may be faulty. Now, for a second test vector, if the response of copies A and B match and the response of C differ as in case two, referencing to the Table 3.6 the faults are not aliased. This indicates that there are two faults in the TMR system. In case 3 where all the three modules have different response for a given test vector, there will be at least two faulty modules in the TMR system.

Aliasing is a direct function of the fault cone of the fault. The data of the three benchmark circuits show that faults with lower fault cones are prone to aliasing. This is an important finding considering the fault aliasing in larger circuit modules.

Nodes that are close to the output ports have lower fault cones. Here, close means the distance in the number of logic levels or the number of gates between the fault and the output port. Hence, faults that are close to the output ports are prone to alias. Thus, the result from this thesis can be generalized to larger circuits.

Chapter 5

Recommendations

In light of the fault aliasing presented in this thesis, this chapter lists the work to be done in exploring efficient alternative for standard reference.

5.1 For sequential circuits

All the experiments carried out in this thesis is on combinational circuits. The reason being ease and simplicity of simulating combinational circuits with random test vectors. Sequential circuits can be broken into smaller combinational circuits with the scan chain and then tested. Also, fault aliasing could be calculated for sequential circuits with some research on the methodology of applying inputs and collecting the response.

Study of fault aliasing and fault cones for sequential circuits is important in order to be able to practically use the methodology described in this thesis.

5.2 Test for larger circuits

The analyzed benchmarks data show that faults closer to the output ports are more prone to aliasing. The small fanout limit the OEP of the fault to a single output port. Assuming this trend holds for larger circuits, determining the probability of aliasing is practical. The larger circuits need to be fault simulated for only the faults close to the output ports.

Aliasing gradually reduces by applying more test vectors to get a different error pattern from the fault. There are two possible methods which can resolve aliasing with less number of test vectors and completely.

- Deterministic pattern to resolve aliasing. Test patterns must be engineered to detect fault and produce different output error patterns.
- A DFT approach, provide auxiliary outputs which reduces the duplication of output error patterns.

Deterministic pattern: Deterministic test vectors engineered to produce a different error pattern by the fault quickly resolves the aliasing. This deterministic test vector is different from the traditional ATPG. The traditional ATPG algorithm is optimized to detect more and more faults and any output port. The traditional ATPG is not concerned about where the fault is detected. The deterministic vector for resolving aliasing need to detect faults at a output port different from output port at which it is already detected.

DFT: Adding an auxiliary output increase the size of the fault cone, thus decreasing the probability of aliasing. The data from the experiments show that most aliasing is because of faults that have a single output port as their fault one. If a fault that has a fault cone equal to one is provided with an auxiliary output which logically combines with other nodes in the circuit aliasing will be easily eliminated for that fault.

5.3 Real-time, real-data testing

Once the need of standard reference unit is eliminated from the test procedure, tests can be designed to run independently within the IC with its own resources available inside the chip. A completely independent built-in-self-test can be designed which test the circuits regularly and keep a log. This give a boost to the reliability of the circuit as any fault will be detected before the fault affect the required operation.

5.4 Limited to stuck-at faults?

The work presented in this thesis is limited to single-stuck-at faults for combinational circuits. It is required to study other fault models such as delay fault and explore the opportunities to eliminate the need of standard reference unit in test procedure.

References

[1] S. Mitra, M. Mitzenmacher, S. Lumetta, and N. Patil, "X-tolerant test response compaction," *Design Test of Computers, IEEE*, vol. 22, no. 6, pp. 566 – 574, nov.-dec. 2005.

[2] A. Maheshwari, W. Burleson, and R. Tessier, "Trading off transient fault tolerance and power consumption in deep submicron (dsm) vlsi circuits," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 12, no. 3, pp. 299 –311, march 2004.

[3] [Online]. Available: http://zone.ni.com/devzone/cda/tut/p/id/2869

[4] R. David and G. Blanchet, "About random fault detection of combinational networks," *Computers, IEEE Transactions on*, vol. C-25, no. 6, pp. 659 –664, june 1976.

[5] D. A. Rennels, "Fault-tolerant computing," in *Encyclopedia of Computer Science*. Chichester, UK: John Wiley and Sons Ltd., pp. 698–702. [Online]. Available: http://dl.acm.org/citation.cfm?id=1074100.1074394

[6] [Online]. Available: http://en.wikipedia.org/wiki/Fault-tolerant-system

[7] C. Stroud, "Reliability of majority voting based vlsi fault-tolerant circuits," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 2, no. 4, pp. 516 –521, dec. 1994.

[8] N. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*, 3rd ed. Boston: Addison Wesley, 2004.

[9] M. L. Bushnell and V. D. Agrawal, *Essentials of electronic testing for digital, memory, and mixed-signal VLSI circuits.* Boston: Kluwer Academic, 2000, vol. 17.

[10] S. Neophytou, M. Michael, and K. Christou, "Generating diverse test sets for multiple fault detections based on fault cone partitioning," in *Defect and Fault Tolerance in VLSI Systems, 2009. DFT '09. 24th IEEE International Symposium on*, oct. 2009, pp. 401 –409.

[11] H. Cox, A. Ivanov, V. Agarwal, and J. Rajski, "On multiple fault coverage and aliasing probability measures," in *Test Conference, 1988. Proceedings. New Frontiers in Testing, International*, sep 1988, pp. 314 –321.

[12] M. L. Bushnell and V. D. Agrawal, *Essentials of electronic testing for digital, memory, and mixed-signal VLSI circuits.* Boston: Kluwer Academic, 2000, vol. 17.

[13] M. Hunger and S. Hellebrand, "The impact of manufacturing defects on the fault tolerance of tmr-systems," in *Defect and Fault Tolerance in VLSI Systems (DFT), 2010 IEEE 25th International Symposium on*, oct. 2010, pp. 101 –108.

[14] [Online]. Available: http://opencircuitdesign.com/irsim/

[15] [Online]. Available: http://iroi.seu.edu.cn/books/asics/Book2/CH14/CH14.4.htm

[16] I. Pomeranz and S. Reddy, "Locstep: a logic-simulation-based test generation procedure," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 16, no. 5, pp. 544 –554, may 1997.

[17] M. Hansen, H. Yalcin, and J. Hayes, "Unveiling the iscas-85 benchmarks: a case study in reverse engineering," *Design Test of Computers, IEEE*, vol. 16, no. 3, pp. 72 –80, 1999.

[18] [Online]. Available: http://www.eecs.umich.edu/ jhayes/iscas/benchmark.html

[19] S. Seth, V. Agrawal, and H. Farhat, "A theory of testability with application to fault coverage analysis," in *European Test Conference, 1989., Proceedings of the 1st*, apr 1989, pp. 139 –143.

[20] E. McCluskey and F. Clegg, "Fault equivalence in combinational logic networks," *Computers, IEEE Transactions on*, vol. C-20, no. 11, pp. 1286 – 1293, nov. 1971.