# ON FAULT COVERAGE AND FAULT SIMULATION FOR MULTIPLE SIGNATURE ANALYSIS BIST SCHEMES

By

Chun Zhang

B. A. Sc. Guilin Institute of Electronic Technology, P.R. China

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

ELECTRICAL ENGINEERING

We accept this thesis as conforming

to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

June 1993

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

(Signature) _____

Department of Electrical Engineering

The University of British Columbia
Vancouver, Canada

Date July 26, 1993

**Abstract**

Fault coverage and fault simulation issues related to multiple signature analysis (MSA) built-in self-test (BIST) schemes are treated here. A model to predict the fault coverage is presented. Based on this model, the nature of the fault coverage for MSA is discussed. The fault coverage of MSA is a function of both the signature sizes and the schedulings. The problem of optimal signature scheduling in MSA to minimize fault simulation time is discussed. Using the developed model, for an arbitrary circuit under test (CUT), given the optimal scheduling positions, given the fault coverage data before data compaction, and given an aliasing threshold, the optimal MSA for the CUT can be readily designed in terms of signature sizes and number of signatures. Analysis and experimental results are presented.

## Table of Contents

**Abstract** ii

**List of Tables** vi

**List of Figures** viii

**Acknowledgement** x

**1 Introduction** 1

1.1 Design for Testability . . . . . 2

1.2 Pseudorandom Testing . . . . . 3

1.3 Conventional IC Testing With Automatic Test Equipments . . . . . 4

1.4 Built-In Self-Test . . . . . 4

1.5 Multiple Signature Analysis . . . . . 6

1.5.1 Hassan's MSA . . . . . 7

1.5.2 Lee's MSA . . . . . 8

1.5.3 Generalized MSA . . . . . 8

1.6 Test Quality Measure . . . . . 10

1.6.1 Fault Model . . . . . 10

1.6.2 Fault Coverage . . . . . 10

1.7 Motivation of This Thesis . . . . . 10

1.7.1 Necessity of Exact Fault Coverage After Compaction . . . . . 10

1.7.2 Necessity of a Fault Coverage Loss Model for MSA . . . . . 11

1.8 Thesis Organization . . . . . 14

**2 Multiple Signature Fault Coverage Loss Model** **15**

2.1 Notations and Preliminaries . . . . . 15

2.1.1 Notations . . . . . 15

2.1.2 Signature Bit Partitions . . . . . 17

2.2 Fault Coverage Model With Signature Analysis . . . . . 18

2.2.1 Basic Definitions . . . . . 19

2.2.2 Fault Coverage of Random Vectors . . . . . 19

2.2.3 Estimated Fault Coverage with Single Signature Analysis . . . . 20

2.2.4 Fault Coverage with Intermediate Signature Analysis . . . . . 21

2.3 A Simplified Fault Coverage Model for Multiple Signature Analysis . . . 22

2.4 Signature Bits Distribution vs. Fault Coverage . . . . . 26

2.5 Experimental Results . . . . . 33

2.5.1 ISCAS'85 Benchmark Circuits . . . . . 33

2.5.2 Experimental Cases . . . . . 34

2.5.3 Results . . . . . 35

2.5.4 Fault Coverage Loss Model Accuracy . . . . . 46

2.6 Conclusions . . . . . 55

**3 Multiple Signature Fault Simulation Time Model** **56**

3.1 Preliminaries and Definitions . . . . . 56

3.1.1 Basic Definitions . . . . . 57

3.2 Time Model . . . . . 58

3.2.1 Fault Simulation Time Model for Multiple Signature Analysis . . 58

3.2.2 Normalized Fault Simulation Time Model . . . . . 60

3.2.3 Justification of the Normalized Fault Simulation Model . . . . . 60

3.3 Equidistant Scheduling & Even Partitioning MSA . . . . . 61
3.3.1 Number of Signatures vs. Fault Simulation Time . . . . . 64
3.4 Optimal Scheduling of Multiple Signature Analysis . . . . . 66
3.4.1 Recursive Relationship . . . . . 66
3.4.2 Optimal Scheduling vs. Aliasing . . . . . 67
3.4.3 Fault Simulation Time vs. Scheduling . . . . . 76
3.5 Fault Coverage Loss vs. Fault Simulation Time . . . . . 79
3.6 Conclusions . . . . . 84

**4 Conclusions** **85**
4.0.1 Future Work . . . . . 86

**Bibliography** **87**

## List of Tables


1.1 Comparison of Aliasing Probability . . . . . 13

2.2 Number of Partitions for $k = 16$ . . . . . 19

2.3 Possible Signature Bit Partitions . . . . . 19

2.4 Hardware Cost Analysis for Multiple Signature Schemes . . . . . 29

2.5 ISCAS'85 Benchmark Circuit Characteristics . . . . . 33

2.6 Fault Coverages for $(16, 16; 1, \ldots, 1; 2048)$ MSA . . . . . 38

2.7 Fault Coverages for $(16, 8; 2, \ldots, 2; 2048)$ MSA . . . . . 38

2.8 Fault Coverages for $(16, 4; 4, \ldots, 4; 2048)$ MSA . . . . . 39

2.9 Fault Coverages for $(16, 2; 8, 8; 2048)$ MSA . . . . . 39

2.10 Fault Coverage Comparison for $(16, 8; 1, \ldots, 1, 9; 2048)$ MSA and $(16, 8; 9, 1, \ldots, 1; 2048)$ MSA . . . . . 40

2.11 Fault Coverage Comparison for (16, 4; 1, 1, 1,13; 2048) MSA, and (16, 4; 13, 1, 1, 1; 2048) MSA . . . . . 40

2.12 Fault Coverage Comparison for (16, 2;1,15; 2048) MSA and (16, 2; 15, 1; 2048) MSA . . . . . 41

2.13 Fault Coverages for (16,16;1,. . . . 1:2048; $O$ ; *aliasing*) MSA and (16,16;1,..., 1;2048; $O$; *no aliasing*) MSA . . . . . 41

2.14 Fault Coverages for (16,8;2,. . . . 2;2048; $O$ ; *aliasing*) MSA and (16,8;2,... , 2;2048;$O$; *no aliasing*) MSA . . . . . 42

2.15 Fault Coverages for (16,4;4,4, 4,4;2048; $O$; *aliasing*) MSA and (16,4;4,4 , 4,4;2048; $O$; *no aliasing*) MSA . . . . . 42

2.16 Fault Coverages for (16,2;8,8;2048; $O$; *aliasing*) MSA and (16,2;8,8;2048;$O$; *no aliasing*) MSA . . . . . 46

3.17 Fault Simulation Time Model Justification for c432, c499, c880, c1355, c1908 62

3.18 Fault Simulation Time Model Justification for c2670, c3540, c5315, c6288, c7552 . . . . . 63

3.19 Fault Simulation Time at Equidistant Scheduling Cases . . . . . 65

3.20 Optimal Scheduling of Signatures for c432, c499, c880, c1355, c1908 for MSA . . . . . 72

3.21 Optimal Scheduling of Signatures for c2670, c3540, c5315, c6288, c7552 for MSA. . . . . 73

3.22 Delta Scheduling Vectors $\delta\mathbf{SV}$ for c432, c499, c880, c1355, c1908 . . . . . 74

3.23 Delta Scheduling Vectors $\delta\mathbf{SV}$ for c2670, c3540, c5315, c6288, c7552 . . . 75

3.24 Fault Simulation Time Saving Comparison for c432, c499, c880, c1355, c1908 77

3.25 Fault Simulation Time Saving Comparison for c2670, c3540, c5315, c6288, c7552 . . . . . 78

3.26 Fault Coverage Data Before Compaction for c1355 . . . . . 80

3.27 Fault Coverage Loss vs. Total Number of Signature Bits . . . . . 81

## List of Figures


1.1 A Generic BIST Scheme . . . . . 5

1.2 Hassan's MSA Model . . . . . 7

1.3 Lee's MSA Model . . . . . 8

1.4 A Generalized MSA Model . . . . . 9

2.5 Balls and Dividers . . . . . 18

2.6 Multiple Signature Analysis Fault Coverage . . . . . 24

2.7 Hardware Cost Analysis . . . . . 31

2.8 Fault Coverage Comparison for $(16, 8; 1, \ldots, 1, 9; 2048)$, $(16, 8; 9, 1, \ldots, 1; 2048)$, and $(16, 8; 2, \ldots, 2; 2048)$ MSA . . . . . 43

2.9 Fault Coverage Comparison for $(16, 4; 1, 1, 1, 13; 2048)$, $(16, 4; 13, 1, 1, 1; 2048)$, and $(16, 4; 4, \ldots, 4; 2048)$ MSA . . . . . 44

2.10 Fault Coverage Comparison for $(16, 2; 1, 15; 2048)$, $(16, 2; 15, 1; 2048)$, and $(16, 2; 8, \ldots, 8; 2048)$ MSA . . . . . 45

2.11 Fault Coverage Before Compaction for c432, c499, c880, c1355, and c1908 47

2.12 Fault Coverage Before Compaction for c2670, c3540, c5315, c6288 and c7552 48

2.13 Fault Coverage Errors for (a) $(16, 16; 1, \ldots, 1; 2048)$ MSA; (b) $(16, 8; 2, \ldots, 2; 2048)$ MSA . . . . . 50

2.14 Fault Coverage Errors for (c) $(16, 4; 4, \ldots, 4; 2048)$ MSA; (d) $(16, 2; 8, 8; 2048)$ MSA . . . . . 51

2.15 Fault Coverage Comparison for MSA and SSA: (a) $(16, 16; 1, \ldots, 1; 2048)$ MSA and $(16, 1; 16; 2048)$ SSA; (b) $(16, 8; 2, \ldots, 2; 2048)$ MSA and $(16, 1; 16; 2048)$ SSA . . . . . 52
2.16 Fault Coverage Comparison for MSA and SSA: (a) $(16, 4; 4, \ldots, 4; 2048)$ MSA and $(16, 1; 16; 2048)$ SSA; (b) $(16, 2; 8, 8; 2048)$ MSA and $(16, 1; 16; 2048)$ SSA . . . . . 53
2.17 Fault Coverage Comparison for Different $n$ . . . . . 54

3.18 Optimal Scheduling of the First Signature of two signatures . . . . . 68
3.19 Optimal Scheduling Algorithm . . . . . 69
3.20 The Movement of Optimal Scheduling . . . . . 70
3.21 Fault Coverage Loss vs. $k$ and $n$ . . . . . 83

## Acknowledgement

I would like to thank Dr. André Ivanov for being as approachable a supervisor as anyone could hope for, for the frequent discussions with him on various aspects of VLSI testing, for his patient reading and correcting mistakes in this thesis, and for his continuous encouragement and support. I also benefited greatly from the productive discussions with Yuejian Wu and the valuable suggestions from him. I am also indebted to my close friends Ellen and Carly for their consistent moral support and help. I owe a very special thanks to my parents and my dear husband for their love and encouragement through my most difficult time.

# Chapter 1

## Introduction

Since the 1970s, a revolution in the field of digital circuit testing has been taking place. This is a revolution that has not ceased, due to the increasing density of integrated circuits (ICs). Pre-1970 circuit packages had perhaps one gate for each two or three package pins. By 1980 the gate-to-pin ratio had not only reversed but also approached 20 gates per pin or more. At present, it is not unusual to find 100 gates or more behind every pin [4]. The increase in package density causes a dramatic reduction per circuit cost, but the percentage of those costs consumed by testing stubbornly increases [4] due to the following reasons:

- Test generation for ICs becomes more and more expensive and difficult as the density of the ICs increases. The test generation algorithms developed for small scale integrated (SSI) or medium scale integrated (MSI) circuits, such as the D-algorithm, are no longer suitable for very large scale integrated (VLSI) circuits.

- Test sequence length required becomes longer and longer as the density of the ICs increases. The increase in test sequence length causes the increase in the required storage for test pattern sets.

- The volume of test output response expands as the test sequence length increases. The increase in the volume of test output response causes the increase in the storage required for the output response, and as well as the increase in the complexity of output response evaluation.

- It is common for the number of memory elements in ICs to increase as the density of ICs increase. The complexities of setting up the memory elements and observing their logic values increase. Although test generation techniques for combinational circuits are relatively mature, test generation techniques for sequential circuits are still struggling.

In order to overcome the difficulties of VLSI testing, research on various aspects of VLSI testing, such as test generation [15][20][21][25][56] and output response evaluation [45][50][13][62], has been conducted actively for over two decades and is still of much interest. From these efforts, the discipline known as design for testability (DFT) [57] has emerged.

## 1.1 Design for Testability

In 1975, Phillip Writer of the Test Equipment Technical in San Diego first proposed [4] the concept of DFT. DFT design techniques use a set of design constraints which lead to more testable designs, especially in the case of sequential circuits. Since then, DFT techniques have rapidly developed.

DFT techniques can be classified into (i) ad hoc design techniques; (ii) structured design techniques. Ad hoc design techniques are just a collection of good design methods that are manually applied with the judgement and skill of the designer. None of these ad hoc methods completely solves the problem of testing sequential circuits or even combinational circuits. For this reason, much more attention has been paid to structured DFT techniques [58][38][18][11]. The most common structured DFT requires compliance to a set of ground rules centered around a uniform design method for latches, generally known as "scan design". The power of scan is that if the states of all latches can be controlled

to any specific value and observed easily, then test generation and fault simulation of a sequential circuit reduce to that of combinational logic. A control signal can switch the memory elements from their normal mode of operation to a mode that makes them controllable and observable. Known scan techniques are level sensitive scan design (LSSD) [10], scan path [16], scan/set logic [51] and random-access scan [3].

With scan design, the test generation of sequential circuits can be reduced to the test generation of combinational circuits. However, the test generation for combinational circuits of VLSI complexity is still very expensive in terms of computing efforts. It has been shown that the test generation problem for combinational circuits belongs to the class of NP-complete problems [24]. In the worst case, problems of this class are exposed to exponential increases in solution time as the size of the problem increases. Pseudorandom testing is one of the possible ways used to avoid the test pattern generation problem.

## 1.2 Pseudorandom Testing

Pseudorandom testing is the testing method which uses pseudorandomly generated test vectors as test patterns. The pseudorandom test patterns can be generated by an autonomous linear feedback shift register (ALFSR) circuit or by a program simulating an ALFSR [39]. Pseudorandom testing can be used to test both combinational and sequential networks. McCluskey et al. [39] discussed combinational circuit pseudorandom testing. Losq [34] discussed sequential circuit pseudorandom testing. Pseudorandom testing can be used either with external automatic test equipment (ATE) testing or with self testing.

## 1.3 Conventional IC Testing With Automatic Test Equipments

In the age of SSI and MSI circuits, the task of testing ICs was mainly fulfilled by ATEs. Testing ICs using ATE includes several steps. First, a test pattern generation algorithm, such as the D-algorithm [6], PODEM [20] or FAN [15], is employed to generate test patterns for each possible modeled fault in the CUT. Second, the previously generated test patterns are applied to the CUT and the resulting outputs are captured by the ATE. Last, the outputs of the CUT are compared with the pre-calculated fault-free output values to decide if the CUT is fault-free. While such test systems can yield excellent performance, they tend to be self-obsolescent since (i) they are extremely expensive, especially for the ATEs designed for circuits of VLSI complexity; (ii) they are made from the same generic integrated circuits that they are designed to test. Hence, by the time they are designed and built by their manufacturers, and purchased and installed by the users, the leading edge of the integrated circuit technology has moved on, creating a test system requirement that the "new" test system can only marginally satisfy. Based on these factors, Bardell, et al. [4] pointed out that the test technology for VLSI must be extended to include BIST in order for progress to continue.

## 1.4 Built-In Self-Test

BIST refers to those testing techniques in which additional hardware is added to a design so that testing is accomplished without the aid of external test equipment. In BIST, both the function of test pattern generation and the function of output response evaluation are incorporated into the CUT. In order to avoid manipulating huge volumes of output data, BIST uses compaction techniques to compact the output data. Examples of compaction techniques include syndrome compaction [45], parity checking [50], signature analysis [13].

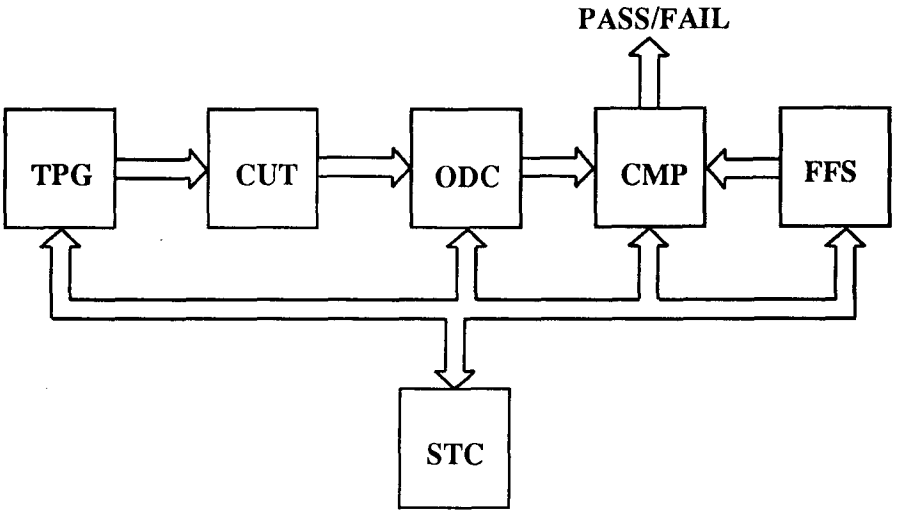



Figure 1.1: A Generic BIST Scheme

Fig. 1.1 illustrates a generic BIST scheme. The CUT is driven by the test pattern generator (TPG) which generally consists of an ALFSR. The output response of the CUT is compacted by an output data compactor (ODC). If the ODC consists of a parallel to serial converter as well as a linear feed back shift register (LFSR) or a multiple input shift register (MISR), the output response is shifted into the LFSR serially or into the MISR in parallel. After the whole output sequence of the CUT is shifted into the LFSR or MISR, the content of the LFSR or MISR forms the *signature* of the output response. At the end, the signature is compared through a comparator (CMP) with the fault free signature (FFS), which is prestored in on-chip memory elements, such as a ROM, to make the final pass or fail decision.

BIST techniques can alleviate the problems of external testing schemes. First, since test patterns are usually generated by an on-chip TPG, the time-consuming effort of test pattern generation is avoided in the design cycle. An additional benefit of the built-in TPG is that the internal nodes of the circuit under test are easily accessed by the

generator and hence a higher fault coverage and shorter test time can be achieved [4]. Moreover, since the output responses are compacted first and the signature is compared only once, the difficulty of storage and analysis of huge volume of both test pattern data and pre-calculated fault-free test response data can be avoided. Thus, BIST can eliminate the use of expensive external test equipment. BIST has another important advantage in that the CUT is fed with test patterns at normal functional clock rate. Hence, BIST is especially suitable for achieve high speed testing.

BIST approaches have been successfully applied to commercial products such as microcomputers [30], signal processing chips [32], and have proved to be a very promising technique. However, BIST is not perfect, and inevitably has weaknesses. Three main drawbacks are:

1. Employing a compactor introduces *aliasing*, the phenomenon of a fault escaping detection after compaction. Test effectiveness is degraded by aliasing.

2. Generally, BIST uses pseudorandom test patterns. Such test sets tend to be long in order to achieve high test quality.

3. Due to compaction, required fault simulation efforts increase tremendously compared to the case of no compaction.

Multiple signature analysis (MSA) has been proposed to overcome the above drawbacks [22] [31] [33].

## 1.5 Multiple Signature Analysis

In [22], Hassan and McCluskey proposed a MSA scheme to increase test quality. In [33], Lee proposed a scheme of checking intermediate signatures to reduce the average testing

time. In [31], Lambidonis et al. proposed a scheme of checking intermediate signatures to reduce the fault simulation efforts for BIST drastically.

The concept of MSA proposed in [22] is different from that of *intermediate signature* proposed in [33]. We review these schemes next, and then give a more general definition of MSA.

### 1.5.1 Hassan's MSA

The MSA proposed in [22], referred to as Hassan's MSA here, is a signature analysis scheme which checks two signatures from the same signature analyzer resulting from two different input sequences. Fig. 1.2 illustrates the scheme. The CUT is driven by the input sequences IS1 and IS2 serially. The resulting output sequence of IS1 is compacted through the ODC to generate the signature of the output sequence. Then the signature is compared by the comparator CMP with predetermined fault free signature FFS1. If the two signatures differ, then the "fail" decision will be made and the test is completed. If they are the same, the same process for IS1 will be repeated for IS2. Final "fail" or "pass" decision will be made. Note that in Hassan's MSA scheme, the input sequences IS1 and IS2 are two different sequences. The output sequences share the same ODC.

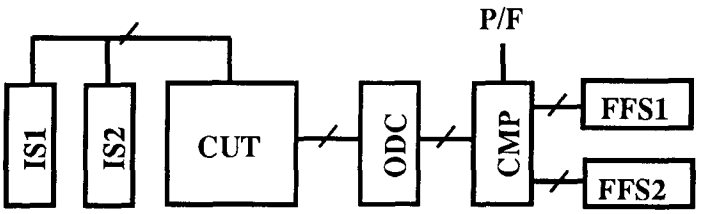



Figure 1.2: Hassan's MSA Model

### 1.5.2 Lee's MSA

The intermediate signature analysis scheme proposed in [33] is a scheme which checks more than one signature from the same signature analyzer at different stages of the test process. Fig. 1.3 gives an example.

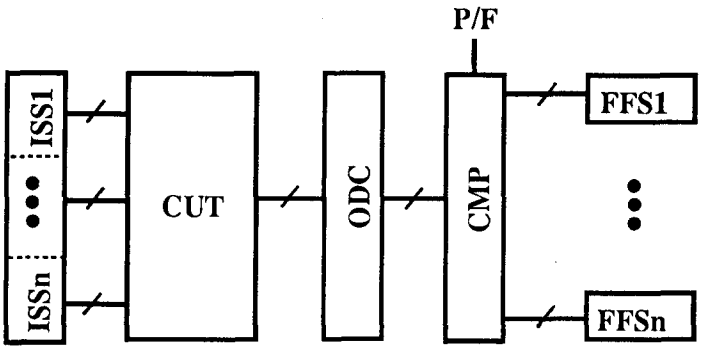



Figure 1.3: Lee's MSA Model

In Lee's MSA model, the input sequences are segments of one single pseudorandom test sequence generated by a LFSR. Input sequence segment ISS1 is applied to the CUT first, the output sequence of ISS1 is compacted through the ODC to generate the signature of the output sequence. Then the signature is compared through CMP with the prestored fault free signature FFS1. If the two signatures differ, then a "fail" decision is made and the test completes. If they are the same, then the same process will repeat for ISS2 and so on until either a "fail" decision is made or the whole test process completes with a "pass" decision. Similar to Hassan's MSA model, Lee's MSA model has only one physical ODC as well.

### 1.5.3 Generalized MSA

Generalized MSA scheme defined here is a scheme which checks multiple signatures from different signature analyzers at different stages of the test process. As illustrated in

Fig. 1.4, for the generalized MSA, multiple FFSs (FFS1, ..., FFSn) are predetermined and prestored in an on-chip ROM or in other forms of storage. The position of the signatures, referred to as *checkpoints* [60], are predetermined as well. Whenever the test session reaches a checkpoint, the signature of the output sequence from a certain signature analyzer is compared to the related FFS, and a pass or fail decision is made. If a fail decision is made, the test is terminated; otherwise, the test continues until either a fail decision is made or the whole test is completed. Similar to Lee's MSA model, the input sequences (ISS1, ISS2, ..., ISSn) for a generalized MSA model are segments of one single pseudorandom test sequence generated by a pseudorandom TPG. For generalized MSA model, it is assumed that there are $n$ physical ODCs, ODC1, ODC2, ..., ODCn. If the number of signature analyzer is one, the generalized MSA model becomes Lee's MSA model.

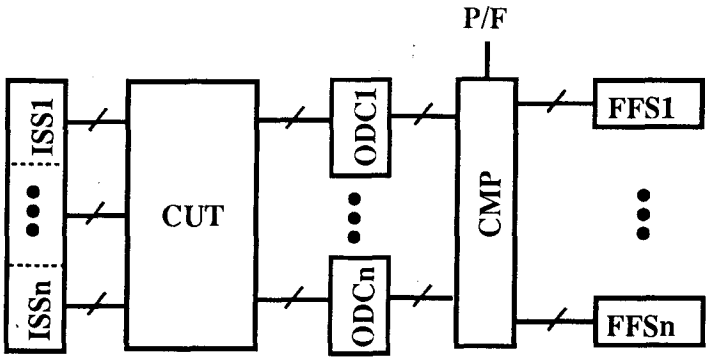



Figure 1.4: A Generalized MSA Model

## 1.6 Test Quality Measure

### 1.6.1 Fault Model

In order to perform functional testing, fault modeling is needed. Fault modeling refers to a set of methodologies through which physical failures are represented by logic models. The logic models used to represent physical failures are called *fault models.* The most often used fault model is the single stuck-at model which allows any node to be stuck-at logical zero (s-a-0) or stuck at logical one (s-a-1) [6].

### 1.6.2 Fault Coverage

One of the most commonly used test quality measure to evaluate testing schemes is *fault coverage.* Fault coverage is defined as the percentage of modeled faults known to be detected [38].

As mentioned in Sec. 1.4, compaction techniques are employed in almost all the BIST techniques. Once compaction is introduced, the fault coverage before and after compaction generally differs due to the compaction information loss. Fault coverage before compaction is performed is referred to as *fault coverage before compaction.* Fault coverage after compaction is performed is referred to as *fault coverage after compaction.*

## 1.7 Motivation of This Thesis

### 1.7.1 Necessity of Exact Fault Coverage After Compaction

In general, a fault goes undetected if none of the input test patterns produces an incorrect circuit output in the presence of the fault. With output response compaction, it is also possible for a fault to fail detection even though the output response differs from the fault-free response. This is known as aliasing, i.e., the case where the output response

from a faulty circuit produces a signature that is identical to the signature of a fault-free circuit.

Due to aliasing, the quality evaluation problem of the test techniques with compaction is different from that of the test techniques without compaction. The effect of compaction is traditionally characterized by aliasing probability [4], i.e., the probability that a fault detected before compaction will escape detection after compaction. Different models have been proposed for characterizing the aliasing behavior of various compaction schemes [9][13][26][27][50][59].

Conventionally, the evaluation of the test quality after compaction is achieved by the fault coverage before compaction and the aliasing probability of compaction. This causes statistical uncertainty in the evaluation of test quality. It may be wasteful to spend considerable hardware and software resources towards achieving a certain known fault coverage before compaction and still end up with uncertainty of the fault coverage after compaction [31]. Thus, Lambidonis et al. [31] proposed a methodology towards the computation of exact fault coverage for signature analysis schemes. The results given in [31] show that the proposed methodology can reduce the fault simulation time with compaction significantly. Hence, exact fault coverage after compaction can be calculated with feasible computation efforts.

### 1.7.2 Necessity of a Fault Coverage Loss Model for MSA

The MSA that the strategy in [31] exploits implies a higher hardware cost. In practice, the allowable BIST overhead limits the total number of signature bits. Given the allowable hardware overhead limit in terms of total number of allowable signature bits, a large number of choices can arise in partitioning the total number of signature bits among various intermediate signatures. The optimal partition of the total number of signature

bits is required such that the fault simulation time with compaction is minimized and fault coverage is maximized. However, before we can meet the requirements, we need to explore the MSA aliasing behavior further since existing models for MSA aliasing behavior [5] are inadequate.

In BIST with single signature analysis (SSA) [4], to compact an output sequence of length $l$, the sequence is shifted bit by bit into a signature analyzer of size $k$. At the end of the test process, the contents of the signature analyzer form the signature. Under the assumption of the equally likely error model [4], the aliasing probability $P_a$ is:

$$P_a = \frac{2^{l-k} - 1}{2^l - 1}. \tag{1.1}$$

When $l \gg k$, $P_a \approx 2^{-k}$.

Similarly, in BIST with MSA, assume that $n$ signatures of sizes $k_1$, $k_2$, ..., $k_n$, are checked at check points $l_1$, $l_2$, ..., $l_n$. The total number of signature bits is $k$, i.e., $\sum_{i=1}^{n} k_i = k$. Under the assumption of the equally likely error model [4], the aliasing probability for the MSA is:

$$P_a = \prod_{i=1}^{n} \frac{2^{l_i - k_i} - 1}{2^{l_i} - 1}. \tag{1.2}$$

When $l_i \gg k_i$ for $i = 1, \ldots, n$, the aliasing probability for the MSA $P_a \approx 2^{-\sum_{i=1}^{n} k_i} = 2^{-k}$.

The $2^{-k}$ result is inadequate for some of the multiple signature analysis cases. In order to illustrate this, an small experiment was done. Consider a case where only two signatures are taken, the total number of signature bits is 4, the total test length is $2^{13} = 8192$, the first signature taken at $2^8 = 256$, and the second signature taken at the end of the test. Fault simulation was done for two different signature bit distributions. For the first signature bit distribution, both signatures are assigned 2 bits. For the second signature bit distribution, the first signature has 1 bit, and the second signature has 3 bits. The

| $D_2(k_1, k_2)$ | $P_a$ | $P_a'$ |
|---|---|---|
| $D_2(2,2)$ | 0.0780 | 0.0625 |
| $D_2(1,3)$ | 0.0668 | 0.0625 |

Table 1.1: Comparison of Aliasing Probability

fault simulation results are presented in Table 1.1. $D_2(k_1, k_2)$ represents the signature bit partition for a MSA scheme when the total number of signatures is 2. $k_1$ denotes the length of the first signature, and $k_2$ denotes the length of the second signature. $P_a$ is the probability of aliasing and is calculated as the ratio of the total number of faults escaped after compaction to the total number of faults detected after compaction. The $P_a$ is the average of ten trials for the ten ISCAS'85 (International Symposium on Circuits and Systems) benchmark circuits [7] and $P_a'$ is the aliasing probability calculated by using $2^{-k}$, which is given in [5].

From the results, we can see that the aliasing probabilities for the two experiments differ even though the total number of signature bit is the same for the two experiments, i.e., $k = 4$.

In order to further explore MSA schemes, more accurate models are needed to depict the aliasing behavior of the MSA schemes. In [61], one such model is developed. Although the model developed in [61] accurately models the aliasing behavior of MSA schemes, the density function of detection probabilities of the faults in a circuit is also required. Usually, it is expensive to obtain a precise detection probability density function for large circuits. In this thesis, a simplified model that only requires the fault coverage information of a circuit without compaction is developed.

## 1.8 Thesis Organization

This thesis proposes a model to predict the fault coverage of MSA. This model allows us to discuss the aliasing behavior of MSA. Furthermore, the model enables the discussion of the relationship between the aliasing behavior of MSA and the latter's acceleration of fault simulation. The remainder of the thesis is organized as follows. Chapter 2 presents a model to predict aliasing and the fault coverage for MSA. Chapter 3 presents a fault simulation time model for MSA. Based on the model, a forward dynamic programming algorithm is developed to yield the optimal signature scheduling to achieve minimal fault simulation time. From the fault coverage prediction model presented in Chapter 2 and the fault simulation time model presented in Chapter 3, trade-offs arise between fault simulation time gain and the fault coverage. Chapter 4 concludes the thesis and points out some directions for future work.

# Chapter 2

# Multiple Signature Fault Coverage Loss Model

## 2.1 Notations and Preliminaries

### 2.1.1 Notations

**$(k, n; k_1, \ldots, k_n; l_1, \ldots, l_n)$ MSA Scheme**

A $(k, n; k_1, \ldots, k_n; l_1, \ldots, l_n)$ MSA scheme represents the following testing scheme: $n$ signatures are checked at the predetermined check points $l_1, l_2, \ldots, l_n$, and that the signatures are of sizes $k_1, k_2, \ldots, k_n$, respectively. The total number of signature bits is $k$.

**$(k, n; k_1, \ldots, k_n; l)$ MSA Scheme**

A $(k, n; k_1, \ldots, k_n; l)$ MSA scheme represents the following testing scheme: $n$ signatures are checked at equidistantly scheduled check points $\frac{l}{n}, \frac{l}{n}, \ldots, \frac{l}{n}$, and that the signatures are of sizes $k_1, k_2, \ldots, k_n$, respectively. The total number of signature bits is $k$.

**$(k, n; k_1, \ldots, k_n; l; O)$ MSA Scheme**

A $(k, n; k_1, \ldots, k_n; l; O)$ MSA scheme represents the following testing scheme: $n$ signatures are checked at optimal scheduled $n$ check points, and that the signatures are of sizes $k_1, k_2, \ldots, k_n$, respectively. The optimal scheduled check points are calculated using the optimal scheduling algorithm developed in chapter 3. The total number of signature bits is $k$.

### $(k, n; k_1, \ldots, k_n; l; O; aliasing)$ MSA Scheme

A $(k, n; k_1, \ldots, k_n; l; O; aliasing)$ MSA scheme represents the following testing scheme: $n$ signatures are checked at optimal scheduled $n$ check points, and that the signatures are of sizes $k_1, k_2, \ldots, k_n$, respectively. The optimal scheduled check points are calculated using the optimal scheduling algorithm developed in chapter 3. Aliasing effect is considered when calculating the optimal scheduling. The total number of signature bits is $k$.

### $(k, n; k_1, \ldots, k_n; l; O; no\ aliasing)$ MSA Scheme

A $(k, n; k_1, \ldots, k_n; l; O; no\ aliasing)$ MSA scheme represents the following testing scheme: $n$ signatures are checked at optimal scheduled $n$ check points, and that the signatures are of sizes $k_1, k_2, \ldots, k_n$, respectively. The optimal scheduled check points are calculated using the optimal scheduling algorithm developed in chapter 3. Aliasing effect is not considered when calculating the optimal scheduling. The total number of signature bits is $k$.

### $(k, 1; k; l)$ SSA Scheme

A $(k, 1; k; l)$ SSA scheme represents the following testing scheme: a single k-bit signature is checked at the end of test of length $l$.

### $D_n(k_1, k_2, \ldots, k_n)$ Partition

A $D_n(k_1, k_2, \ldots, k_n)$ partition denotes the signature bit partition for the case where $n$ signatures are taken and the $n$ signatures are of sizes $k_1, k_2, \ldots, k_n$, respectively.

### 2.1.2 Signature Bit Partitions

There are a number of ways to partition the signature bits among a number of signatures. For example, if the total number of signatures $n = 8$ and the total number of signature bits $k = 16$, the number of possible signature partitions is 6435. For a $(k, n; k_1, \ldots, k_n; l_1, \ldots, l_n)$ MSA scheme, the problem of finding the number of bit partitions is equivalent to determining how many different combinations of $k_i's$ exist which satisfy the conditions:

- $k_i \in \{1, 2, \ldots, k\}, \qquad i = 1, 2, \ldots, n;$
- $\sum_{i=1}^{n} k_i = k;$
- $n \leq k.$

The first condition gives a constraint on the length of each signature analyzer, i.e., the length of each signature analyzer has to be an integer within range $[1, k]$. The second condition implies that the total number of signature bits is fixed. The third condition gives a constraint on the total number of checked signatures.

**Theorem 1:** The number of possible partitions of signature sizes $k_1, k_2, \ldots, k_n$ $(k_i \geq 1)$ for a total of $k$ signature bits and a total of $n$ signatures checkpoints is $\begin{pmatrix} k-1 \\ n-1 \end{pmatrix}$.

**Proof:** Imagine $k$ balls in a straight line with $k-1$ spaces between those balls (see Fig. 2.5). We need to divide the $k$ balls into $n$ groups. Therefore, $n-1$ dividers should be put between the $k$ balls under the constraint that for each space, at most one divider can be put in. Then the number of ways of putting $(n-1)$ dividers in the $(k-1)$ spaces between $k$ balls equals the number of all possible partitions of signature lengths

$k_1, k_2, \ldots, k_n$. Obviously, the number of ways of putting $(n-1)$ dividers in the $(k-1)$ spaces between $k$ balls is $\begin{pmatrix} k-1 \\ n-1 \end{pmatrix}$. □

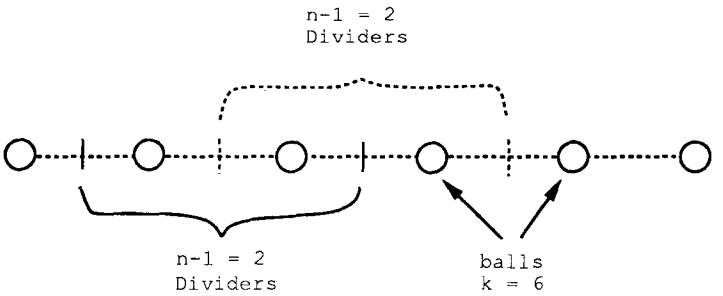



Figure 2.5: Balls and Dividers

Each ball in the proof above represents one signature bit. The number of balls between any two adjacent dividers represents the number of signature bits allocated to some signature, say, $k_i$.

From Theorem 1, given $k$ and $n$, the number of all possible partitions of $k_1, k_2, \ldots, k_n$ is

$$\frac{(k-1)!}{(n-1)!(k-n)!}. \tag{2.3}$$

Table 2.2 gives the number of partitions for different $n$ with $k = 16$. Table 2.3 gives all 15 possible partitions for the $k = 16$, $n = 2$ case. $k_1$ is the size of the first signature and $k_2$ is the size of the second signature.

## 2.2 Fault Coverage Model With Signature Analysis

In this section, we introduce some definitions and briefly review some results presented in [49] and [61].

| $n$ | # of Partitions | $n$ | # of Partitions |
|---|---|---|---|
| 1 | 1 | 9 | 6435 |
| 2 | 15 | 10 | 5005 |
| 3 | 105 | 11 | 3003 |
| 4 | 455 | 12 | 1365 |
| 5 | 1365 | 13 | 455 |
| 6 | 3003 | 14 | 105 |
| 7 | 5005 | 15 | 15 |
| 8 | 6435 | 16 | 1 |

Table 2.2: Number of Partitions for $k = 16$

| Possible Signature Bit Partitions ($k = 16$, $n = 2$) | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $k_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| $k_2$ | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

Table 2.3: Possible Signature Bit Partitions for $n = 2$ and $k = 16$.

### 2.2.1 Basic Definitions

**Definition 1:** The *detection probability* of a fault is the probability of detecting an existing fault by a single random test vector [49].

Detection probabilities of faults in a circuit can be represented by a density function or distribution $p(x)$ such that $p(x)dx$ corresponds to the fraction of testable faults with detection probability between $x$ and $x + dx$. Since $x$ represents a probability, therefore:

$$\int_0^1 p(x)dx = 1. \tag{2.4}$$

### 2.2.2 Fault Coverage of Random Vectors

For the sake of completeness, we review a derivation from [49] next. Since there are $p(x)dx$ faults with detection probability $x$, the mean coverage among these faults by a

random vector is $xp(x)dx$ [49]. Suppose we apply a sequence of random vectors to the CUT. The mean coverage by the first vector is:

$$y_1 = \int_0^1 xp(x)dx. \tag{2.5}$$

Actual coverage by a random vector might be different from the mean by a random quantity. However, the variance will be small for almost all circuits [49]. After removing the faults detected by the first vector, the normalized number of the remaining undetected faults $UDT$ is:

$$\begin{aligned} UDT &= 1 - \int_0^1 xp(x)dx \\ &= \int_0^1 (1-x)p(x)dx. \end{aligned} \tag{2.6}$$

Therefore, the distribution of detection probabilities of the remaining faults is $(1-x)p(x)$. Thus, the coverage of two random vectors is:

$$\begin{aligned} y_2 &= y_1 + \int_0^1 x(1-x)p(x)dx \\ &= \int_0^1 x[1+(1-x)]p(x)dx. \end{aligned} \tag{2.7}$$

Similarly, we can find the coverage of $l$ vectors to be:

$$\begin{aligned} y_l &= \int_0^1 x[1+(1-x)+(1-x)^2+\ldots+(1-x)^{l-1}]p(x)dx \\ &= 1 - \int_0^1 (1-x)^l p(x)dx \\ &= 1 - I(l), \end{aligned} \tag{2.8}$$

where $I(l) = \int_0^1 (1-x)^l p(x)dx$.

### 2.2.3 Estimated Fault Coverage with Single Signature Analysis

Assuming a $k$-bit signature is checked after applying $l$ random vectors to a CUT, the asymptotic aliasing probability is $\rho = 2^{-k}$ under the equally likely error sequences assumption [5]. If we denote the probability of no aliasing by $\beta$, i.e., $\beta = 1 - \rho$, then the

fault coverage, $FC^*$, after compaction when checking a single signature is:

$$\begin{aligned} FC^* &= \beta y_l \\ FC^* &= \beta[1 - \int_0^1 (1-x)^l p(x)dx] \\ &= \beta(1 - I(l)). \end{aligned} \tag{2.9}$$

This result has been been verified experimentally by Rajski in [43].

### 2.2.4 Fault Coverage with Intermediate Signature Analysis

For a $(k, n; k_1, \ldots, k_n, l_1, \ldots, l_n)$ MSA scheme, assume that the aliasing probabilities associated with the $n$ signature analyzers are $\rho_1, \rho_2, \ldots, \rho_n$, respectively, and the corresponding probabilities of no aliasing are $\beta_1, \beta_2, \ldots, \beta_n$, where $\rho_i = 2^{-k_i}, \beta_i = 1 - \rho_i$ and $i = 1, 2, \ldots, n$. Then, similar to the analysis in Sec. 2.2.3, the expected fault coverage after checking the first signature at $l_1$ is:

$$\begin{aligned} FC_1 &= \beta_1 y_{l_1} \\ FC_1 &= \beta_1[1 - \int_0^1 (1-x)^{l_1} p(x)dx] \\ &= \beta_1[1 - I(l_1)]. \end{aligned} \tag{2.10}$$

The proportion of faults that remain undetected after checking the first signature is:

$$\begin{aligned} UDT_1 &= 1 - FC_1 \\ &= 1 - \beta_1(1 - \int_0^1 (1-x)^{l_1} p(x)dx) \\ &= \int_0^1 [1 - \beta_1 + \beta_1(1-x)^{l_1}]p(x)dx. \end{aligned} \tag{2.11}$$

The new distribution of the detection probabilities of the remaining faults can be represented by $p_1(x) = [1 - \beta_1 + \beta_1(1-x)^{l_1}]p(x)$. Therefore, the fault coverage after checking

the second signature at check point $l_2$ is:

$$\begin{aligned} FC_2 &= FC_1 + \beta_2[UDT_1 - \int_0^1 (1-x)^{l_2} p_1(x)]dx \\ &= FC_1 + \beta_2\{UDT_1 - \int_0^1 (1-x)^{l_2}[1-\beta_1+\beta_1(1-x)^{l_1}]p(x)dx\} \\ &= [\beta_2 + \beta_1(1-\beta_2)] - \beta_1(1-\beta_2)I(l_1) - \\ &\quad \beta_2(1-\beta_1)I(l_2) - \beta_1\beta_2 I(l_1+l_2). \end{aligned} \tag{2.12}$$

Similarly, we can find the distribution of the detection probabilities of the remaining faults, and the fault coverage after checking the third signature, and so forth. In general, the fault coverage after checking the $n^{th}$ signature is:

$$FC = \beta_n + \sum_{i=1}^{n-1}[\beta_i \prod_{j=i+1}^{n}(1-\beta_j)] + \sum_{k=1}^{n}\sum_{A_k} \beta_{i_1}\ldots\beta_{i_k}\rho_{i_{k+1}}\ldots\rho_{i_n} I(l_{i_1}+\ldots+l_{i_k}), \tag{2.13}$$

where $A_k = \{(i_1,\ldots,i_k,i_{k+1},\ldots,i_n);\ 1 \le i_1, < \ldots, < i_k \le n, 1 \le i_{k+1}, < \ldots, < i_n \le n$ and $(i_1,\ldots,i_k,i_{k+1},\ldots,i_n)$ is a permutation of $(1,2,...,n)\}$. For example, if $n=4, k=2$, then $A_2 = \{(1,2,3,4),(1,3,2,4),(1,4,2,3),(2,3,1,4),(2,4,1,3),(3,4,1,2)\}$.

## 2.3 A Simplified Fault Coverage Model for Multiple Signature Analysis

In Sec. 2.2.4 a model for predicting the fault coverage for MSA is given. The model is based on the knowledge of a density function of detection probabilities of the faults in a circuit. Usually, it is expensive to obtain a precise detection probability density function for large circuits. In [43], it has been shown that fault coverage with single signature analysis can be well estimated by using the fault coverage before data compaction and the aliasing performance of the signature analyzer. In this section, we present a simplified model for predicting the fault coverage for MSA. As in [43], this model is also based on the fault coverage before data compaction and the aliasing performance of the signature analyzers.

For a $(k, n; k_1, \ldots, k_n; l_1, \ldots, l_n)$ MSA scheme, assume that the corresponding aliasing probabilities associated with the $n$ signature analyzers are $\rho_1, \rho_2, \ldots, \rho_n$, respectively. Further, assume that, at the check points, the corresponding fault coverage before data compaction is known to be $F_1, F_2, \ldots, F_n$. The portion of additional faults detected during the segment $[l_{i-1}, l_i]$ of test patterns is thus $F_i - F_{i-1}$, for $i = 1, 2, \ldots, n$, with $F_0 \equiv 0$. For example, at the check point $l_1$, the percentage of faults detected is $F_1 - F_0 = F_1$, and the faults detected during segment $[l_1, l_2]$ is $F_2 - F_1$ (see Fig. 2.6). According to [43], if a signature is checked at $l_1$, $F_1\rho_1$ of the faults detected in the segment $[l_0, l_1]$ would be aliased. Assume that all the aliased faults will be redetected by the test vectors during the segments $[l_{i-1}, l_i]$ for $i = 2, 3, \ldots, n$. Consequently, after checking the second signature at $l_2$, the number of faults from $F_1$ escaping detection due to the aliasing of the two signatures would be $F_1\rho_1\rho_2$. Thus, the portion of faults from $F_1$ that would escape detection after checking $n$ signatures is:

$$FCL_1 = F_1 \prod_{j=1}^{n} \rho_j. \tag{2.14}$$

Similarly, for the $(F_2 - F_1)$ faults first detected in segment $[l_1, l_2]$, after checking the second signature at $l_2$, $(F_2 - F_1)\rho_2$ of the faults would be aliased due to data compaction. Note that the aliasing of the first signature checked at $l_1$ has no impact on the faults detected after $l_1$. Again, assuming that these aliased faults will be redetected by the test vectors during segments $[l_{i-1}, l_i]$, for $i = 3, 4, \ldots, n$, the portion of faults aliased from $(F_2 - F_1)$ when $n$ signatures are checked is:

$$FCL_2 = (F_2 - F_1) \prod_{j=2}^{n} \rho_j. \tag{2.15}$$

In general, for the $(F_i - F_{i-1})$ faults first detected during the segment $[l_{i-1}, l_i]$, the portion of the faults aliased from these after taking $n$ signatures is:

$$FCL_i = (F_i - F_{i-1}) \prod_{j=i}^{n} \rho_j, \qquad i = 1, 2, \ldots, n. \tag{2.16}$$

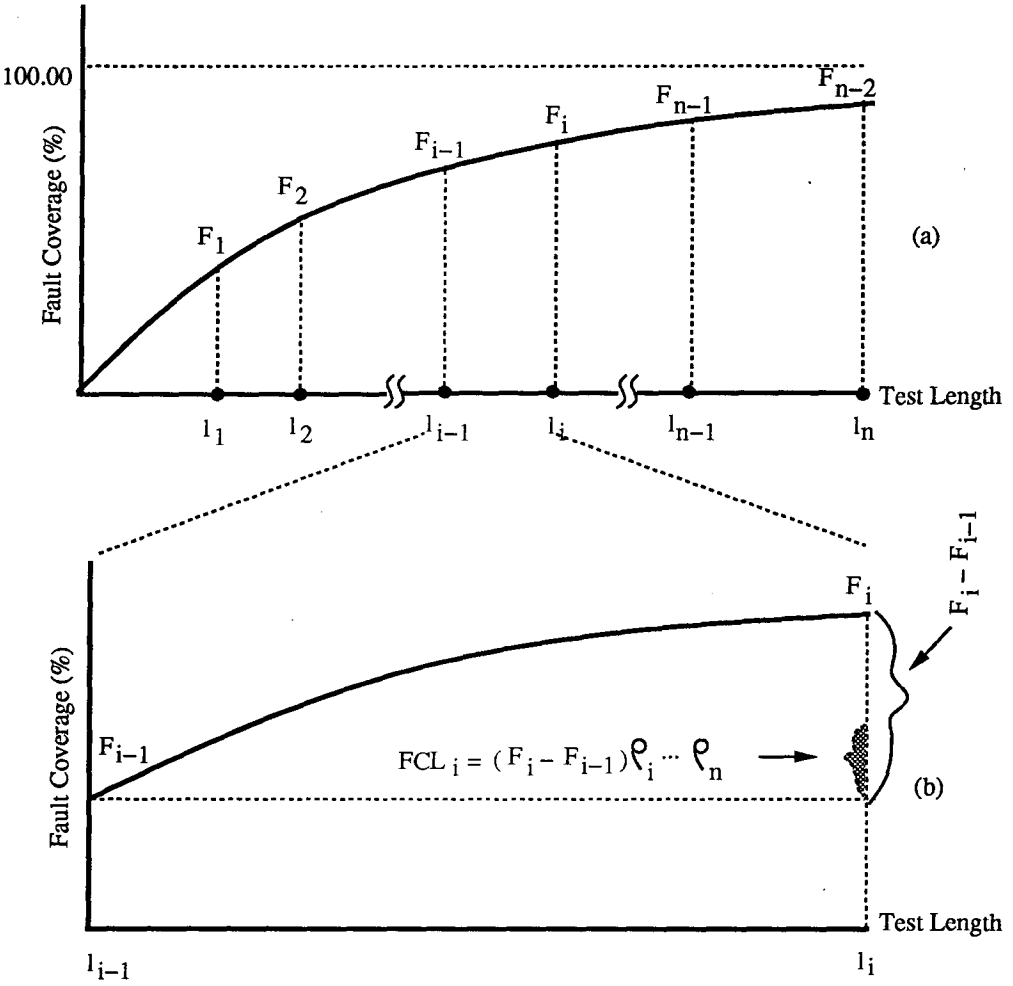



Figure 2.6: (a) Solid curve represents the fault coverage before data compaction. $F_1$ denotes the fault coverage before data compaction at check point $l_1$. $F_2$ denotes the fault coverage before data compaction at check point $l_2$, and so on. (b) $FCL_i$ is the fault coverage loss on $(F_i - F_{i-1})$ segment due to the $n$ signatures. The total fault signature loss $FCL = \sum_{i=1}^{n} FCL_i$

Therefore, the total fault coverage loss, $FCL$, with $n$ signatures is:

$$\begin{aligned} FCL &= \sum_{i=1}^{n} FCL_i \\ &= \sum_{i=1}^{n} (F_i - F_{i-1}) \prod_{j=i}^{n} \rho_j. \end{aligned} \tag{2.17}$$

Since the final fault coverage before data compaction is $F_n$, the fault coverage, $FC$, for MSA is:

$$\begin{aligned} FC &= F_n - FCL \\ &= F_n - \sum_{i=1}^{n} (F_i - F_{i-1}) \prod_{j=i}^{n} \rho_j. \end{aligned} \tag{2.18}$$

Eqn.2.18 gives a prediction for the fault coverage with multiple signature analysis. This model is solely based on the knowledge of the fault coverage before data compaction and the estimated statistical performance of the signature analyzer. The efforts needed to calculate fault coverage before data compaction are much less than the efforts to calculate exact detection probability. Thus, Eqn.2.18 is much simpler to compute compared to the model discussed in Sec. 2.2.4. However, this simplified model is optimistic since we assume that all the faults aliased by a signature at a check point $l_i$ are redetected during later segments $[l_{j-1}, l_j]$ for $j = (i+1), (i+2), \ldots, n$. In practice, this assumption may not be true. Some faults aliased at $l_i$ may be redetected in some of the later segments, but not necessarily in all of the segments. Some faults aliased at $l_i$ may not be redetected at all. Check points are usually scheduled as $(l_i - l_{i-1}) \leq (l_{i+1} - l_i)$ for $i = 1, \ldots, n$, so as to minimize either the required fault simulation time [31] or the average test time [33], most faults detected in segment $[l_{i-1}, l_i]$ are likely to be redetected in later segments. Experimental results in Sec. 2.5.3 show that this model holds very well.

## 2.4 Signature Bits Distribution vs. Fault Coverage

As Eqn.2.18 implies, the final fault coverage achieved with checking $n$ signatures of sizes $k_1, k_2, \ldots, k_n$ depends on the scheduling of the check points $l_1, l_2, \ldots, l_n$ and the signature sizes $k_1, k_2, \ldots, k_n$, i.e.,

$$FC = f(k_1, k_2, \ldots, k_n, l_1, l_2, \ldots, l_n). \tag{2.19}$$

**Theorem 2:** The expected fault coverage loss $FCL$ for a $(k, n; k_1, \ldots, k_n, l_1, \ldots, l_n)$ MSA scheme is greater than that for the (k,1;k;l) SSA scheme.

**Proof:** Assume that at the check points $l_1, l_2, \ldots, L_n$, the corresponding fault coverage before data compaction is known to be $F_1, F_2, \ldots, F_n$, with $F_0 \equiv 0$.

According to Eqn. 2.17, the fault coverage loss, $FCL^*$, after checking a single signature at the end of test is:

$$FCL^* = F_n \prod_{i=1}^{n} \rho_i, \tag{2.20}$$

and, the fault coverage loss, $FCL$, for checking $n$ signatures is:

$$\begin{aligned} FCL &= \sum_{i=1}^{n} FCL_i \\ &= \sum_{i=1}^{n} (F_i - F_{i-1}) \prod_{j=i}^{n} \rho_j. \end{aligned} \tag{2.21}$$

Hence,

$$\begin{aligned} FCL - FCL^* &= (\sum_{i=1}^{n} (F_i - F_{i-1}) \prod_{j=i}^{n} \rho_j) - (F_n \prod_{i=1}^{n} \rho_i) \\ &= \sum_{i=1}^{n} (F_i - F_{i-1}) \prod_{j=i}^{n} \rho_j - \sum_{i=1}^{n} (F_i - F_{i-1}) \prod_{j=1}^{n} \rho_j \\ &= \sum_{i=1}^{n} (F_i - F_{i-1}) (\prod_{j=i}^{n} \rho_j - \prod_{j=1}^{n} \rho_j) \\ &= \sum_{i=2}^{n} (F_i - F_{i-1}) \prod_{j=i}^{n} \rho_j (1 - \prod_{j=1}^{i-1} \rho_j) + F_1 \prod_{j=1}^{n} \rho_j \geq 0. \end{aligned} \tag{2.22}$$

The equality holds when $n = 1$. □

Theorem 2 reveals that when checking multiple signatures at distinct check points to either calculate the exact fault coverage of multiple signature analysis [31], or to reduce the average test time [33], fault coverage is sacrificed if the same amount of hardware overhead, i.e., measured in number of signature bits, is used.

**Theorem 3:** Given that the $n$ check points are predetermined for a $(k, n; k, k_1, \ldots, k_n; l_1, l_2, \ldots, l_n)$ MSA scheme, $FCL$ for the MSA scheme is minimized when $k_1 = k_2 = \ldots = k_{n-1} = 1$ and $k_n = k - n + 1$.

**Proof:** By Eqn. 2.17, we have,

$$\begin{aligned} FCL &= \sum_{i=1}^{n} FCL_i \\ &= \sum_{i=1}^{n} (F_i - F_{i-1}) \prod_{j=i}^{n} \rho_j, \end{aligned} \tag{2.23}$$

where $\rho_j = 2^{-k_j}$, $j = 1, 2, \ldots, n$.

Let

$$c_i = F_i - F_{i-1}, \qquad i = 1, \ldots, n. \tag{2.24}$$

Since fault coverage is monotonic in test length, $c_i \geq 0$ for $i = 1, \ldots, n$. Also since the check points are predetermined, $c_i$ is fixed for $i = 1, \ldots, n$.

Let

$$\begin{aligned} t_i &= \prod_{j=i}^{n} \rho_j \\ &= \prod_{j=i}^{n} 2^{-k_j} \\ &= 2^{-\sum_{j=i}^{n} k_j}, \qquad i = 1, 2, \ldots, n. \end{aligned} \tag{2.25}$$

By Eqn. 2.24 and 2.25, we have:

$$FCL = \sum_{i=1}^{n} c_i t_i. \tag{2.26}$$

Since $k_j \geq 1$ for $j = 1, \ldots, n$,

$$\sum_{j=1}^{i-1} k_j \geq \sum_{j=1}^{i-1} 1 = i - 1. \tag{2.27}$$

Also, we have,

$$\begin{aligned} k &= \sum_{j=1}^{n} k_j \\ &= \sum_{j=1}^{i-1} k_j + \sum_{j=i}^{n} k_j \\ &\geq i - 1 + \sum_{j=i}^{n} k_j. \end{aligned} \tag{2.28}$$

Therefore,

$$\sum_{j=i}^{n} k_j \leq k - i + 1, \qquad i = 1, \ldots, n. \tag{2.29}$$

The equality holds when $k_1 = k_2 = \ldots = k_{i-1} = 1$.

$$\begin{aligned} t_i &= 2^{-\sum_{j=i}^{n} k_j} \\ &\geq 2^{-(k-i+1)}, \qquad i = 1, \ldots, n. \end{aligned} \tag{2.30}$$

Since $c_i \geq 0$,

$$c_i t_i \geq c_i 2^{-(k-i+1)}, \qquad i = 1, \ldots, n. \tag{2.31}$$

Therefore,

$$\sum_{i=1}^{n} c_i t_i \geq \sum_{i=1}^{n} c_i 2^{-(k-i+1)}. \tag{2.32}$$

Equality holds when $k_1 = \ldots = k_{n-1} = 1$.

| $D_n(k_1, k_2, \ldots, k_n)$ | $\#ofDFF$ | $FCL$ |
|---|---|---|
| $D_n(1, 1, \ldots, k-n+1)$ | $2k-n+2$ | $\sum_{i=1}^{n}(F_i - F_{i-1})\prod_{j=i}^{n}\rho_j$ |
| $D_1(k)$ | $2k$ | $2^{-k}$ |

Table 2.4: Hardware Cost Analysis for Multiple Signature Schemes

Since $k_1 + k_2 + \ldots + k_n = k$, $FCL$ is minimized when $k_1 = k_2 = \ldots = k_{n-1} = 1$, and $k_n = k - n + 1$. □

Theorem 3 gives the optimal partition for a total number of $k$ signature bits when $n$ signatures are checked, assuming that the check points are predetermined and fixed.

An example of the application of Theorem 3 follows. Let $\#ofDFF$ denote the number of memory elements (flip-flops) required to implement the signature analyzers and to store the FFSs. Checking a total of $k$ bits of signature requires storing $k$ bits of information. This requires $k$ latches or flip flops. Table 2.4 reports the fault coverage and hardware requirements for the above optimal scheme and the single signature scheme.

From Table 2.4, we see that the fault coverage loss (FCL) of a $(k, n; 1, 1, \ldots, 1, k-n+1; l_1, \ldots, l_n)$ MSA scheme is greater than that of a $(k, 1; k; l)$ SSA scheme, where $\rho_i = 2^{-k_i}$. However, the hardware cost of the former, measured in terms of the number of memory elements, is less than the hardware cost of the latter as long as $n > 2$. For example, if $k = 16$ and $n = 5$, the hardware cost of the $(16, 5; 1, 1, 1, 1, 12; l_1, \ldots, l_5)$ MSA scheme is 29 D flip-flops, while the hardware cost of the $(16, 1; 16; l_5)$ SSA scheme is 32 D flip-flops. The hardware saving is 9%. Hardware saving is achieved at the expense of fault coverage loss if the number of checked signatures is increased. If we embed the single-bit signature analyzer in the $k-n+1$ bit signature analyzer, then we can further decrease the number of D flip-flops used by the $(k, n; 1, 1, \ldots, 1, k-n+1; l_1, \ldots, l_n)$ MSA scheme by 1, and the hardware cost becomes $2k - n + 1$. As long as $n > 1$, the hardware cost of the

$(k, n; 1, 1, \ldots, 1, k-n+1; l_1, \ldots, l_n)$ MSA scheme is smaller than that of the $(k, 1; k; l_n)$ SSA scheme.

Fig. 2.7 shows a single-bit signature analyzer embedded in a $k-n+1$ signature analyzer. Fig. 2.7$(a)$ and Fig. 2.7$(b)$ show the 1-bit signature analyzer and the $k$-bit signature analyzer, respectively. Fig. 2.7$(c)$ shows a $k$-bit signature analyzer where any D flip-flop (DFF) can be used as a 1-bit signature analyzer. The second DFF is arbitrarily chosen as the embedded single-bit signature analyzer. It is easy to show that the asymptotic aliasing probability of an embedded 1-bit signature analyzer is the same as the 1-bit signature analyzer shown in Fig. 2.7 (a).

**Theorem 4:** Given a $(k, n; k_1, \ldots, k_n; l_1, \ldots, l_n)$ MSA scheme, the fault coverage loss at the end of the test $FCL$ is maximized when $k_1 = k - n + 1$, $k_2 = k_3 = \ldots = k_n = 1$.

**Proof:** By Eqn. 2.17, we have,

$$\begin{aligned} FCL &= \sum_{i=1}^{n} FCL_i \\ &= \sum_{i=1}^{n} (F_i - F_{i-1}) \prod_{j=i}^{n} \rho_j, \end{aligned} \tag{2.33}$$

where $\rho_j = 2^{-k_j}$, $j = 1, 2, \ldots, n$.

Let

$$c_i = F_i - F_{i-1}, \qquad i = 1, \ldots, n. \tag{2.34}$$

Since fault coverage is monotonic in test length, $c_i \geq 0$ for $i = 1, \ldots, n$. Also since the check points are predetermined, $c_i$ is fixed for $i = 1, \ldots, n$.

Let

$$t_i = \prod_{j=i}^{n} \rho_j$$

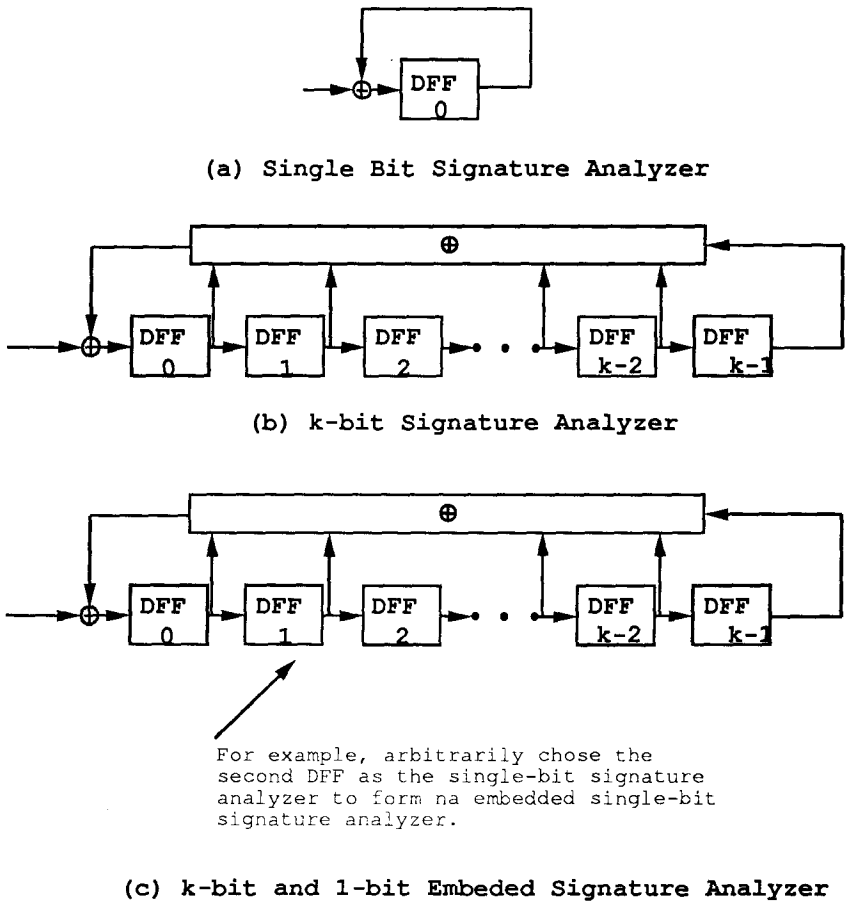



Figure 2.7: This figure shows the hardware costs to implement a $(k+n-1, n; 1, 1, \ldots, k; l_1, \ldots, l_n)$ MSA scheme. (a) is the block diagram of a single bit signature analyzer. (b) is the block diagram of a $k$-bit signature analyzer. (c) shows how to embed a single bit signature analyzer into a $k$-bit signature analyzer. $DFF$ is a D flip-flop.

$$= \prod_{j=i}^{n} 2^{-k_j}$$
$$= 2^{-\sum_{j=i}^{n} k_j}, \qquad i = 1, 2, \ldots, n. \tag{2.35}$$

By Eqn. 2.34 and 2.35, we have:

$$FCL = \sum_{i=1}^{n} c_i t_i. \tag{2.36}$$

Since $k_j \geq 1$ for $j = 1, \ldots, n$,

$$\sum_{j=i}^{n} k_j \geq \sum_{j=i}^{n} 1 = n - i + 1. \tag{2.37}$$

The equality holds when $k_i = k_{i+1} = \ldots = k_n = 1$.

Therefore,

$$t_i = 2^{-\sum_{j=i}^{n} k_j} \leq 2^{-(n-i+1)}, \qquad i = 2, \ldots, n. \tag{2.38}$$

Since $c_i \geq 0$,

$$c_i t_i \leq c_i 2^{-(n-i+1)}, i = 2, 3, \ldots, n. \tag{2.39}$$

Therefore,

$$FCL = F_1 2^{-k_1} + \sum_{i=2}^{n} c_i t_i \leq \sum_{i=2}^{n} c_i 2^{-(n-i+1)} + F_1 2^{-k_1}. \tag{2.40}$$

Equality holds when $k_2 = \ldots = k_n = 1$.

Since $k_1 + k_2 + \ldots + k_n = k$, $FCL = \sum_{i=1}^{n} c_i t_i$ is maximized when $k_1 = k - n + 1$, $k_2 = \ldots = k_n = 1$. □

Theorem 4 states that if $k$ bits of signature are to be distributed among $n$ signatures, then the worst choice of distributions is where the first signature is of size $k - n + 1$ and the remaining $n - 1$ signatures are of size 1. This would yield minimal expected fault coverage.

| Circuit Name | Circuit Function | Total Gates | Inputs Lines | Output Faults | Stuck-at Faults |
|---|---|---|---|---|---|
| c432 | Priority Decoder | 160(18 EXOR) | 36 | 7 | 864 |
| c499 | ECAT | 202(104 EXOR) | 41 | 32 | 998 |
| c880 | ALU and Control | 383 | 60 | 26 | 1760 |
| c1355 | ECAT | 546 | 41 | 32 | 2710 |
| c1908 | ECAT | 880 | 33 | 25 | 3168 |
| c2670 | ALU and Control | 1193 | 233 | 140 | 4448 |
| c3540 | ALU and Control | 1669 | 50 | 22 | 6188 |
| c5315 | ALU and Selector | 2307 | 178 | 123 | 9400 |
| c6288 | 16-bit Multiplier | 2406 | 32 | 32 | 12576 |
| c7552 | ALU and Control | 3512 | 207 | 108 | 13048 |

Table 2.5: ISCAS'85 Benchmark Circuit Characteristics

## 2.5 Experimental Results

Most of the fault simulators that have been implemented use the stuck-at fault model [55] [35] [36]. In the experiments presented here, we only consider single stuck-at faults as well. A single stuck-at fault assumes a circuit failure corresponding to one line of the circuit being permanently fixed at the logic value 0 or 1. A circuit with $p$ lines thus has $2p$ possible single stuck-at faults. In our experiments, no *fault collapsing* [6] is done.

### 2.5.1 ISCAS'85 Benchmark Circuits

The combinational circuits used in our experiments are from the ISCAS'85 benchmark circuits [7]. The characteristics of the ten benchmark circuits are given in Table 2.5. The stuck-at faults in the last column of Table 2.5 is the complete set of single stuck-at faults in the circuit. No equivalent fault collapsing is done. Circuit c499 and c1355 are functionally equivalent. c499 uses XOR gates, while c1355 uses 4-NAND gates to realized the XOR function.

### 2.5.2 Experimental Cases

In this section, fourteen experiment cases are defined. Experiments on the fourteen experiment cases were conducted and the results are reported in Sec. 2.5.3. The fourteen experimental cases are:

1. $(16, 16; 1, \ldots, 1; 2048)$ MSA
2. $(16, 8; 2, \ldots, 2; 2048)$ MSA
3. $(16, 4; 4, \ldots, 4; 2048)$ MSA
4. $(16, 2; 8, 8; 2048)$ MSA
5. $(16, 8; 1, \ldots, 1, 9; 2048)$ MSA
6. $(16, 4; 1, 1, 1, 13; 2048)$ MSA
7. $(16, 2; 1, 15; 2048)$ MSA
8. $(16, 8; 9, 1, \ldots, 1; 2048)$ MSA
9. $(16, 4; 13, 1, \ldots, 1; 2048)$ MSA
10. $(16, 2; 15, 1; 2048)$ MSA
11. $(16, 16; 1, \ldots, 1; 2048; O)$ MSA
12. $(16, 8; 2, \ldots, 2; 2048; O)$ MSA
13. $(16, 4; 4, \ldots, 4; 2048; O)$ MSA
14. $(16, 2; 8, 8; 2048; O)$ MSA

### 2.5.3 Results

In this section, the experimental results on the ten ISCAS'85 benchmark circuits for the fourteen experimental cases are presented. The experimental results are listed in Tables 2.6 - 2.9, Tables 2.10 - 2.12 and Tables 2.13- 2.16. The definitions of the symbols used in those tables are as follows:

- $FC_{simp}$ denotes the fault coverage predicted by the simplified fault coverage model for $(k, n; k_1, \ldots, k_n; l)$ MSA.
- $FC_{bha84}$ denotes the fault coverage predicted by the model presented in [5] for $(k, n; k_1, \ldots, k_n; l)$ MSA.
- $FC_{ms}$ denotes the exact fault coverage determined by fault simulation for $(k, n;\ k_1, \ldots, k_n;\ l)$ MSA.
- $FC_{ms1k}$ denotes the exact fault coverage determined by fault simulation for $(k, n; 1, \ldots, k-n+1; l)$ MSA.
- $FC_{msk1}$ denotes the exact fault coverage determined by fault simulation for $(k, n; k-n+1, 1, \ldots, 1; l)$ MSA.
- $FC_{mso1}$ denotes the exact fault coverage determined by fault simulation for $(k, n; k_1, \ldots, k_n; l; O; aliasing)$ MSA.
- $FC_{mso2}$ denotes the exact fault coverage determined by fault simulation for $(k, n; k_1, \ldots, k_n; l; O; no\ aliasing)$ MSA.
- $FC_{ss}$ denotes the fault coverage obtained by fault simulation for $(16, 1; 16; l)$ SSA.
- $FC_{nc}$ denotes the exact fault coverage before compaction.

- $Err_{simp}$ is the absolute value of the difference between $FC_{simp}$ and $FC_{ms}$, i.e., $Err_{simp} = \mid FC_{simp} - FC_{ms} \mid$.
- $Err_{bha84}$ is the absolute value of the difference between $FC_{bha84}$ and $FC_{ms}$, i.e., $Err_{bha84} = \mid FC_{bha84} - FC_{ms} \mid$.
- $AveErr_{simp}$ is the average of $Err_{simp}$ over the ten benchmark circuits.
- $AveErr_{bha84}$ is the average of $Err_{bha84}$ over the ten benchmark circuits.

All the fault coverages obtained by fault simulation reported in this chapter are the average of three trials. In the figures presented in this chapter, wherever it is appropriate, $c1$ denotes $c432$, $c2$ denotes $c499$, $c3$ denotes $c880$, $c4$ denotes $c1355$, $c5$ denotes $c1908$, $c6$ denotes $c2670$, $c7$ denotes $c3540$, $c8$ denotes $c5315$, $c9$ denotes $c6288$, and $c10$ denotes $c7552$.

Table 2.6 reports the fault coverage predicted by various models as well as the fault coverage obtained by fault simulation for case 1. Table 2.7 presents the experimental results for case 2, Table 2.8 for case 3, and Table 2.9 for case 4.

From the $Err_{simp}$ and $Err_{bha84}$ columns of Tables 2.6-2.9, the average difference between $Err_{simp}$ and $Err_{bha84}$ is 0.0684% for $(16, 16; 1, \ldots, 1; 2048)$ MSA, the average difference between $Err_{simp}$ and $Err_{bha84}$ is 0.0469%, for $(16, 8; 2, \ldots, 2; 2048)$ MSA, the average difference between $Err_{simp}$ and $Err_{bha84}$ is 0.0060%, for $(16, 4; 4, \ldots, 4; 2048)$ MSA, the average difference between $Err_{simp}$ and $Err_{bha84}$ is 0.0013%, for $(16, 2; 8, 8; 2048)$ MSA. Although for small circuits, the difference is not significant, for large circuits, e.g., circuits which have stuck-at faults over $10,000$, the difference can be significant. Figs. 2.13-2.14 present $Err_{simp}$ and $Err_{bha84}$ for the ten benchmark circuits graphically. Also note that as $n$ increases, the model presented here gives better prediction than that model in [5].

Table 2.10 presents the experimental results for case 5 and 8. Table 2.11 reports the experimental results for case 6 and 9. Table 2.12 gives the experimental results for case 7 and 10. Tables 2.13-2.16 report the experimental results for case 11, 12, 13 and 14, respectively.

Comparing the fault coverage reported in $FC_{ms}$ columns in Tables 2.6-2.9, $FC_{msk1}$ columns in Tables 2.10-2.12, $FC_{mso1}$ and $FC_{mso2}$ columns in Tables 2.13-2.16 with the fault coverages reported in $FC_{ss}$ column in Table 2.6, we can see that the fault coverage for $(k, n; k_1, \ldots, k_n; l)$ MSA is less than that for $(k, 1; k; l)$ SSA, which supports the claim in Theorem 2.

Comparing the fault coverages in $FC_{ms1k}$ column in Tables 2.10-2.12 with that in $FC_{ms}$ column in Tables 2.6-2.9, $FC_{msk1}$ column in Tables 2.10-2.12, $FC_{mso1}$ and $FC_{mso2}$ columns in Tables 2.13-2.16, we can see that the fault coverage of $(k, n; 1, \ldots, 1, k-n+1; l)$ MSA are better than that obtained by other various MSA, which supports the claim in Theorem 3.

Comparing the fault coverages in $FC_{msk1}$ column in Tables 2.10-2.12 with that in $FC_{ms}$ column in Tables 2.6-2.9, $FC_{ms1k}$ column in Tables 2.10-2.12, $FC_{mso1}$ and $FC_{mso2}$ columns in Tables 2.13-2.16, we can see that the fault coverage of $(k, n; k-n+1, 1, \ldots; l)$ MSA are less than that obtained by other various MSA, which supports the claim in Theorem 4.

Comparing the fault coverages in the $FC_{mso1}$ columns in Tables 2.13-2.16 with that in the $FC_{mso2}$ columns in Tables 2.13-2.16, and the $FC_{ms}$ columns in Tables 2.6-2.9, we can see that the fault coverage for a MSA is the function of signature scheduling.

Figs. 2.15 and 2.16 presents $FC_{ms}$ and $FC_{ss}$ for the four experimental cases. From the figures we can see that given $k$, in 24 out of 40 trials, $FC_{ms}$ is less than $FC_{ss}$. In 5 out

| Circuit Name | $FC_{simp}$ (%) | $FC_{bha84}$ (%) | $FC_{ms}$ (%) | $FC_{ss}$ (%) | $FC_{nc}$ (%) | $Err_{simp}$ (%) | $Err_{bha84}$ (%) |
|---|---|---|---|---|---|---|---|
| c432 | 98.8409 | 98.8411 | 98.6111 | 98.8426 | 98.8426 | 0.2298 | 0.2300 |
| c499 | 99.1958 | 99.1969 | 96.3929 | 99.1984 | 99.1984 | 2.8029 | 2.8040 |
| c880 | 97.3353 | 97.5932 | 97.3864 | 97.5947 | 97.5947 | 0.0511 | 0.2068 |
| c1355 | 99.3515 | 99.3958 | 97.8844 | 99.3973 | 99.3973 | 1.4671 | 1.5114 |
| c1908 | 98.1052 | 98.4413 | 95.9701 | 98.4007 | 98.4428 | 2.1351 | 2.4712 |
| c2670 | 82.2073 | 82.2079 | 81.3699 | 82.2092 | 82.2092 | 0.8374 | 0.8380 |
| c3540 | 94.8512 | 94.9458 | 94.3816 | 94.9095 | 94.9472 | 0.4696 | 0.5642 |
| c5315 | 99.2545 | 99.2645 | 99.1986 | 99.2660 | 99.2660 | 0.0559 | 0.0659 |
| c6288 | 99.4578 | 99.4578 | 99.4407 | 99.4593 | 99.4593 | 0.0171 | 0.0171 |
| c7552 | 93.0237 | 93.0653 | 92.8342 | 93.0667 | 93.0667 | 0.1895 | 0.2311 |
| $AveErr_{simp} = 0.8256\%$, $AveErr_{bha84} = 0.8940\%$ | | | | | | | |

Table 2.6: Fault Coverages for $(16, 16; 1, \ldots, 1; 2048)$ MSA

| Circuit Name | $FC_{simp}$ (%) | $FC_{bha84}$ (%) | $FC_{ms}$ (%) | $FC_{ss}$ (%) | $FC_{nc}$ (%) | $Err_{simp}$ (%) | $Err_{bha84}$ (%) |
|---|---|---|---|---|---|---|---|
| c432 | 98.8410 | 98.8411 | 98.8040 | 98.8426 | 98.8426 | 0.0370 | 0.0371 |
| c499 | 99.1963 | 99.1969 | 99.0648 | 99.1984 | 99.1984 | 0.1315 | 0.1321 |
| c880 | 97.4635 | 97.5932 | 97.3863 | 97.5947 | 97.5947 | 0.0772 | 0.2069 |
| c1355 | 99.3540 | 99.3958 | 99.1882 | 99.3973 | 99.3973 | 0.1658 | 0.2076 |
| c1908 | 98.2257 | 98.4413 | 97.7378 | 98.4007 | 98.4428 | 0.4879 | 0.7035 |
| c2670 | 82.2074 | 82.2079 | 82.0669 | 82.2092 | 82.2092 | 0.1405 | 0.1410 |
| c3540 | 94.8831 | 94.9458 | 91.5697 | 94.9095 | 94.9472 | 3.3134 | 3.3761 |
| c5315 | 99.2581 | 99.2645 | 99.2660 | 99.2660 | 99.2660 | 0.0079 | 0.0015 |
| c6288 | 99.4578 | 99.4578 | 99.4566 | 99.4593 | 99.4593 | 0.0012 | 0.0012 |
| c7552 | 93.0414 | 93.0653 | 93.0257 | 93.0667 | 93.0667 | 0.0157 | 0.0396 |
| $AveErr_{simp} = 0.4378\%$, $AveErr_{bha84} = 0.4847\%$ | | | | | | | |

Table 2.7: Fault Coverages for $(16, 8; 2, \ldots, 2; 2048)$ MSA

| Circuit Name | $FC_{simp}$ (%) | $FC_{bha84}$ (%) | $FC_{ms}$ (%) | $FC_{ss}$ (%) | $FC_{nc}$ (%) | $Err_{simp}$ (%) | $Err_{bha84}$ (%) |
|---|---|---|---|---|---|---|---|
| c432 | 98.8411 | 98.8411 | 98.8426 | 98.8426 | 98.8426 | 0.0015 | 0.0015 |
| c499 | 99.1965 | 99.1969 | 99.1984 | 99.1984 | 99.1984 | 0.0019 | 0.0015 |
| c880 | 97.5528 | 97.5932 | 97.5947 | 97.5947 | 97.5947 | 0.0419 | 0.0015 |
| c1355 | 99.3716 | 99.3958 | 99.3973 | 99.3973 | 99.3973 | 0.0257 | 0.0015 |
| c1908 | 98.3199 | 98.4413 | 98.3270 | 98.4007 | 98.4428 | 0.0071 | 0.1142 |
| c2670 | 82.2078 | 82.2079 | 82.1493 | 82.2092 | 82.2092 | 0.0585 | 0.0586 |
| c3540 | 94.9170 | 94.9458 | 93.9190 | 94.9095 | 94.9472 | 0.9980 | 1.0268 |
| c5315 | 99.2610 | 99.2645 | 99.2553 | 99.2660 | 99.2660 | 0.0057 | 0.0092 |
| c6288 | 99.4578 | 99.4578 | 99.4593 | 99.4593 | 99.4593 | 0.0015 | 0.0015 |
| c7552 | 93.0568 | 93.0653 | 93.0667 | 93.0667 | 93.0667 | 0.0099 | 0.0014 |
| $AveErr_{simp} = 0.1152\%$, $AveErr_{bha84} = 0.1218\%$ | | | | | | | |

Table 2.8: Fault Coverages for $(16, 4; 4, \ldots, 4; 2048)$ MSA

| Circuit Name | $FC_{simp}$ (%) | $FC_{bha84}$ (%) | $FC_{ms}$ (%) | $FC_{ss}$ (%) | $FC_{nc}$ (%) | $Err_{simp}$ (%) | $Err_{bha84}$ (%) |
|---|---|---|---|---|---|---|---|
| c432 | 98.8411 | 98.8411 | 98.8426 | 98.8426 | 98.8426 | 0.0015 | 0.0015 |
| c499 | 99.1969 | 99.1969 | 99.1984 | 99.1984 | 99.1984 | 0.0015 | 0.0015 |
| c880 | 97.5898 | 97.5932 | 97.5947 | 97.5947 | 97.5947 | 0.0049 | 0.0015 |
| c1355 | 99.3903 | 99.3958 | 99.3973 | 99.3973 | 99.3973 | 0.0070 | 0.0015 |
| c1908 | 98.4240 | 98.4413 | 98.2744 | 98.4007 | 98.4428 | 0.1496 | 0.1669 |
| c2670 | 82.2079 | 82.2079 | 82.2092 | 82.2092 | 82.2092 | 0.0013 | 0.0013 |
| c3540 | 94.9402 | 94.9458 | 94.9095 | 94.9095 | 94.9472 | 0.0307 | 0.0363 |
| c5315 | 99.2637 | 99.2645 | 99.2589 | 99.2660 | 99.2660 | 0.0048 | 0.0056 |
| c6288 | 99.4578 | 99.4578 | 99.4593 | 99.4593 | 99.4593 | 0.0015 | 0.0015 |
| c7552 | 93.0639 | 93.0653 | 93.0667 | 93.0667 | 93.0667 | 0.0028 | 0.0014 |
| $AveErr_{simp} = 0.0206\%$, $AveErr_{bha84} = 0.0219\%$ | | | | | | | |

Table 2.9: Fault Coverages for $(16, 2; 8, 8; 2048)$ MSA

| Circuit Name | $FC_{ms1k}$ (%) | $FC_{msk1}$ (%) |
|---|---|---|
| 432 | 98.8426 | 97.8781 |
| 499 | 99.1984 | 97.5284 |
| 880 | 97.5947 | 96.6667 |
| 1355 | 99.3973 | 98.4379 |
| 1908 | 98.4007 | 96.9171 |
| 2670 | 82.2092 | 81.4299 |
| 3540 | 94.9095 | 94.3870 |
| 5315 | 99.2660 | 99.0461 |
| 6288 | 99.4593 | 99.2393 |
| 7552 | 93.0667 | 92.7039 |

Table 2.10: Fault Coverage Comparison for $(16, 8; 1, \ldots, 1, 9; 2048)$ MSA and $(16, 8; 9, 1, \ldots, 1; 2048)$ MSA

| Circuit Name | $FC_{ms1k}$ (%) | $FC_{msk1}$ (%) |
|---|---|---|
| 432 | 98.8426 | 95.5633 |
| 499 | 99.1984 | 95.5244 |
| 880 | 97.5947 | 94.1667 |
| 1355 | 99.3973 | 96.1255 |
| 1908 | 98.4007 | 94.7075 |
| 2670 | 82.2092 | 78.6271 |
| 3540 | 94.9095 | 91.8229 |
| 5315 | 99.2660 | 96.3404 |
| 6288 | 99.4593 | 96.1965 |
| 7552 | 93.0667 | 89.9908 |

Table 2.11: Fault Coverage Comparison for (16, 4; 1, 1, 1,13; 2048) MSA, and (16, 4; 13, 1, 1, 1; 2048) MSA

| Circuit Name | $FC_{ms1k}$ (%) | $FC_{msk1}$ (%) |
|---|---|---|
| 432 | 98.8426 | 88.3102 |
| 499 | 99.1984 | 86.5063 |
| 880 | 97.5947 | 85.1326 |
| 1355 | 99.3973 | 87.6753 |
| 1908 | 98.4007 | 85.3325 |
| 2670 | 82.2092 | 70.9158 |
| 3540 | 94.9095 | 82.9617 |
| 5315 | 99.2660 | 86.7837 |
| 6288 | 99.4593 | 86.6836 |
| 7552 | 93.0667 | 81.3381 |

Table 2.12: Fault Coverage Comparison for (16, 2;1,15; 2048) MSA and (16, 2; 15, 1; 2048) MSA

| Circuit Name | $FC_{mso1}$ (%) | $FC_{mso2}$ (%) |
|---|---|---|
| 432 | n/a | n/a |
| 499 | 96.2926 | 96.4262 |
| 880 | 96.4015 | 96.5909 |
| 1355 | 97.2448 | 97.8352 |
| 1908 | 95.6860 | 95.9806 |
| 2670 | 80.8603 | 80.8828 |
| 3540 | 93.6813 | 94.3762 |
| 5315 | 98.7624 | 99.0674 |
| 6288 | n/a | n/a |
| 7552 | 92.7601 | 92.7958 |

Table 2.13: Fault Coverages for (16,16;1,. . . , 1;2048;$O$; *aliasing*) MSA and (16,16;1,. . ., 1;2048;$O$; *no aliasing*) MSA. For $c432$ and $c6288$, no optimal schedulings are available for $n = 16$ and $l = 2048$. This will be dicussed in chapter 3.

| Circuit Name | $FC_{mso1}$ (%) | $FC_{mso2}$ (%) |
|---|---|---|
| 432 | 98.6497 | 98.6883 |
| 499 | 98.4302 | 98.4302 |
| 880 | 96.9129 | 96.9886 |
| 1355 | 98.8315 | 99.0652 |
| 1908 | 97.0749 | 97.4432 |
| 2670 | 81.9169 | 82.0069 |
| 3540 | 91.4135 | 90.5839 |
| 5315 | 98.6809 | 99.0780 |
| 6288 | n/a | n/a |
| 7552 | 92.8316 | 92.8904 |

Table 2.14: Fault Coverages for $(16, 8; 2, \ldots, 2; 2048; O; \textit{aliasing})$ MSA and $(16, 8; 2, \ldots, 2; 2048; O; \textit{no aliasing})$ MSA. For $c6288$, no optimal schedulings are available for $n = 8$ and $l = 2048$. This will be discussed in chapter 3.

| Circuit Name | $FC_{mso1}$ (%) | $FC_{mso2}$ (%) |
|---|---|---|
| 432 | 98.7269 | 98.7269 |
| 499 | 98.9980 | 99.1984 |
| 880 | 97.5379 | 97.5379 |
| 1355 | 98.8684 | 98.9299 |
| 1908 | 98.3376 | 98.3376 |
| 2670 | 82.1193 | 82.1193 |
| 3540 | 92.9379 | 92.9379 |
| 5315 | 98.8759 | 98.9575 |
| 6288 | 99.4063 | 99.4328 |
| 7552 | 92.8520 | 92.9542 |

Table 2.15: Fault Coverages for (16,4;4,4, 4,4;2048; $O$; *aliasing*) MSA and (16,4;4,4 , 4,4;2048; $O$; *no aliasing*) MSA

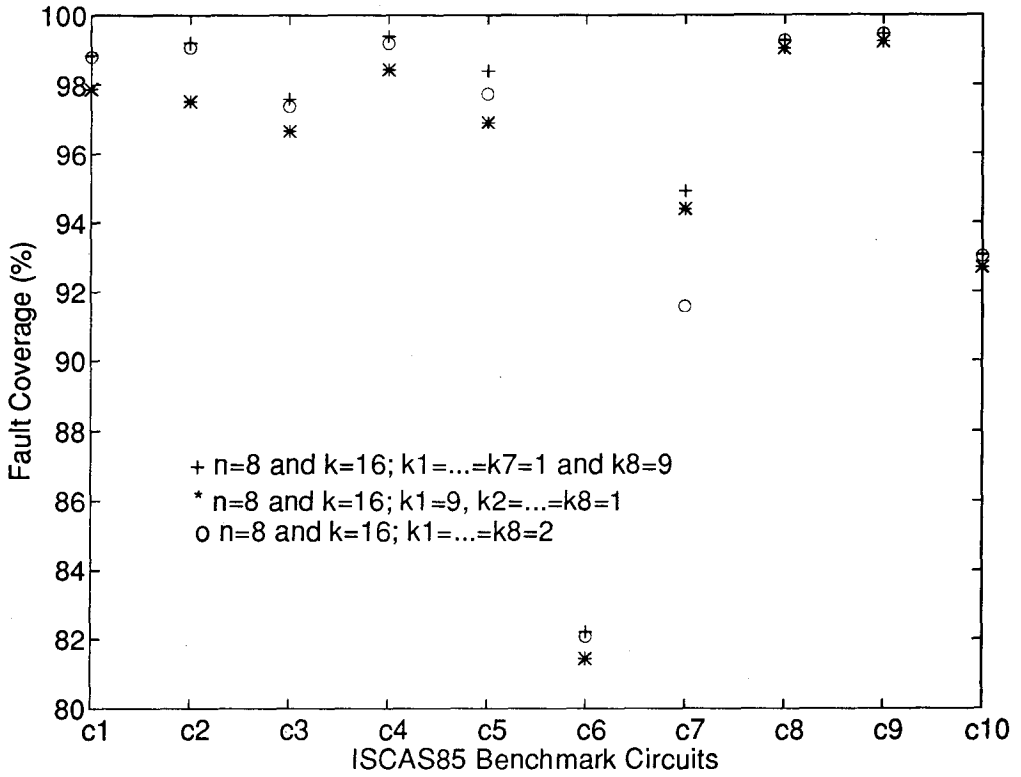



Figure 2.8: Fault Coverage Comparison for $(16,8;1,\ldots,1,9;2048)$, $(16,8;9,1,\ldots,1;2048)$, and $(16,8;2,\ldots,2;2048)$ MSA

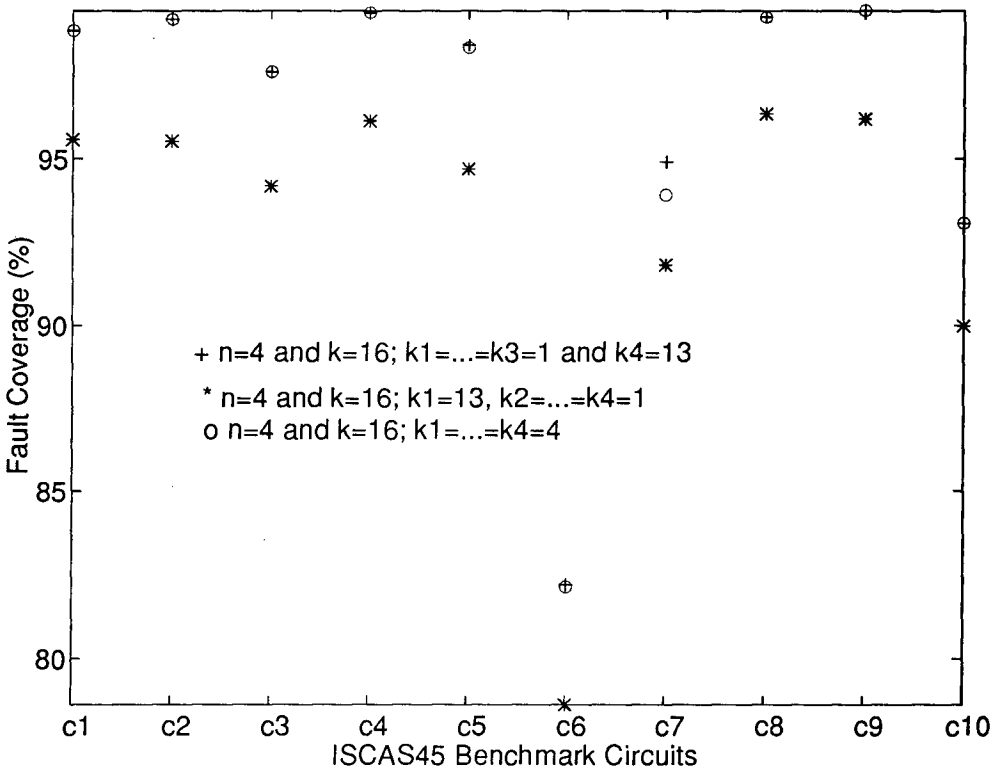



Figure 2.9: Fault Coverage Comparison for $(16, 4; 1, 1, 1, 13; 2048)$, $(16, 4; 13, 1, 1, 1; 2048)$, and $(16, 4; 4, \ldots, 4; 2048)$ MSA

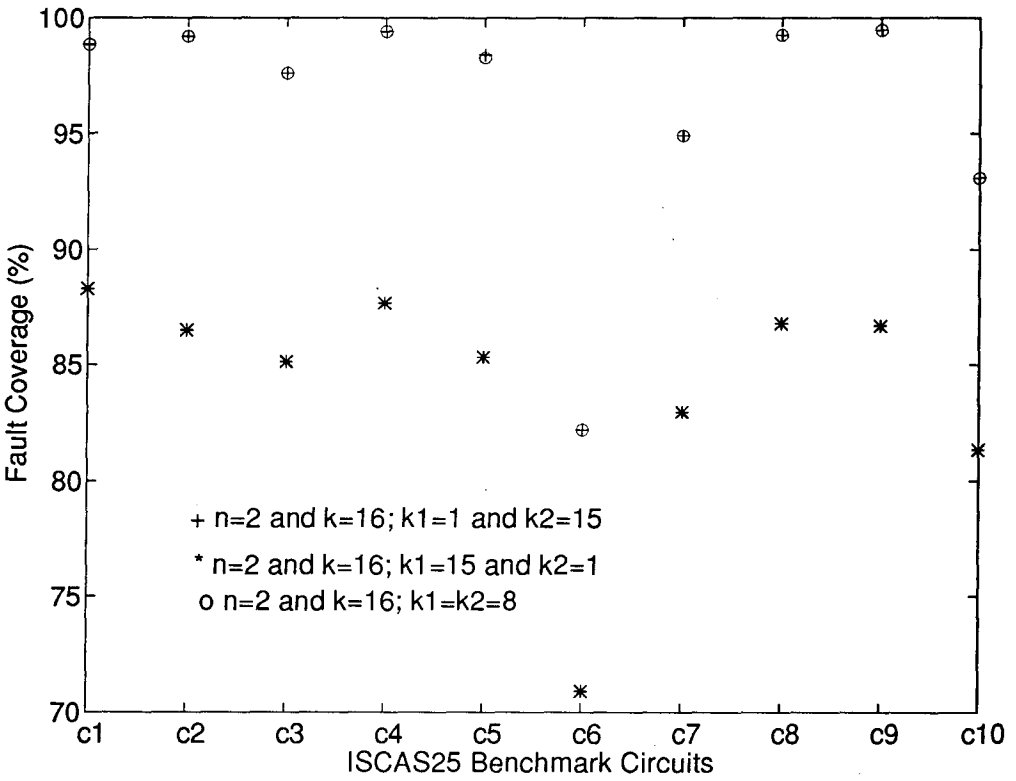



Figure 2.10: Fault Coverage Comparison for $(16, 2; 1, 15; 2048)$, $(16, 2; 15, 1; 2048)$, and $(16, 2; 8, \ldots, 8; 2048)$ MSA

| Circuit Name | $FC_{mso1}$ (%) | $FC_{mso2}$ (%) |
|---|---|---|
| 432 | 98.7655 | 98.7655 |
| 499 | 99.1650 | 99.1650 |
| 880 | 97.4432 | 97.4432 |
| 1355 | 98.3850 | 98.3850 |
| 1908 | 98.0430 | 98.0430 |
| 2670 | 82.0444 | 82.0444 |
| 3540 | 94.6563 | 94.6563 |
| 5315 | 98.9397 | 98.9397 |
| 6288 | 99.2896 | 99.2896 |
| 7552 | 92.7575 | 92.7575 |

Table 2.16: Fault Coverages for (16,2;8,8;2048; $O$; *aliasing*) MSA and (16,2;8,8;2048;$O$; *no aliasing*) MSA

of 40 trials, $FC_{ms}$ is the same as $FC_{ss}$. In 11 out of the 40 trials, $FC_{ms}$ is greater than $FC_{ss}$. When $n$ is large, $FC_{ms}$ has a better chance to be less than $FC_{ss}$.

Fig. 2.17 presents the $FC_{ms}$ for the four experimental cases. From Fig. 2.17, we can see that when $n = 16$ and $n = 8$, $FC_{ms}$ for $n = 16$ is less than that for $n = 8$ for 9 out of the ten circuits, i.e., except for circuit c3540. $FC_{ms}$ for $n = 4$ is very close to that for $n = 2$.

### 2.5.4 Fault Coverage Loss Model Accuracy

The fault coverage loss model for a MSA scheme presented in Sec. 2.3 is:

$$\begin{aligned} FCL &= \sum_{i=1}^{n} FCL_i \\ &= \sum_{i=1}^{n} (F_i - F_{i-1}) \prod_{j=i}^{n} \rho_j \\ &= \sum_{i=1}^{n} (F_i - F_{i-1}) \prod_{j=i}^{n} 2^{-k_j} \\ &= (F_n - F_{n-1}) 2^{-(k_{n-1}+k_n)} + (F_{n-1} - F_{n-2}) 2^{-(k_{n-2}+k_{n-1}+k_n)} + \ldots + \end{aligned}$$

$$(F_2 - F_1)2^{-(k_2+\ldots+k_n)} + F_1 2^{-k}. \tag{2.41}$$

As we can see from Eqn. 2.41, if $F_1 = F_2 = \ldots = F_n$, then $FCL = F_n 2^{-k}$, which is the model presented in [5]. In other words, the FCL model presented in Sec. 2.3 is sensitive to the fault coverage before compaction for the CUT, but the model presented in [5] is not. Given a signature scheduling, when the fault coverage before compaction for the CUT increases slowly, the FCL model gives a better prediction. Given a signature scheduling, when the fault coverage before compaction for the CUT increases rapidly, the FCL model and the model presented in [5] give similar predictions.

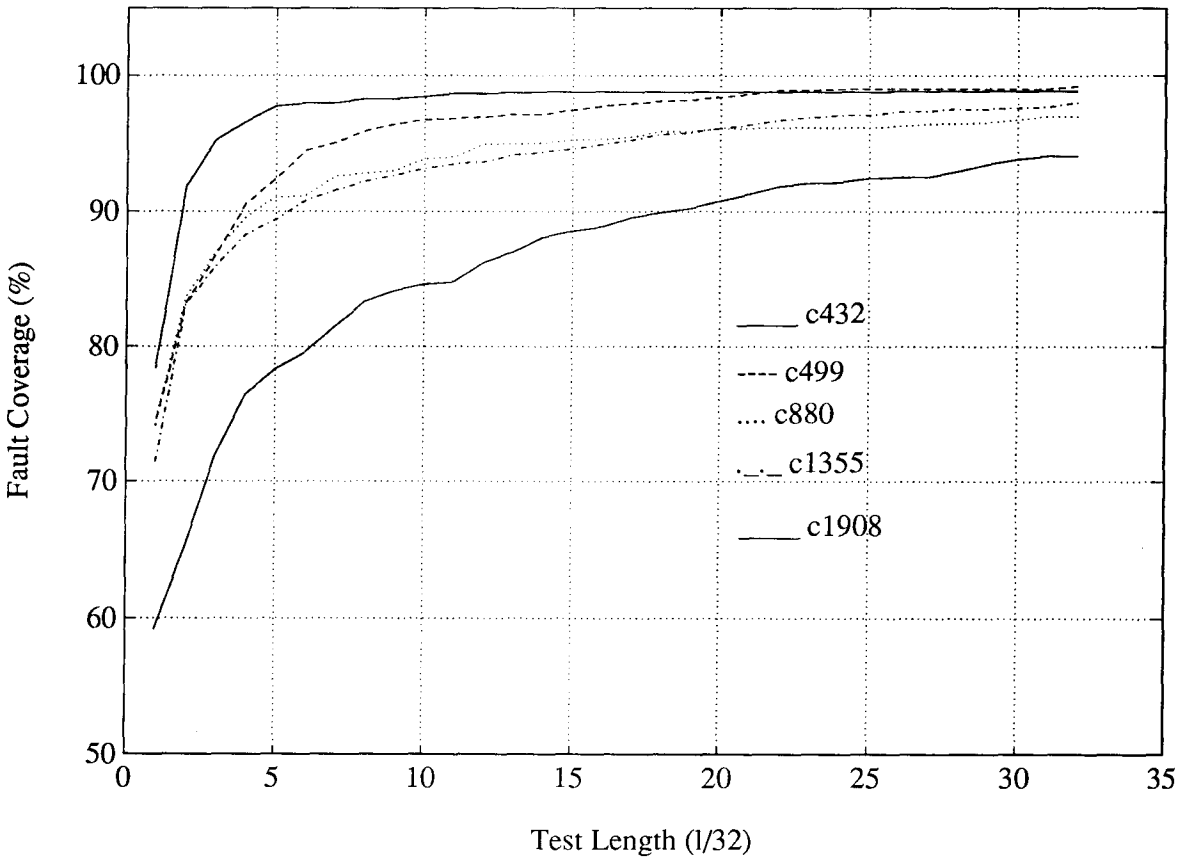



Figure 2.11: Fault Coverage Before Compaction for c432, c499, c880, c1355, and c1908

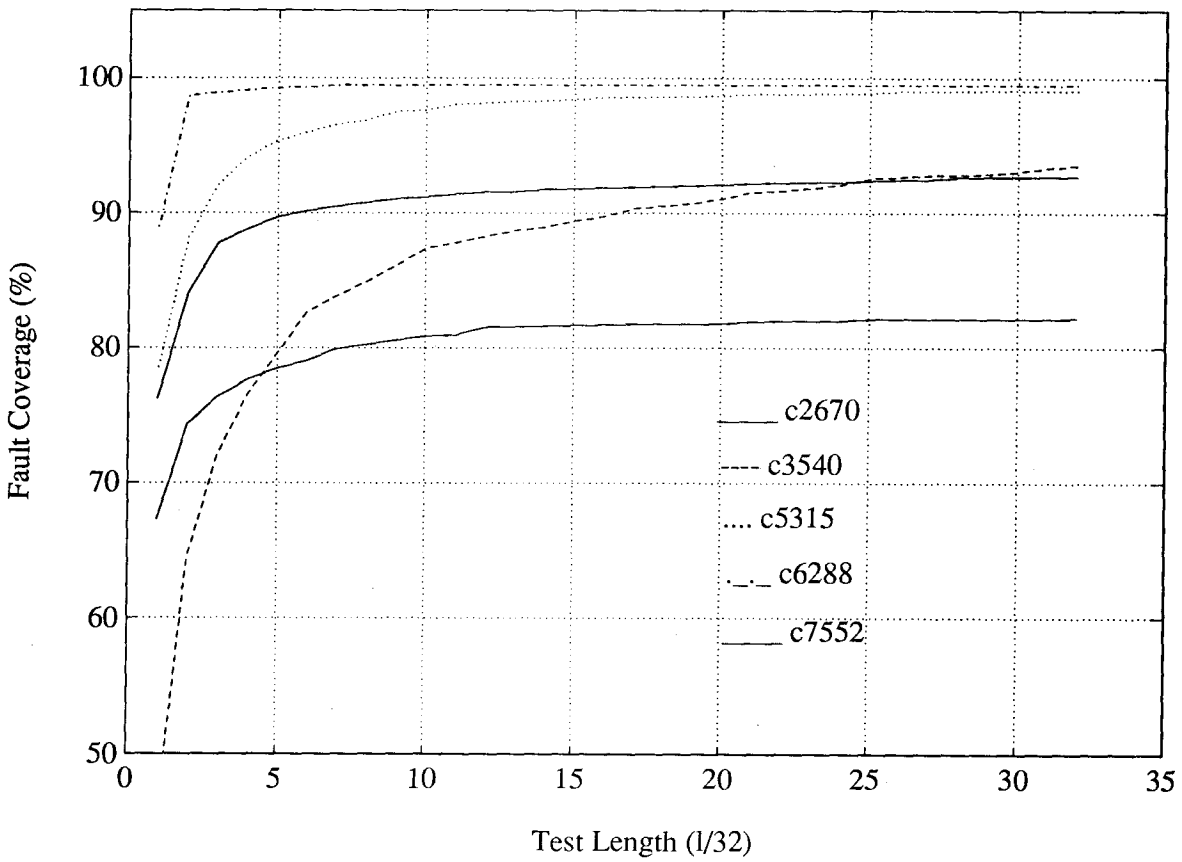



Figure 2.12: Fault Coverage Before Compaction for c2670, c3540, c5315, c6288 and c7552

Figures 2.11 and 2.12 give the fault coverage before compaction for the ten ISCAS'85 benchmark circuits. As we can observe from the figures, the fault coverage before compaction for circuits c880, c1908 and c1355 increase relatively slowly. The fault coverage before compaction for circuit c6288 increases most rapidly among the ten circuits. From Table 2.6, the $Err_{simp}$'s for c880, c1908, and c3540 are 0.0511%, 2.1351% and 0.4696%, respectively, while the $Err_{bha84}$'s for the three circuits are 0.2065%, 2.4291% and 0.5642%, respectively. The differences between $Err_{simp}$ and $Err_{bha84}$ for these three circuits are significant comparing to the differences for other circuits. From Table 2.7, similar conclusion can be drawn. Since the fault coverage before compaction for c6288 increases most rapidly among the ten circuits, from Tables 2.6-2.9 we can see that the $Err_{simp}$ and $Err_{bha84}$ are the same for all the four cases.

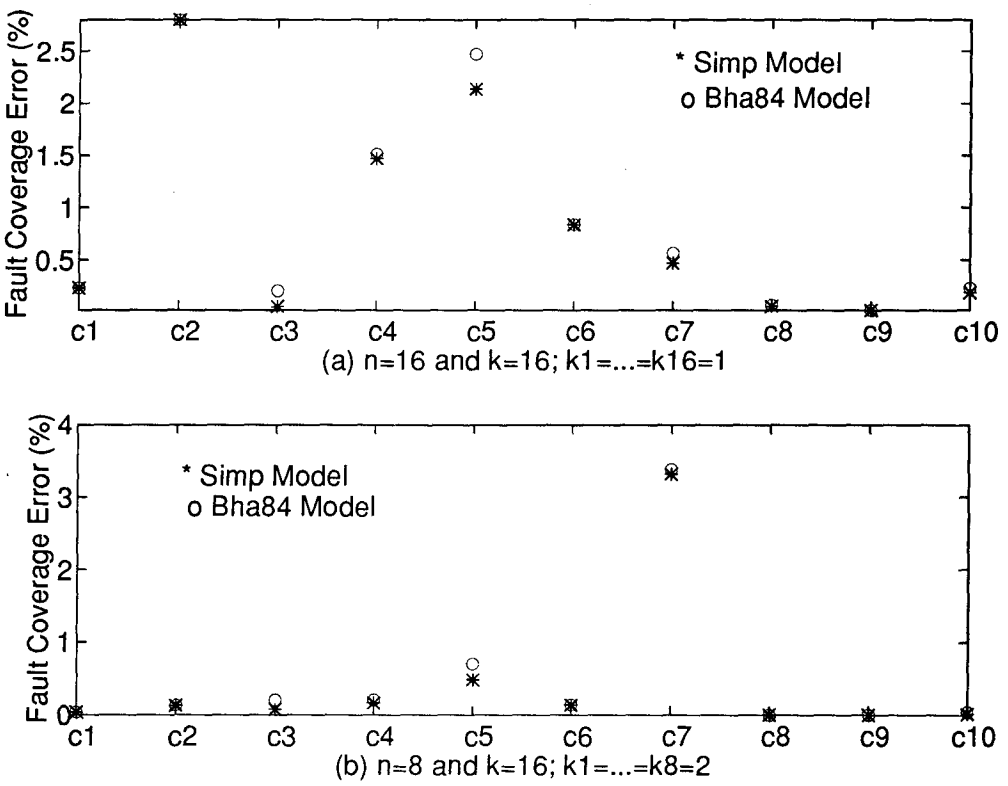



Figure 2.13: Fault Coverage Errors for (a) $(16, 16; 1, \ldots, 1; 2048)$ MSA; (b) $(16, 8; 2, \ldots, 2; 2048)$ MSA

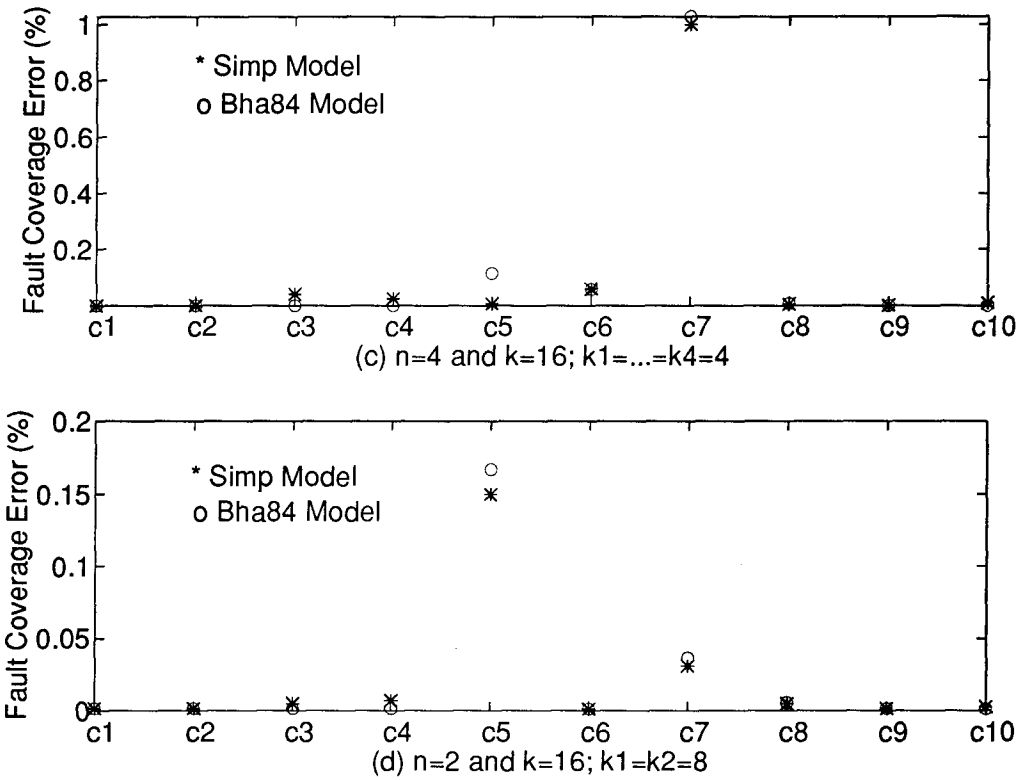



Figure 2.14: Fault Coverage Errors for (c) $(16, 4; 4, \ldots, 4; 2048)$ MSA; (d) $(16, 2; 8, 8; 2048)$ MSA

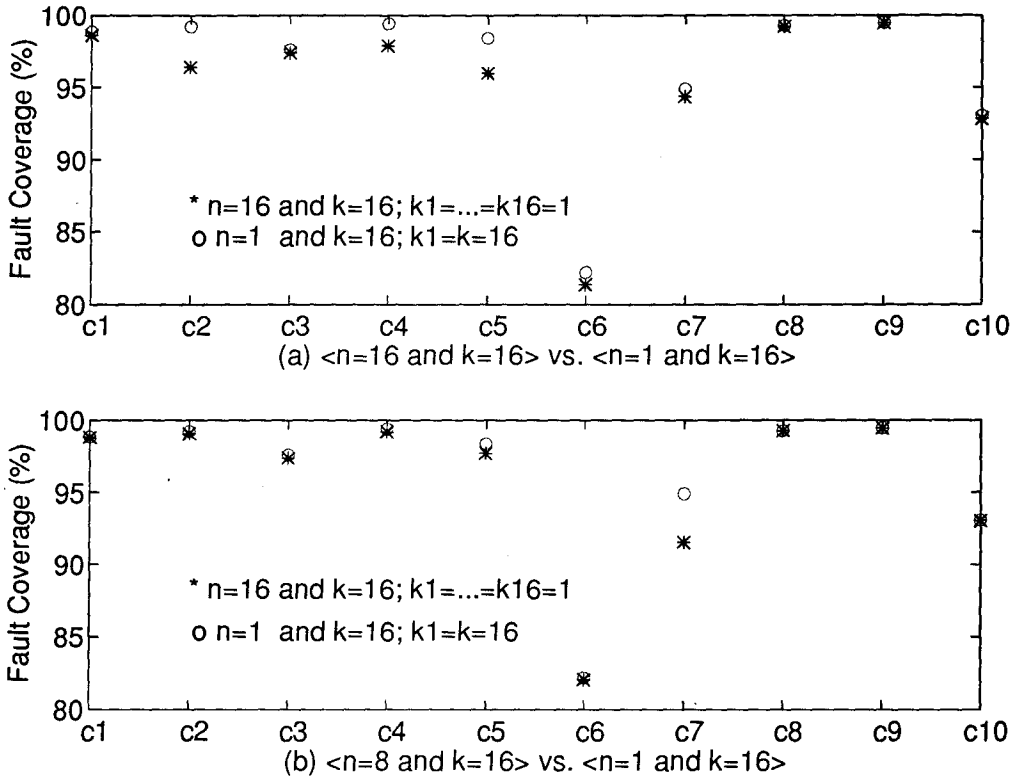



Figure 2.15: Fault Coverage Comparison for MSA and SSA: (a) $(16,16;1,\ldots,1;2048)$ MSA and $(16,1;16;2048)$ SSA; (b) $(16.8;2,\ldots,2;2048)$ MSA and $(16,1;16;2048)$ SSA

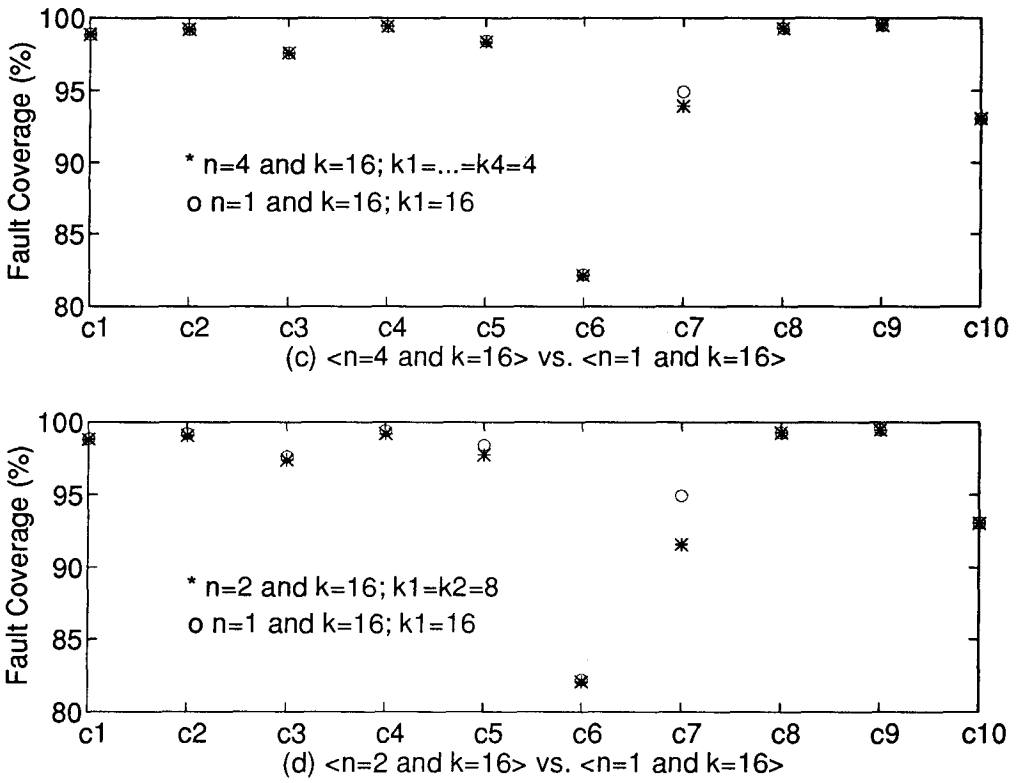



Figure 2.16: Fault Coverage Comparison for MSA and SSA: (a) $(16,4;4,\ldots,4;2048)$ MSA and $(16,1;16;2048)$ SSA; (b) $(16,2;8,8;2048)$ MSA and $(16,1;16;2048)$ SSA

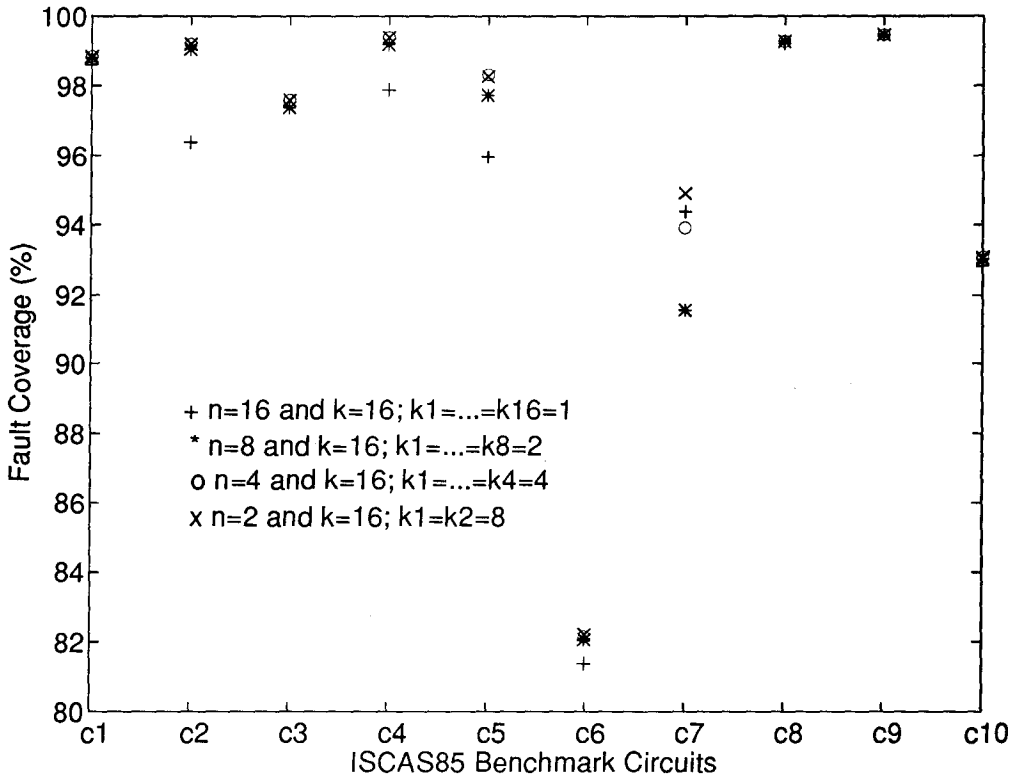



Figure 2.17: Fault Coverage Comparison for Different $n$

## 2.6 Conclusions

A prediction model for fault coverage of multiple signature schemes has been presented in this chapter. From the model and the experimental results presented in this chapter, the following conclusions can be drawn:

- The fault coverage for multiple signature analysis based schemes is a function of both signature scheduling and signature size associated with each signature analyzer. Hence, when designing a MSA BIST scheme, the signature scheduling and signature sizes need to be considered.

- When the fault coverage before compaction for the CUT increases slowly, the model presented in this chapter gives better fault coverage loss prediction than the model presented in [5]. When the fault coverage before compaction for the CUT increases rapidly, the model presented in this chapter gives similar prediction as the model in [5].

- When the total number of signature bits are fixed, on average the fault coverage loss increases with the number of signatures.

- When both the total number of signature bits and the number of signatures are fixed, the partition of $D_n(1, 1, \ldots, 1, k-n+1)$ yields the smallest total aliasing.

- When the total number of signature bits is fixed, taking a single signature at the end of the test on average gives the best fault coverage.

# Chapter 3

## Multiple Signature Fault Simulation Time Model

The complexity of fault simulation in terms of the CPU time and the memory requirements is known to grow at least as the square of the number of gates in the circuit [29]. For VLSI circuits this may be a serious limitation. However, research has been done on methods of reducing fault simulation time [2][28][54]. Circuit structure oriented methods take advantage of the characteristics of circuit structure to improve fault simulation time [41] [36]. On the other hand, modeling-oriented methods use a fault simulation time model dependent on the nature of the fault simulator, and apply an optimal algorithm to reduce fault simulation time [31][33]. In [31] such a fault simulation time model was presented. The fault simulation time model was used in a recursive relationship to determine the optimal schedulings for MSA schemes. However, the fault simulation time model in [31] did not take aliasing into account. In this chapter, we present a fault simulation time model for MSA schemes which takes aliasing into account. With the presented fault simulation time model, the relationship between optimal scheduling and aliasing for MSA schemes can be discussed.

### 3.1 Preliminaries and Definitions

In this section, we introduce some definitions and briefly review some of the results presented in [31].

### 3.1.1 Basic Definitions

**Definition 1:** The *detection constant* $\tau$ is the average simulation time per pattern and per fault for a given fault simulation algorithm.

**Definition 2:** *Full fault simulation* $T_{full}$ is the process of fault simulation without fault dropping, i.e., a fault is dropped from further consideration as soon as it is detected by the fault simulator.

The full fault simulation time, $T_{full}$, is:

$$T_{full} = lf\tau, \tag{3.42}$$

where $l$ is the total number of test patterns applied to the CUT and $f$ is the number of faults simulated.

**Definition 3:** *No compaction fault simulation* $T_0$ is the fault simulation process without compaction but with fault dropping.

**Definition 4:** The *first detection constant* $\tau_d$ is the average fault simulation time per pattern to simulate one fault until it is detected for the first time.

The no compaction fault simulation time $T_0$ is given by

$$\begin{aligned} T_0 &= \tau_d \int_0^l [1 - F(x)]dx \\ &= \tau_d[l - \int_0^l F(x)dx], \end{aligned} \tag{3.43}$$

where $F(x)$ denotes the fault coverage obtained after $x$ patterns have been applied to the CUT, and $\tau_d$ is the average time per pattern to simulate one fault until it is first detected.

**Definition 5:** *Multiple signature compaction fault simulation* is the fault simulation process with multiple signature compaction and fault dropping.

The multiple signature compaction fault simulation time, $T_n$, is:

$$\begin{aligned} T_n &= \sum_{i=0}^{n-1} \tau_d(1 - F(l_i))(l_{i+1} - l_i) \\ &= \tau_d[(l_n - l_0) - \sum_{i=0}^{n-1} F(l_i)(l_{i+1} - l_i)] \\ &= \tau_d[l - \sum_{i=0}^{n-1} F(l_i)(l_{i+1} - l_i)], \end{aligned} \tag{3.44}$$

where $l_1, l_2, \ldots, l_n$ are check points. Obviously, $l_n$ equals $l$ since the last signature is checked at the end of the test.

## 3.2 Time Model

The fault simulation time model is a function of the fault simulation process. Different fault simulation schemes will yield different fault simulation time models. The fault simulation process used in this section is parallel pattern single fault propagation(PPSFP) [55][54].

### 3.2.1 Fault Simulation Time Model for Multiple Signature Analysis

Given a $(k, n; k_1, \ldots, k_n; l_1, \ldots, l_n)$ MSA scheme, assume that the corresponding aliasing probabilities for each signature analyzer are $\rho_1, \rho_2, \ldots, \rho_n$, respectively, and that the corresponding fault coverage before data compaction is known to be $F_1, F_2, \ldots, F_n$ at the check points. Let the total number of faults of interest be $f$. Initially, there are $f$ faults to be simulated during the segment $[l_0, l_1]$. Hence, the fault simulation time FST during the segment $[l_0, l_1]$ is:

$$FST_1 = f l_1 \tau, \tag{3.45}$$

where $l_0 = 0$, $l_1$ corresponds to the first check point.

Since a signature of length $k_1$ is taken at check point $l_1$, $f(F_1 - FCL_1)$ faults will have been detected after the first signature is checked at check point $l_1$, by Eqn. 2.17. That is, $f(F_1 - FCL_1)$ faults will be dropped after the first signature is checked at check point $l_1$, and $f(1 - F_1 + FCL_1)$ remaining faults will be simulated during the segment $[l_1, l_2]$. Therefore, the fault simulation time for the segment $[l_1, l_2]$ is:

$$FST_2 = f\tau(l_2 - l_1)(1 - F_1 + FCL_1). \tag{3.46}$$

Similarly, $F_{i-1} - FCL_{i-1}$ faults will be detected after the $(i-1)^{th}$ signature is checked. Only $f(1 - F_{i-1} + FCL_{i-1})$ faults need to be simulated during segment $[l_{i-1}, l_i]$, and the fault simulation time for the segment $[l_{i-1}, l_i]$ is

$$FST_i = f\tau(l_i - l_{i-1})(1 - F_{i-1} + FCL_i), \qquad i = 1, 2, \ldots, n. \tag{3.47}$$

The total fault simulation time, $FST$, needed after $n$ signatures have been checked at $l_1, l_2, \ldots, l_n$ check points is:

$$\begin{aligned} FST &= \sum_{i=1}^{n} FST_i \\ &= f\tau\{l_n - \sum_{i=1}^{n-1}(l_{i+1} - l_i)F_i + \sum_{i=1}^{n-1}(l_{i+1} - l_i)FCL_i\}. \end{aligned} \tag{3.48}$$

The first component in Eqn. 3.48 represents the fault simulation time needed if only one signature was checked at the end of the test. The second component represents the fault simulation time saving due to the fact that multiple signatures are checked and hence partial fault dropping can be performed. The last component represents the impact of aliasing due to multiple signature analysis.

Eqn. 3.48 predicts fault simulation time with multiple signature analysis. The model is based only on fault coverage before compaction and the aliasing characteristics of the signature analyzers used.

### 3.2.2 Normalized Fault Simulation Time Model

In Sec. 3.2.1, a fault simulation time model is given. The total number of faults in the fault set, $f$, and the average simulation time per fault and per pattern required until a fault is first detected, $\tau$, are used as parameters of the model.

**Definition 6:** The *normalized fault simulation time* is the ratio of the fault simulation time to the product of the number of faults in the fault set $f$ and the detection constant $\tau$.

By Eqn. 3.48, the normalized fault simulation time ,$fst$, is:

$$\begin{aligned} fst &= \frac{\sum_{i=1}^{n} FST_i}{f\tau} \\ &= l_n - \sum_{i=1}^{n-1}(l_{i+1} - l_i)F_i + \sum_{i=1}^{n-1}(l_{i+1} - l_i)FCL_i. \end{aligned} \tag{3.49}$$

Substituting Eqn. 2.17 into Eqn. 3.49, we have

$$fst = l_n - \sum_{i=1}^{n-1}(l_{i+1} - l_i)F_i + \sum_{i=1}^{n-1}(l_{i+1} - l_i)\sum_{j=1}^{i}(F_j - F_{j-1})\prod_{k=j}^{i}\rho_k. \tag{3.50}$$

The normalized fault simulation model from Eqn. 3.50 predicts the fault simulation time with multiple signature analysis, given the fault coverage at the check points before data compaction and the aliasing characteristics of the signature analyzers. In Sec. 3.2.3, experimental results to justify this fault simulation model are presented.

### 3.2.3 Justification of the Normalized Fault Simulation Model

In Sec. 3.2.2, a normalized fault simulation time model is given. Here, the concept of *fault simulation time ratio* is introduced to validate the normalized fault simulation time model.

**Definition 7:** *Fault simulation time ratio* $FST_{ratio}$ is the ratio of the FST for a $(k, n; k_1, \ldots, k_n; l_1, \ldots, l_n)$ MSA scheme to the FST for a $(k, n_0; k_1, \ldots, k_{n_0}; l_1, \ldots, l_{n_0})$ MSA. $n_0$ can be arbitrarily chosen as long as $n_0 \leq k$. Note that once we have selected $n_0$, the FST for $(k, n_0; k_1, \ldots, k_{n_0}; l_1, \ldots, l_{n_0})$ MSA becomes the standard, FST's for other MSA schemes should compare with it.

$FST_{ratio}$ can be used as a measure of the accuracy of the fault simulation time model. By defining $FST_{ratio}$, we can avoid the problem of determining the detection constant $\tau$ as well.

Tables 3.17 and 3.18 report $FST_{ratio}$'s calculated from the normalized fault simulation time model presented in Eqn. 3.48 and $FST_{ratio}$'s calculated from the exact fault simulations. We chose the FST for $k = 16$ and $n_0 = 16$ as the denominator of $FST_{ratio}$. The ratios are calculated for the $n = 16$, $n = 8$, $n = 4$, and $n = 2$ cases. The "Prediction" columns report the $FST_{ratio}$'s calculated from the normalized fault simulation time mode, while the "Actual" columns report the $FST_{ratio}$'s calculated from the exact fault simulation. From the results presented in Tables 3.17 and 3.18 we can see that the normalized fault simulation time model in Eqn. 3.48 gives a relatively good prediction of fault simulation time ratios.

## 3.3 Equidistant Scheduling & Even Partitioning MSA

One possible way to schedule a MSA scheme is to separate the check points evenly. This is called *equidistant scheduling*. Meanwhile, the easist signature bit partition is even partitioning. This is called *even partitioning*. One of the advantages of an equidistant scheduling and even partitioning MSA is its associated implementation simplicity. We discuss the fault simulation time for a special case of equidistant scheduling and equal partitioning.

| Circuit | $n$ | Prediction | Actual |
|---|---|---|---|
| c432 | 16 | 1.0000 | 1.0000 |
| | 8 | 1.2655 | 1.1315 |
| | 4 | 1.9425 | 1.7895 |
| | 2 | 3.5635 | 3.3475 |
| c499 | 16 | 1.0000 | 1.0000 |
| | 8 | 1.2195 | 1.0689 |
| | 4 | 1.7877 | 1.5354 |
| | 2 | 3.2193 | 2.8128 |
| c880 | 16 | 1.0000 | 1.0000 |
| | 8 | 1.2535 | 1.2108 |
| | 4 | 1.8849 | 1.8730 |
| | 2 | 3.3919 | 3.4172 |
| c1355 | 16 | 1.0000 | 1.0000 |
| | 8 | 1.2389 | 1.1221 |
| | 4 | 1.8459 | 1.6792 |
| | 2 | 3.2975 | 3.0148 |
| c1908 | 16 | 1.0000 | 1.0000 |
| | 8 | 1.1625 | 1.0786 |
| | 4 | 1.5996 | 1.4715 |
| | 2 | 2.6802 | 2.5018 |

Table 3.17: This table lists the fault simulation time ratios for $n = 16$, $n = 8$, $n = 4$, and $n = 2$. The preselected case is $n_0 = 16$. The results are for the 5 ISCAS'85 benchmark circuits c432, c499, c880, c1355, c1908. The third column is the fault simulation time ratio calculated from the normalized fault simulation time model in Eqn. 3.48, and the last column is the fault simulation time ratio calculated from the fault simulation results. All the results are averages of three trials.

| Circuit | $n$ | Prediction | Actual |
|---|---|---|---|
| c2670 | 16 | 1.0000 | 1.0000 |
| | 8 | 1.1650 | 1.0482 |
| | 4 | 1.3712 | 1.3120 |
| | 2 | 2.0143 | 1.9296 |
| c3540 | 16 | 1.0000 | 1.0000 |
| | 8 | 1.1159 | 1.2441 |
| | 4 | 1.4440 | 1.4941 |
| | 2 | 2.2915 | 2.2890 |
| c5315 | 16 | 1.0000 | 1.0000 |
| | 8 | 1.2428 | 1.0158 |
| | 4 | 1.8776 | 1.3805 |
| | 2 | 3.4028 | 2.4750 |
| c6288 | 16 | 1.0000 | 1.0000 |
| | 8 | 1.2548 | 1.0429 |
| | 4 | 1.9713 | 1.6013 |
| | 2 | 3.6630 | 2.9930 |
| c7552 | 16 | 1.0000 | 1.0000 |
| | 8 | 1.1912 | 1.1846 |
| | 4 | 1.6607 | 1.6566 |
| | 2 | 2.7789 | 2.7791 |

Table 3.18: This table lists the fault simulation time ratios for $n = 16$, $n = 8$, $n = 4$, and $n = 2$. The preselected case is $n_0 = 16$. The results are for the 5 ISCAS'85 benchmark circuits c2670, c3540, c5315, c6288, c7552. The third column is the fault simulation time ratio calculated from the normalized fault simulation time model in Eqn. 3.48, and the last column is the fault simulation time ratio calculated from the fault simulation results. All the results are averages of three trials.

**Definition 8:** *Dual even MSA* is a equidistant scheduling and even partitioning multiple signature analysis scheme.

### 3.3.1 Number of Signatures vs. Fault Simulation Time

Assume that $n$ signatures are checked, that the total number of signature bits is $k$, and that $l$ test patterns are applied to the CUT. For dual even signature analysis, $n$ signatures of length $\frac{k}{n}$ are checked at $i\frac{l}{n}$, $i = 1, 2, \ldots, n$.

**Example 1:** Assume that the total number of signature bits $k = 16$ and that the test length $l = 2048$. There are four possible dual even multiple signature analysis combinations:

- $n = 16$, $k_1 = k_2 = \ldots = k_{16} = 1$. $l_1 = 128$, $l_2 = 256$, $\ldots, l_{15} = 1920, l_{16} = 2048$;
- $n = 8$, $k_1 = k_2 = \ldots = k_8 = 2$. $l_1 = 256$, $l_2 = 512, \ldots, l_7 = 1792$, $l_8 = 2048$;
- $n = 4$, $k_1 = k_2 = k_3 = k_4 = 4$. $l_1 = 512$, $l_2 = 1024$, $l_3 = 1536$, $l_4 = 2048$;
- $n = 2$, $k_1 = k_2 = 8$. $l_1 = 1024$, $l_2 = 2048$.

Let $FST(n, k, l)$ denote the fault simulation time for the case where dual even multiple signatures are used with $n$ signatures, $k$ signature bits, and test length $l$. Table 3.19 lists the experimental results of fault simulation time for the above four dual even multiple signature cases. The experiments were conducted on the ISCAS'85 benchmark circuits. The results are averages of three trials. From the results of Table 3.19, we can see that $FST(16, 16, 2048) < FST(8, 16, 2048) < FST(4, 16, 2048) < FST(2, 16, 2048)$.

| Circuits | $n$ | $FST(n,k,l)(sec)$ | Circuits | $n$ | $FST(n,k,l)(sec)$ |
|---|---|---|---|---|---|
| c432 | 16 | $n/a$ | c2670 | 16 | 681 |
| | 8 | 12 | | 8 | 725 |
| | 4 | 19 | | 4 | 915 |
| | 2 | 36 | | 2 | 1367 |
| c499 | 16 | 21 | c3540 | 16 | 778 |
| | 8 | 23 | | 8 | 1030 |
| | 4 | 35 | | 4 | 1276 |
| | 2 | 65 | | 2 | 2056 |
| c880 | 16 | 48 | c5315 | 16 | 1626 |
| | 8 | 58 | | 8 | 1983 |
| | 4 | 85 | | 4 | 3026 |
| | 2 | 155 | | 2 | 5624 |
| c1355 | 16 | 103 | c6288 | 16 | $n/a$ |
| | 8 | 116 | | 8 | $n/a$ |
| | 4 | 173 | | 4 | 4308 |
| | 2 | 311 | | 2 | 7934 |
| c1908 | 16 | 180 | c7552 | 16 | 3754 |
| | 8 | 195 | | 8 | 4448 |
| | 4 | 264 | | 4 | 6220 |
| | 2 | 449 | | 2 | 10434 |

Table 3.19: Fault simulation time for four dual even multiple signature analysis cases. The total number of signature bits $k = 16$, test length $l = 2048$. Fault simulation time is in *cpu* seconds on a SPARC 2 workstation.

## 3.4 Optimal Scheduling of Multiple Signature Analysis

In [31], a backward dynamic programming based recursive relationship for fault simulation time prediction is presented. It is used to determine the optimal scheduling of signatures. In this section, a similar dynamic programming based recursive relationship for fault simulation time is presented. Unlike [31], it takes into account the effect of aliasing by using a forward, instead of backward, dynamic programming based recursive relationship.

### 3.4.1 Recursive Relationship

We now present a recursive relationship to determine the optimal scheduling of $j+1$ signatures based on the optimal scheduling of $j$ signatures. The recursive relationship is derived via a dynamic programming approach [12].

Assume that we have $l$ test vectors labeled $1, 2, \ldots, l$. We need to find the minimal fault simulation time $T_{opt}[q, j+1]$ when $j+1$ signatures are optimally scheduled between vectors 1 and $q$, and the $(j+1)^{th}$ signature is scheduled at vector $q$. If the $j^{th}$ signature is optimally scheduled at vector $p_0$, then $T_{opt}[q, j+1]$ is the summation of two terms. The first term is the minimal fault simulation time $T_{opt}[p_0, j]$ when $j$ signatures are optimally scheduled between vectors 1 and $p_0$, and the $j^{th}$ signature is scheduled at vector $p_0$. The second term is the fault simulation time $FST[p_0, q]$ to simulate $1 - F(p_0) + FCL_j$ faults between vectors $p_0$ and $q$ without fault dropping. The possible scheduling for the $j^{th}$ signature is at vectors $j, j+1, \ldots, q-1$. The scheduling for $j^{th}$ signature which yields the minimal value among $T_{opt}[j, j] + FST[j, q]$, $T_{opt}[j+1, j] + FST[j+1, q]$, ..., $T_{opt}[q-1, j] + FST[q-1, q]$, is the optimal scheduling $p_0$ for the $j^{th}$ signature. The minimal value is $T_{opt}[p_0, j] + FST[p_0, q]$, i.e., $T_{opt}[q, j+1]$. In summary:

$$T_{opt}[q, j+1] = \min_{p=j}^{q-1} \{T_{opt}[p, j] + FST[p, q]\}. \tag{3.51}$$

This recursive relationship is illustrated by Fig. 3.18 To use the above recursive relationship, we start with $j = 1$. For $j = 1$, $T_{opt}[q, 1] = FST[1, q]$, where $1 \leq q \leq l$. The optimal scheduling is at vector q. To calculate $FST[q, l]$ for $1 \leq q \leq l$, we need to perform $l$ fault simulations. For $j = 2$, $T_{opt}[q, 2]$ is the minimal value of $T_{opt}[p, 1] + FST[p, q]$ for $1 \leq p \leq q - 1$, where $2 \leq q \leq l$. The value of $p$ which corresponds to the minimal summation is the optimal scheduling for the first signature. The second signature is scheduled at $q$. To determine $T_{opt}[q, 2]$ for each $q \in [2, l]$, we need $q - 1$ fault simulations. Similarly, for $j = 3, \ldots, n$, we can determine $T_{opt}[q, j + 1]$ and the optimal scheduling for the $j^{th}$ signature.

The algorithm to calculate optimal scheduling is summarized as follows. Let $P_{opt}[q, j+1]$ denote the optimal scheduling for the $j^{th}$ signature between vectors 1 and $q$ with the $(j + 1)^{th}$ signature scheduled at vector $q$. The pseudo-code of the algorithm is presented in Fig. 3.18.

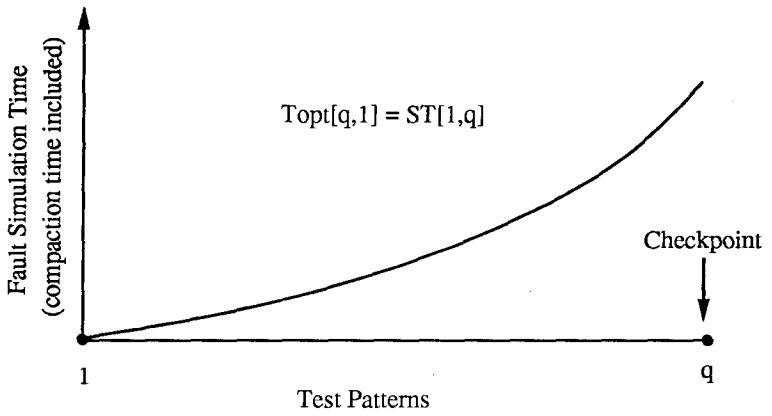
The computation complexity of the *findopt* algorithm is $O(nl^2)$, while the computation complexity of an exhaustive algorithm would be $\begin{pmatrix} l \\ n-1 \end{pmatrix}$. When $l >> n$, the latter would be $O(l^{(n-1)}) = O(l^n)$.

### 3.4.2 Optimal Scheduling vs. Aliasing

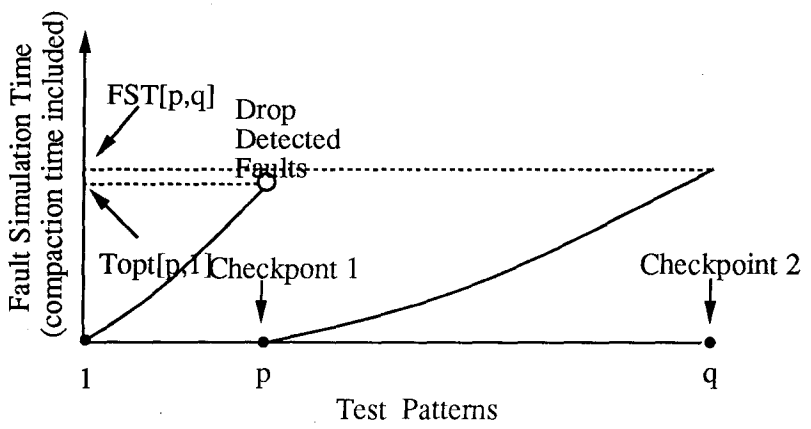
In the algorithm from [31], no aliasing effect was taken into account. In Sec. 3.4.1, we presented a forward dynamic programming based algorithm that considers aliasing. With the effect of aliasing, some of the optimal schedulings move towards the starting point of the test as shown in Fig. 3.20.

To measure the movement quantitatively, we need to define a delta scheduling vector.

**Definition 9:** The *scheduling vector* **SV** for a $(k, n; k_1, \ldots, k_n; l_1, \ldots, l_n)$ MSA scheme




(a) Taking one signature at the end of test q. The optimal fault simulation time Topt[q,1] equals the fault simulation time for a single signature taken at the end of the test. q is in the range of [1, l], where l is the total test length.




(b) Taking two signatures, one is taken between vector 1 and q, the other is taken at the end of test q. The p value which yields the shortest fault simulation time is the optimal point for the first check point. The range of q is [2,l], since the first signature should be checked at least at vector 1.

Figure 3.18: Optimal Scheduling of the First Signature of two signatures

```
findopt(F_i,n,k,l,FCL_i).
    for(q=1 to l) do
        T_opt[q,1] = FST[q,1];
    endfor
    for(j=1 to n-1) do
        for(q=j+1 to l) do
            tmp_min = 99999999; /* temporary storage */
            for(p=j to q-1) do
                T_opt[q,j+1] = T_opt[p,j] + (1 - F_p + FCL_j)(q - p);
                if(tmp_min > T_opt[q,j+1]) then
                    tmp_min = T_opt[q,j+1];
                    p_0 = p;
                endif
            endfor
            P_opt[q,j+1] = p_0;
            T_opt[q,j+1] = tmp_min;
        endfor
    endfor
endfindopt
```

Figure 3.19: Optimal Scheduling Algorithm

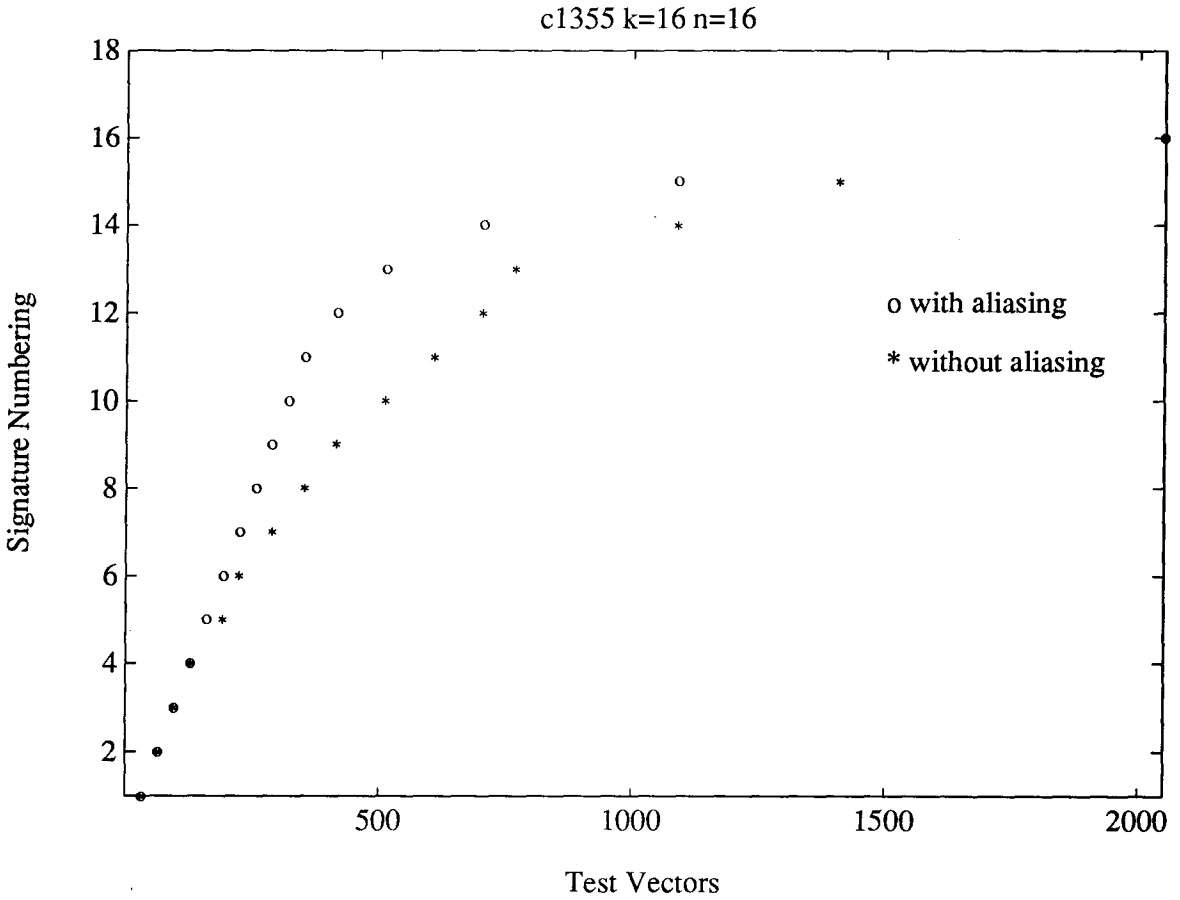



Figure 3.20: This figure presents the movement of optimal scheduling positions when aliasing is taken into account. The horizontal axis represents the test vectors. The vertical axis represents the numbering of signatures, e.g., value $i$ in the vertical axis represents the $i$th signature.

is the vector $(l_1, l_2, \ldots, l_n)$. The *delta scheduling vector* $\delta$**SV** is the difference between the **SV** for an optimal scheduled MSA **SV** with aliasing taken into account and the **SV** for an optimally scheduled MSA without aliasing taken into account. The optimal scheduling for a MSA is obtained using the algorithm shown in Figure 3.19.

Tables 3.20 and 3.21 list the **SV**'s with aliasing for $n = 16$, $n = 8$, $n = 4$, and $n = 2$. The signature partitioning is even signature partitioning and the total number of signature bits is 16. Since we use a PPSFP fault simulator with 32 parallel patterns, the earliest scheduling of signatures is at test vector 32. From Tables 3.20 and 3.21, we can see that the check points for the first signature is always at 32. The reason is that the first signature should be checked at a very early stage of the test so that a big portion of modeled faults can be dropped early. If we used a non-parallel pattern fault simulator, the first check point could be at a test vector earlier than 32. Some of the scheduling is not available for some circuits with $n = 16$ and/or $n = 8$. This is due to the constraint we put for searching for the optimal scheduling. The constraint is that the check points have to be scheduled at a test vector on which at least one more fault has been detected compared to the previous check points. If such points are less than the number of signatures $n$, then there isn't an optimal scheduling for the given $n$.

Tables 3.22 and 3.23 report the $\delta$**SV**'s and the average $\delta$**SV**'s for the ten ISCAS'85 benchmark circuits. The average $\delta$**SV** equals to the summation of all the elements of the $\delta$**SV** divided by the dimension of the $\delta$**SV**. For example, if $\delta\mathbf{SV} = (0, 0, 0, 0, 0, 32, 384, 0)$, the average $\delta\mathbf{SV} = \frac{(0+0+0+0+0+32+384+0)}{8} = 52$. The average $\delta$**SV** gives a measure on the overall movement of the optimal scheduling. The $\delta$**SV**'s in Tables 3.22 and 3.23 are all positive, i.e., the optimal schedulings with aliasing move towards the starting point of the test compared to that without aliasing. When $n$ increases, the movements become more significant for most of the circuits. When $n$ decreases, the movements become less

| Circuit | $n$ | **SV** (with aliasing) |
|---|---|---|
| c432 | 16 | $n/a$ |
| | 8 | 32, 64, 96, 128, 160, 224, 448, 2048 |
| | 4 | 32, 64, 128, 2048 |
| | 2 | 32,2048 |
| c499 | 16 | 32, 64, 96, 128, 160, 192, 224, 256, 288, 320, 352, 384, 480, 512, 704, 2048 |
| | 8 | 32, 64, 96, 128, 192, 288, 480, 2048 |
| | 4 | 32, 96, 192, 2048 |
| | 2 | 32, 64 |
| c880 | 16 | 32, 64, 96, 128, 160, 192, 224, 256, 288, 320, 352, 384, 480, 640, 864, 2048 |
| | 8 | 32, 64, 96, 128, 224, 320, 480, 2048 |
| | 4 | 32, 128, 384, 2048 |
| | 2 | 32, 2048 |
| c1355 | 16 | 32, 64, 96, 128, 160, 192, 224, 256, 288, 320, 352, 416, 512, 704, 1088, 2048 |
| | 8 | 32, 64, 96, 128, 288, 448, 704, 2048 |
| | 4 | 32, 96, 384, 2048 |
| | 2 | 32, 2048 |
| c1908 | 16 | 32, 64, 96, 128, 160, 192, 224, 256, 288, 384, 544, 704, 928, 1344, 1696, 2048 |
| | 8 | 32, 64, 96, 224, 512, 736, 1248, 2048 |
| | 4 | 32, 128, 736, 2048 |
| | 2 | 32, 2048 |

Table 3.20: Optimal Scheduling of Signatures for c432, c499, c880, c1355, c1908 for MSA

| Circuit | $n$ | **SV** (with aliasing) |
|---|---|---|
| c2670 | 16 | 32, 64, 96, 128, 160, 192, 224, 256, 288, 320, 384, 448, 480, 544, 672, 2048 |
| | 8 | 32, 64, 96, 128, 160, 224, 384, 2048 |
| | 4 | 32, 96, 384, 2048 |
| | 2 | 32, 2048 |
| c3540 | 16 | 32, 64, 96, 128, 160, 192, 224, 256, 288, 320, 352, 448, 544, 672, 1120, 2048 |
| | 8 | 32, 64, 96, 160, 352, 576, 1120, 2048 |
| | 4 | 32, 160, 512, 2048 |
| | 2 | 32, 2048 |
| c5315 | 16 | 32, 64, 96, 128, 160, 192, 224, 256, 288, 320, 352, 384, 416, 512, 832, 2048 |
| | 8 | 32, 64, 96, 128, 160, 288, 512, 2048 |
| | 4 | 32, 96, 352, 2048 |
| | 2 | 32, 2048 |
| c6288 | 16 | $n/a$ |
| | 8 | $n/a$ |
| | 4 | 32, 64, 96, 2048 |
| | 2 | 32, 2048 |
| c7552 | 16 | 32, 64, 96, 128, 160, 192, 224, 256, 288, 320, 352, 384, 512, 640, 1088, 2048 |
| | 8 | 32, 64, 96, 128, 160, 288, 640, 2048 |
| | 4 | 32, 96, 280, 2048 |
| | 2 | 32, 2048 |

Table 3.21: Optimal Scheduling of Signatures for c2670, c3540, c5315, c6288, c7552 for MSA.

| Circuit | $n$ | $\delta$**SV** | Average $\delta$**SV** |
|---|---|---|---|
| c432 | 16 | $n/a$ | $n/a$ |
| | 8 | 0, 0, 0, 0, 0, 0, 32, 384, 0 | 52 |
| | 4 | 0,0,0,0 | 0 |
| | 2 | 0,0 | 0 |
| c499 | 16 | 0, 0, 0, 0, 0, 32, 32, 32, 64, 64, 64,64,128,128,128,192,64,0 | 60 |
| | 8 | 0,0,0,0,0,0,0,0 | 0 |
| | 4 | 0,32,128,0 | 40 |
| | 2 | 0, 0 | 0 |
| c880 | 16 | 0, 0, 0, 0, 0, 0, 32, 32, 32, 32, 64, 128, 160, 160, 224, 96, 0 | 60 |
| | 8 | 0, 0, 32, 96, 96, 64, 160, 0 | 56 |
| | 4 | 0, 0, 0, 0 | 0 |
| | 2 | 0, 0 | 0 |
| c1355 | 16 | 0, 0, 0, 0, 32, 32, 64, 96, 128, 192, 256, 288, 256, 384, 320, 0 | 128 |
| | 8 | 0, 0, 32, 160, 128, 256, 384, 0 | 120 |
| | 4 | 0, 32, 32, 0 | 16 |
| | 2 | 0, 0 | 0 |
| c1980 | 16 | 0, 0, 0, 0, 64, 96, 160, 288 384,352,384,416,320,128,128,0 | 170 |
| | 8 | 0, 32, 192, 320, 224, 384, 448, 0 | 200 |
| | 4 | 0, 96, 0, 0 | 24 |
| | 2 | 0, 0 | 0 |

Table 3.22: Delta scheduling vectors $\delta$**SV** for circuits c432, c499, c880, c1355, c1908.

| Circuit | $n$ | $\delta$**SV** | Average $\delta$**SV** |
|---|---|---|---|
| c2670 | 16 | 0,0,0,0,0,0,0,0,<br>0,0,0,32,64,128,64,0 | 18 |
| | 8 | 0,0,0,32,64,160,288,0 | 68 |
| | 4 | 0,0,0,0 | 0 |
| | 2 | 0,0 | 0 |
| c3540 | 16 | 0,0,0,0,0,0,64,96<br>200,192,320,480,640,736,608,0 | 208 |
| | 8 | 0,0,0,0,0,64,64,0 | 16 |
| | 4 | 0,0,0,0 | 0 |
| | 2 | 0,0 | 0 |
| c5315 | 16 | 0,0,0,0,0,0,0,32,<br>64,96,160,256,416,416,448,0 | 118 |
| | 8 | 0,0,0,32,128,128,320,0 | 76 |
| | 4 | 0,32,0,0 | 8 |
| | 2 | 0,0 | 0 |
| c6288 | 16 | $n/a$ | $n/a$ |
| | 8 | $n/a$ | $n/a$ |
| | 4 | 0,64,256,0 | 80 |
| | 2 | 0,0 | 0 |
| c7552 | 16 | 0,0,0,0,0,0,0,0,<br>32,64,160,256.352,448,352,0 | 104 |
| | 8 | 0,0,0,32,96,96,0,0 | 28 |
| | 4 | 0,32,104,0 | 34 |
| | 2 | 0,0 | 0 |

Table 3.23: Delta Scheduling Vectors $\delta$**SV** for c2670, c3540, c5315, c6288, c7552

significant, and even becomes zero for $n = 2$. The most significant average movement occurs for the circuit c3540 when $n = 16$, and the average movement is 208 vectors.

### 3.4.3 Fault Simulation Time vs. Scheduling

From Eqn. 3.48, the fault simulation time FST is a function of signature scheduling, signature bit partitions, number of signatures and test length. We next briefly discuss this relationship for three special cases:

1. Optimal scheduling with aliasing taken into account;
2. Optimal scheduling without aliasing taken into account;
3. Dual even MSA.

The second case is equivalent to the case discussed in [31] except that the check points are confined to multiples of 32 test vectors.

Tables 3.24 and 3.25 report fault simulation time savings compared to the fault simulation time for single signature of size 16 taken at the end of the test. All the fault simulations are conducted on a SPARC 2 workstation. The table entry is the average of three trials. $fst_{sav}$ denotes the fault simulation time saving in the case of optimal scheduling with aliasing taken into account, $fst'_{sav}$ denotes the fault simulation time saving in the case of optimal scheduling without aliasing taken into account, which is the case discussed in [31], and $fst^e_{sav}$ denotes the fault simulation time saving in the case of equidistant scheduling.

According to the results listed in Table 3.24 and 3.25, we can draw the following conclusions:

| Circuit | $n$ | $fst_{sav}(\%)$ | $fst'_{sav}(\%)$ | $fst^{e}_{sav}(\%)$ |
|---|---|---|---|---|
| c432 | 16 | $n/a$ | $n/a$ | $n/a$ |
| | 8 | 95.25 | 95.09 | 83.57 |
| | 4 | 94.35 | 94.29 | 73.96 |
| | 2 | 72.29 | 72.29 | 50.59 |
| c499 | 16 | 90.88 | 91.16 | 84.01 |
| | 8 | 93.45 | 93.26 | 82.60 |
| | 4 | 90.27 | 91.85 | 73.77 |
| | 2 | 58.62 | 58.62 | 50.66 |
| c880 | 16 | 91.02 | 91.23 | 84.00 |
| | 8 | 91.71 | 91.85 | 80.50 |
| | 4 | 89.42 | 89.42 | 71.37 |
| | 2 | 58.62 | 58.62 | 48.05 |
| c1355 | 16 | 90.44 | 90.90 | 83.23 |
| | 8 | 91.53 | 91.98 | 81.18 |
| | 4 | 88.99 | 89.19 | 71.84 |
| | 2 | 54.18 | 54.18 | 49.43 |
| c1908 | 16 | 84.03 | 84.30 | 78.50 |
| | 8 | 84.88 | 84.06 | 76.81 |
| | 4 | 83.09 | 81.13 | 68.84 |
| | 2 | 39.53 | 39.53 | 46.22 |

Table 3.24: Comparisons of fault simulation time saving for (i) optimal scheduling with aliasing taken into account; (ii) optimal scheduling without aliasing taken into account; (iii) equidistant scheduling. Note that $fst_{sav} < fst'_{sav}$ in some cases. This is because that the FCL model under or over estimating the fault coverage loss in those cases. Results for ISCAS'85 benchmark circuits c432, c499, c880, c1355, c1908 are presented.

| Circuit | $n$ | $fst_{sav}(\%)$ | $fst'_{sav}(\%)$ | $fst^{e}_{sav}(\%)$ |
|---|---|---|---|---|
| c2670 | 16 | 76.87 | 76.58 | 70.03 |
| | 8 | 77.65 | 77.96 | 68.07 |
| | 4 | 76.47 | 76.15 | 59.17 |
| | 2 | 30.18 | 30.18 | 39.83 |
| c3540 | 16 | 85.10 | 85.42 | 78.96 |
| | 8 | 81.75 | 81.31 | 72.14 |
| | 4 | 79.83 | 79.90 | 65.49 |
| | 2 | 32.83 | 32.83 | 44.38 |
| c5315 | 16 | 90.41 | 90.42 | 84.59 |
| | 8 | 89.58 | 89.35 | 81.21 |
| | 4 | 82.12 | 86.08 | 71.36 |
| | 2 | $n/a$ | $n/a$ | $n/a$ |
| c6288 | 16 | $n/a$ | $n/a$ | $n/a$ |
| | 8 | $n/a$ | $n/a$ | $n/a$ |
| | 4 | 95.92 | 95.99 | 70.91 |
| | 2 | 37.94 | 37.94 | 46.42 |
| c7552 | 16 | 86.86 | 86.80 | 79.84 |
| | 8 | 86.91 | 86.98 | 76.11 |
| | 4 | 84.45 | 84.38 | 66.60 |
| | 2 | 17.28 | $n/a$ | 46.93 |

Table 3.25: Fault simulation time saving results for the ISCAS'85 benchmark circuits c2670, c3540, c5315, c6288, c7552.

1. With the PPSFP fault simulator, optimal scheduling yields $70 - 90\%$ fault simulation time saving compared to the fault simulation time of a single signature taken at the end of test.

2. Optimal scheduling gives better fault simulation time than the case where no optimal scheduling is performed regardless of whether aliasing is taken into account or not. With the PPSFP fault simulator, optimal scheduling yields an average of 7% more savings than equidistant scheduling for the case with aliasing.

3. Although aliasing has some impact on the optimal scheduling, the impact in terms of fault simulation time is not significant.

## 3.5 Fault Coverage Loss vs. Fault Simulation Time

Using the fault coverage loss model given by Eqn. 2.17 and the normalized fault simulation time model captured by Eqn. 3.49, we can discuss the relationship between fault coverage loss and fault simulation time. We need to use the normalized fault simulation time model presented in Eqn. 3.48 here. The equation is rewritten as:

$$fst = l_n - \sum_{i=1}^{n-1}(l_{i+1} - l_i)F_i + \sum_{i=1}^{n-1}(l_{i+1} - l_i)FCL_i. \tag{3.52}$$

The second term is only a function of check points and $F_i$. The third term is a function of check points and $FCL_i$. $FCL_i$ is a function of $F_i$, the checkpoints, and signature bit partitions. An example is given next to illustrate the relationship between $fst$ and FCL. The relationship can be used to determine $n$ and $k_i$'s for a MSA scheme given the following conditions:

1. The FCL for the MSA scheme is less than a FCL threshold.

2. The hardware cost should be minimized.

**Example 2:** We arbitrarily choose one of the ISCAS'85 benchmark circuits c1355 as an example. The test length used in the example is 2048. Table 3.26 gives the fault coverage data before compaction for c1355.

| test length | fault coverage(%) |
|---|---|
| 128 | 88.3026 |
| 256 | 92.2017 |
| 384 | 93.6285 |
| 512 | 94.8831 |
| 640 | 96.0640 |
| 768 | 97.0480 |
| 896 | 97.4662 |
| 1024 | 97.9828 |
| 1152 | 98.4625 |
| 1280 | 98.6470 |
| 1408 | 99.0283 |
| 1536 | 99.0898 |
| 1664 | 99.3358 |
| 1792 | 99.3358 |
| 1920 | 99.3973 |
| 2048 | 99.3973 |

Table 3.26: Fault coverage data before compaction for c1355 circuit.

The signature scheduling used in this example is equidistant scheduling and the signature bit partitions are even partitions. We discuss the fault simulation time and fault coverage loss for the following four cases.

- $n = 16$, $l_i = i * \frac{2048}{16}$, $i = 1, \ldots, 16$.
- $n = 8$, $l_i = i * \frac{2048}{8}$, $i = 1, \ldots, 8$.
- $n = 4$, $l_i = i * \frac{2048}{4}$, $i = 1, \ldots, 4$.
- $n = 2$, $l_i = i * \frac{2048}{2}$, $i = 1, 2$.

Using the fault coverage loss model from Eqn. 2.17, and using the fault coverage data before compaction listed in Table 3.26, we calculate the fault coverage losses for the total number of signature bits $k = 16, 32, 64, 128$ for the four cases. The results are shown in Fig. 3.21 and Table. 3.27.

| k | $n$ | FCL(%) |
|---|---|---|
| 16 | 16 | 0.046318 |
| | 8 | 0.043996 |
| | 4 | 0.025748 |
| | 2 | 0.007020 |
| 32 | 16 | 0.004979 |
| | 8 | 0.004924 |
| | 4 | 0.001218 |
| | 2 | $3.293386e-10$ |
| 64 | 16 | $2.440702e-04$ |
| | 8 | $2.440146e-04$ |
| | 4 | $4.692333e-06$ |
| | 2 | $7.668017e-20$ |
| 128 | 16 | $9.384729e-07$ |
| | 8 | $9.384722e-07$ |
| | 4 | $7.159540e-11$ |
| | 2 | $4.156841e-39$ |

Table 3.27: FCL vs. k and n for c1355.

Suppose that we need to design a MSA scheme. We require that the FCL of the MSA scheme is at most equal to the FCL of a $k$-bit (e.g., $k = 16$) SSA scheme, i.e., we need to choose the fault coverage threshold such that,

$$\begin{aligned} FCL &\leq (FC_{nc}\ at\ vector2048) * 2^{-k} \\ &= (FC_{nc}\ at\ vector2048) * 2^{-16} \\ &= 0.001516(\%). \end{aligned} \tag{3.53}$$

By looking at Fig. 3.21 and Table 3.27, we can see that the following cases satisfy the condition in Eqn. 3.54:

- When $k = 16$, none of the signatures satisfy the condition given in Eqn. 3.54.
- When $k = 32$, $n = 4$ and $n = 2$ would satisfy the condition given in Eqn. 3.54. $n = 2$ partition will give a better fault coverage than $n = 4$ partition, but $n = 4$ partition will yield shorter fault simulation time from the simulation results presented in Tables 3.24-3.25.
- When $k = 64$, $n = 16$, $n = 8$, $n = 4$ and $n = 2$ will all satisfy the condition given in Eqn. 3.54. From the simulation results presented in Tables 3.24-3.25, and analysis presented in chapter 2, the $n = 2$ partition will give the highest fault coverage among these four case, but the longest fault simulation time, and $n = 16$ will give the fastest fault simulation time among these four cases, but will give the lowest fault coverage.
- When$k = 128$: $n = 16$, $n = 8$, $n = 4$ and $n = 2$ will all satisfy the condition given in Eqn. 3.54. $n = 2$ partition will give the highest fault coverage comparing to that for $n = 16$, $n = 8$, and $n = 4$. However the $n = 2$ partition will yield the longest fault simulation time. The $n = 16$ will give the shortest fault simulation time, but will yield the lowest fault coverage.

When designing a MSA scheme, we need to consider (i) required fault coverage, (ii) hardware overhead, (iii) fault simulation time. By giving the fault coverage threshold in Eqn. 3.54, we set up the minimum acceptable fault coverage. Among some possible choices, there are 10 out of 16 cases where the fault coverage is higher than the minimum acceptable fault coverage. We need to design a test scheme that minimizes hardware overhead and fault simulation time, and yet satisfy the minimum acceptable fault coverage

condition.

To obtain the shortest fault simulation time, we need $n$ as large as possible. So, we select $n = 16$. There are only two choices that remaining, $k = 64, n = 16$ and $k = 128, n = 16$. Both choices satisfy the condition in Eqn. 3.54, but $k = 64, n = 16$ gives less hardware overhead. Therefore, the best choice is $k = 64, n = 16$, i.e., checking 16 4-bit signatures.

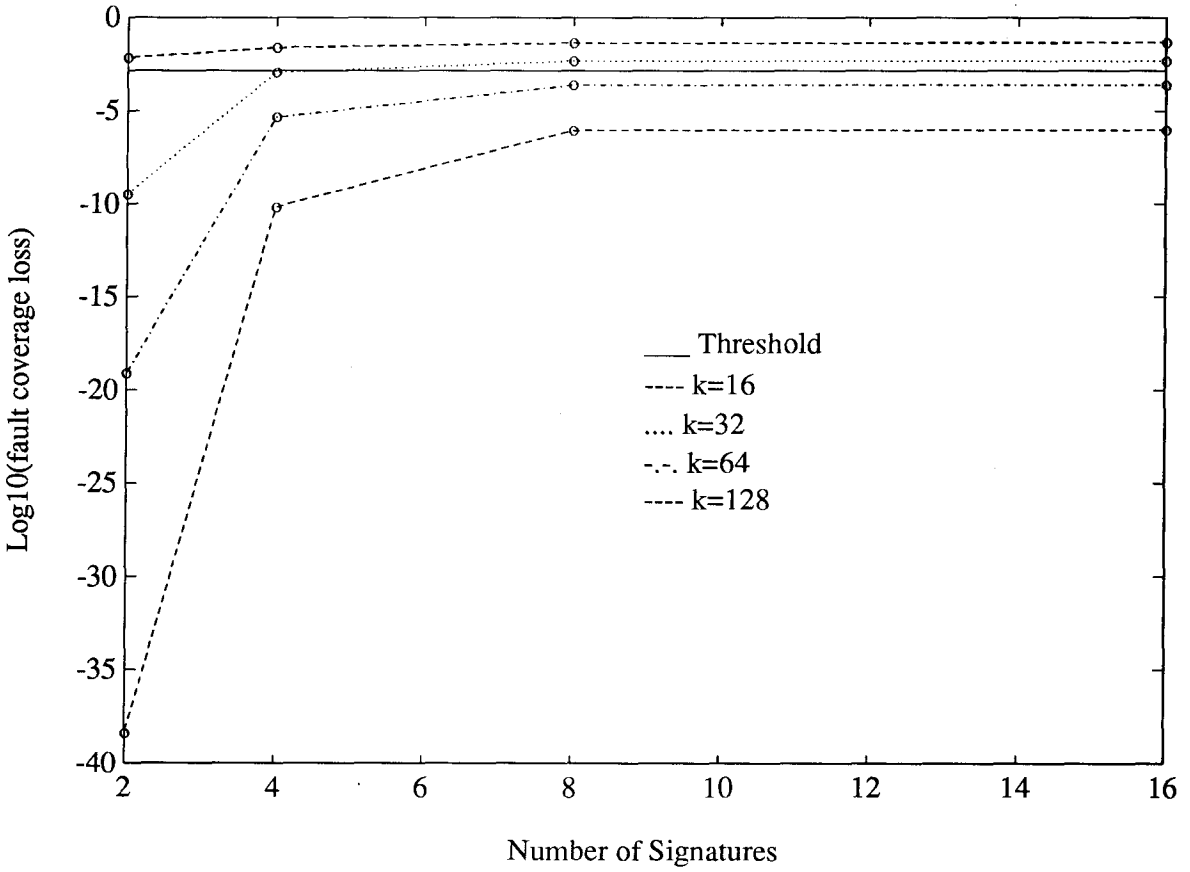



Figure 3.21: Fault Coverage Loss vs. $k$ and $n$. The horizontal axis represents the number of signatures n, the vertical axis represents the logarithm of fault coverage loss.

## 3.6 Conclusions

A fault simulation time model which takes into account the aliasing effect of signature analyzers was developed in this chapter. There are at least two ways to schedule multiple signature analysis: (i) equidistant check points and (ii) optimally scheduled check points. In this thesis, we have discussed these cases. For the equidistant scheduled multiple signatures, for fixed $k$, the fault simulation time decreases as $n$ increases. For the optimally scheduled multiple signature analysis, aliasing affects the optimally scheduled check points slightly, but according to the experimental results, the impact on the fault simulation time is generally less than 5%. Thus, aliasing effects when finding the optimal scheduling check points can be ignored.

# Chapter 4

## Conclusions

The fault coverage model for multiple signature analysis (MSA) proposed previously in this thesis allows us to predict the fault coverage for MSA more accurately than the existing model derived from the assumption of equally likely error model especially for large circuits. With the model, some interesting conclusions towards MSA can be drawn and the conclusions are supported by fault simulation results. Furthermore, the model can be used to aid the design of MSA schemes.

The fault coverage model for MSA developed in this thesis leads to the following conclusions. First, fault coverage for MSA schemes is a function of both checked points scheduling and the signature lengths. Second, when the total number of signature bits is fixed, aliasing increases when the number of signatures increases. Third, when both the total number of signature bits and the number of signatures are fixed, the signature bit partition of $k_1 = k_2 = \ldots = k_{n-1} = 1, k_n = k - n + 1$ yields the smallest aliasing.

The fault simulation time model developed in this thesis takes the aliasing effect of signature analyzers into account. Two ways to schedule multiple signature analysis are: (i) schedule the multiple signatures at equidistant check points, (ii) schedule the multiple signatures at optimally scheduled check points. For the equidistantly scheduled multiple signatures, fault simulation time decreases as the number of signatures increases for a given total number of signature bits. For the optimally scheduled multiple signature analysis, aliasing affects the optimally scheduled check points slightly, but the impact on

the fault simulation time is generally less than 5% according to the experimental results. Therefore, aliasing has negligible effects on finding the optimal scheduling check points. If we only need to find the optimal scheduling check points for a MSA scheme, we can use the fault simulation time model developed in [31].

### 4.0.1 Future Work

The basic ideas used in this thesis to develop the fault coverage prediction model for multiple signature based schemes can be used to analyze other multiple compaction schemes, such as multiple one's counting schemes, multiple partial syndrome schemes, and even the hybrid multiple compaction schemes such as simultaneous signature and syndrome compaction [44].

## Bibliography


[1] M. Abramovici, P. R. Menon and D. T. Miller, "Critical Path Tracing: An Alternative to Fault Simulation," *IEEE Design & Test of Computers*, February 1984, pp. 83-92.

[2] M. Abramovici, B. Krishnamurthy, R. Mathews, B. Rogers, etc. "What is the Path to Fast Fault Simulation? (A Panel Discussion)" *Proc. ITC*, 1988, pp. 183-192.

[3] H. Ando, "Testing VLSI with random access scan," *Dig. Papers Compcon 80*, February 1980, pp. 50-52.

[4] P. H. Bardell, W. H. McAnney and J. Savir, *Built-In Test for VLSI: Pseudorandom Techniques*, John Wiley & Sons, 1987.

[5] D. Bhavsar and B. Krishnamurthy, "Can We Eliminate Fault Escape in Self-Testing by Polynomial Division (Signature Analysis)?" *Proc. ITC*, October 1984, pp. 134-139.

[6] M. A. Breuer, and A. D. Friedman, *Diagnosis & Reliable Design of Digital Systems*, Computer Science Press, INC., 1976.

[7] F. Brglez and H. Fujiwara, "A neutral netlist of 10 combinational benchmark circuits and a target translator in FORTRAN," *Proc. IEEE Int. Symp. Circuits and Systems*, 1985 pp. 151-158.

[8] W. C. Carter, "The Ubiquitous Parity Bit," *Proc. FTCS-82*, 1982, pp. 289-296.

[9] H. Cox, A. Ivanov, V. K. Agarwal and J. Rajski, "On Multiple Fault Coverage and Aliasing Probability Measures," *Proc. ITC*, 1988, pp. 314-321.

[10] S. DasGupta, E. B. Eichelberger, and T.W. Williams, "LSI chip design for testability," *Dig. Tech. Papers, 1978 Int. Solid State Circuits Conf.*, February 1978, pp. 216-217.

[11] B. I. Dervisoglu, "Scan-Path Architecture for Pseudorandom Testing," *IEEE Design & Test of Computers*, 1989, pp. 32-48.

[12] S. E. Dreyfus, and A. M. Law, *The Art and Theory of Dynamic Programming*, Academic Press, 1977.

[13] R. A. Frohwerk, "Signature Analysis: A New Digital Field Service Method," *Hewlett-Parkard J.*, May 1977, pp. 2-8.

[14] H. Fujiwara and S. Toida, "The Complexity of Fault Detection Problem for Combinational Logic Circuits," *IEEE Trans. on Comput.*, Vol. C-31, No. 6., June 1982, pp. 555-560.

[15] H. Fujiwara and T. Shimono, "On the acceleration of test generation algorithms," *IEEE Trans. on Comput.*, Vol. C-32, December 1983, pp. 1137-1144.

[16] S. Funatsu, N. Wakatsuki, and T. Arima, "Test generation systems in Japan," *Proc. 12th Design Auto. Symp.*, June 1975, pp. 114-122.

[17] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman, San Francisco, 1978.

[18] C. S. Gloster and F. Brglez, "Boundary Scan with Built-in Self-Test," *IEEE Design & Test of Computers,* 1989, pp. 36-44.

[19] S. K. Gupta and D. K. Pradhan, "A New Framework for Designing & Analyzing BIST Techniques: Computation of Exact Aliasing Probability," *Proc. ITC*, 1988, pp. 329-342.

[20] P. Goel, "An Implicit Enumeation Algorithm to Generate Tests for Combinational Logic Circuits," *IEEE Trans. on Comput.* Vol. C-30, No. 3, March 1981, pp. 215-222.

[21] L. H. Goldstein and E. L. Tigpen, "SCOAP: Sandia Controllability Observability Analysis Program," *Proc. of 16th Design Automation Conf.*, June 1979, pp. 190-196.

[22] S.Z.Hassan and E.J.McCluskey, "Increase Fault Coverage Through Multiple Signatures," *Proc. 14th Int. Sym. Fault-Tolerant Comp.*, 1984, pp. 354-359.

[23] D. Harvel, "Is There Hope for Linear Time Fault Simulation¿' *Proc. FTCS-87*, 1987, pp. 28-33.

[24] O. H. Ibarra and S. K.Sahni, "Polynomially Complete Fault Detection Problems," *IEEE Tran. on Comput.*, Vol. C-24. No. 3, March 1975, pp. 242-249.

[25] A. Ivanov and V.K. Agarwal, "Testability Measures - What Do They Do for ATPG?" *Proc. 1986 Int. Test Conf.*, 1986, pp. 129-138.

[26] A. Ivanov and V.K. Agarwal, "An Iterative Technique for Calculating Aliasing Probability of Linear Feedback Signature Registers," *Proc. FTCS-88*, June 1988, pp. 70-75.

[27] A. Ivanov, and V.K. Agarwal, "An Analysis of the Probabilistic Behavior of Linear Feedback Signature Registers," *IEEE Trans. on CAD*, Vol. 8, No. 10, October. 1989, pp. 1074-1088.

[28] S.K. Jain and V.D. Agrawal, "STAFAN: An Alternative to Fault Simulation," *Proc. of the 21st Design Auto. Conf.*, 1984, pp. 18-23.

[29] S.K. Jain and V.D. Agrawal, "Statistical Fault Analysis," *IEEE Design & Test*, Vol. 2, No. 1, February 1985, pp. 38-44.

[30] J.R.Kuban and W.C.Bruce,"Self-Testing the Motorola MC6804P2," *IEEE Design & Test*, Vol. 1 No. 2, May 1984, pp. 33-41.

[31] D. Lambidonis, V. K. Agarwal, A. Ivanov, and D. Xavier, "Computation of Exact Fault Coverage for Signature Analysis Schemes," *Proc. ISCAS*, June 1991, pp. 1873-1876.

[32] J.J.LeBlanc, "LOCST: A Built-In Self-Test Techniques," *IEEE Design & Test*, Vol. 1, No. 4, November 1984, pp. 45-52.

[33] Y.-H. Lee, and C.M.Krishna, "Optimal Scheduling of Signature Analysis for VLSI Testing," *Proc. ITC*, 1988, pp. 443-449.

[34] J. Losq, "Efficiency of Random compact testing," *IEEE Tran. on Comput.*, Vol. C-27, June 1978, pp. 516-525.

[35] F. Maamari and J. Rajski, "A Reconvergent Fanout Analysis for Efficient Exact Fault Simulation of Combinational Circuits," *Proc. FTCS-88*, 1988, pp. 122-127.

[36] F. Maamari and J. Rajski, "A Method of Fault Simulation Based on Stem Region," *IEEE Trans. on CAD*, Vol. 9, No. 2, February 1990, pp. 212-220.

[37] E. J. McCluskey, "Built-In Self-Test Techniques," *IEEE Design & Test of Computers*, April 1985, pp. 21-28.

[38] E. J. McCluskey, "Built-in Self-Test Structures," *IEEE Design & Test of Computers*, April 1985, pp. 29-36.

[39] E. J. McCluskey, S. Makar, Samiha Mourad, and K. D. Wagner, "Probability Models for Pseudorandom Test Sequences," *IEEE Tran. on CAD*, Vol. 7, No. 1, January 1988, pp. 68-74.

[40] A. Miczo, *Digital Logic Testing and Simulation*, Happer & Row, Publishers, New York, 1986.

[41] H. B. Min and W. A. Rogers, "Search Strategy Switching: An Alternative to Increase Backtracking," *Proc. ITC*, 1989, pp. 803-811.

[42] R. Raina and P. N. Marinos, "Signature Analysis with Modified Linear Feedback Shift Register (M-LFSR)," *Proc. FTCS-91*, pp. 88-95.

[43] J. Rajski and J. Tyszer, "Experimental Analysis of Fault Coverage in Systems with Signature Registers," *Proc. ETC*, 1991, pp. 45-48.

[44] J. R. Robinson and N. R. Saxena, "Simultaneous Signature and Syndrome Compression," *IEEE Trans. on CAD*, Vol. 7, No. 5, May 1988.

[45] J. Savir, "Syndrome-Testable Design of Combinational Circuits," *IEEE Trans. on Comput.*, Vol. C-29, No. 6, January 1980, pp. 442-450.

[46] J. Savir, G.S. Ditlow and P.H. Bardell, "Random Pattern Testability," *IEEE Trans. on Comput.,* Vol. C-33, Jan. 1984, pp. 79-90.

[47] J. Savir and W.H. McAnney, "On the Masking Probability with One's Count and Transition Count," *Proc. ICAD*, November 1985, pp.

[48] N. R. Saxena, P. Franco, and E. J. McCluskey, "Bounds on Signature Analysis Aliasing for Random Testing," *Proc. FTCS-91*, 1991, pp. 104-111.

[49] S. C. Seth and V.D.Agrawal, "A Theory of Testability with Application to Fault Coverage Analysis," *Proc. ETC*, 1989, pp. 139-143.

[50] J. E. Smith, "Measures of the Effectiveness of Fault Signature Analysis," *IEEE Trans. on Comput.*, Vol. C-29, No. 6, June 1980, pp. 510-514.

[51] J. H. Stewart, "Future testing of large LSI circuit cards," *Dig. Papers 1977 Semiconductor Test Symp.*, October 1977, pp. 6-17.

[52] A. P. Stróle and H. -J. Wunderlich, "Signature Analysis and Test Scheduling for Self-Testable Circuits," *Proc. FTCS-91*, 1991, pp. 96-103.

[53] K. D. Wagner, C. K. Chin, and E. J. McCluskey, "Pseudorandom Testing," *IEEE Trans. Comput.*, Vol. C-36, No. 3. March 1987, pp. 332-343.

[54] J.A. Waicukauski, E.B. Eichelberger, D.O. Forlenza, E. Lindbloom and T. McCarthy, "A Statistical Calculation of Fault Detection Probabilities by Fast Fault Simulation," *Proc. ITC*, 1985, pp. 779-784.

[55] J. A. Waicukauski, E. B. Eichelberger, D. O. Forlenza, E. Lindbloom and T. McCarthy, "Fault Simulation for Structured VLSI," *VLSI Systems Design*, December 1985, pp. 20-32.

[56] R.-S Wei and A.L.Sangiovanni-Vincentelli, "New Front- End and Line Justification Algorithm for Automatic Test Generation," *Proc. 1986 Int. Test Conf.*, 1986, pp. 122-128.

[57] T. W. Williams and K. P. Parker, "Design for Testability - A Survey," *IEEE Trans. on Comput.*, Vol. C-31, No. 1, January 1982, pp. 2-16.

[58] T. W. Williams, "VLSI Testing." *Computer.* Oct. 1984, pp. 126-136.

[59] T. W. Williams, W. Daehn, M. Gruetzner and C.W. Starke, "Comparison of Aliasing Errors for Primitive and Non-Primitive Polynomials," *Proc. ITC*, September 1986, pp. 282-288.

[60] Y. Wu and A. Ivanov, "A Minimal Hardware Overhead BIST Data Compaction Scheme," *Proc. Int. Conf. ASIC*, September 1992, pp.

[61] Y. Wu, C. Zhang, and A. Ivanov, "On Fault Coverage for Multiple Signature Analysis," *in preparation.*

[62] Y. Zorian, and V.K.Agarwal, "A general Scheme to Optimize Error Masking in Built-In Self-Testing," *Proc. 18th Int. Symp. on Fault-Tolerant Computing*, July 1986, pp. 410-415.

[63] Y. Zorian, "Automated Built-In Self-Test for Embedded Macrocells," *Proc. ATE and Instrumentation*, January 1991, pp. 57-62.