A NEW TEST METRIC AND A NEW SCAN ARCHITECTURE FOR EFFICIENT VLSI TESTING

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING

AND THE COMMITTEE ON GRADUATE STUDIES

OF STANFORD UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Kyoung Youn Cho

December 2007

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

_____

(Edward J. McCluskey) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

_____

(H.-S. Philip Wong)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

_____

(Samy Makar)

Approved for the Stanford University Committee on Graduate Studies.

ABSTRACT

With the advance of semiconductor chip technology, the complexity of design and the number of devices in a chip have increased. This trend has made screening defective chips a difficult and expensive task. To overcome the difficulty and cost of VLSI testing, we need to search for better testing techniques.

Chip testing can be classified into two categories: production testing and characterization testing. In production testing, test cost is one of the most important factors because the test is applied to all the fabricated chips. On the other hand, thoroughness is more important than test cost in characterization testing because sample chips are tested to verify design and to develop a production test program.

Traditionally, fault models, such as the single stuck-at fault (SSF) model and the transition fault model, have been used to generate test sets and to measure the thoroughness of test sets. Some defective chips, however, escape test sets that detect all the detectable target faults, demonstrating that these fault models do not represent actual defects well. One approach to reducing the number of test escapes is to find better fault models that represent actual defects, although successful results have not been reported yet. Moreover, the process depends on semiconductor fabrication technologies and is quite expensive. Another approach is to detect each fault multiple times, which is referred to as *N*-detect testing.

This dissertation presents three techniques that improve the efficiency of VLSI testing: gate exhaustive testing, test set reordering, and California scan architecture. The techniques are more efficient than previous techniques with respect to defect detection and test cost. The presented techniques can be used for production testing or characterization testing. Test chip experiments and fault simulation were conducted to measure defect coverage. Test cost was estimated by the number of test patterns or the power consumption during test application.

Gate exhaustive (GE) testing is presented as a new test metric; a *GE test set* applies all possible gate input combinations to each gate and observes the gate

response at an observation point, such as a primary output or a scan flip-flop. The Stanford CRC ELF35 test chips were used to evaluate the GE test metric. The ELF35 test chip is a digital CMOS test chip fabricated using LSI Logic 0.35 micron technology and contains four combinational cores and two sequential cores; the total number of gates is 54,242. Experimental results using 324 defective cores show that a GE test set detects all the defective ELF35 cores, whereas three defective cores escape a SSF test set, and one defective core escapes a 15-detect test set. The test set size of the GE test set is 170% that of the SSF test set and 12% that of the 15-detect test set.

Test set reordering is a technique for reducing the number of test patterns with minimal impact on defect detection. Two test metrics were used to reorder test sets: the GE test metric and the gate super-exhaustive (GSE) test metric. The *GSE test coverage* of a test set is the percentage of observed pairs of gate input combinations. An *observed pair of gate input combinations* is a pair that is applied to the inputs of a gate with the effect of the second combination being propagated to at least one observation point. Test sets were reordered to maximize the cumulative GE coverage or GSE coverage. The Stanford ELF18 test chips were used in the experiments. The ELF18 test chip is an implementation of a R.E.A.L.^{TM} digital signal processor fabricated using Philips 0.18 micron Corelib technology; the total number of gates is 53,732. In this experiment, 140 defective ELF18 cores that failed the GE test set are used. A GE test set with 1,556 patterns is reordered using the GE test metric and the SSF test metric. The unordered test set required 758 patterns to detect all the defective ELF18 cores. The test set reordered using the SSF test metric required 739 patterns, whereas that reordered using the GE test metric required 286 test patterns. A transition fault test set and a 2-detect transition fault test set were reordered using the GSE test metric, and the experimental results on the ELF18 test chips show that the reordering technique is more efficient than that using the transition fault test metric.

California Scan Architecture (CSA) is a scan architecture that modifies test patterns during the scan shift-in operations to improve test quality and to reduce power consumption during test application. This technique is feasible because most of the bits in test patterns generated by automatic test pattern generation (ATPG) tools are

don’t-care bits. Don’t-care bits are assigned using the repeat-fill technique, reducing switching activity during the scan shift-in operations. These repeat-fill patterns are altered to toggle-fill patterns when the patterns are applied to the combinational logic, improving the defect detection of the applied test patterns. Simulation results using the b19 circuit of the ITC’99 benchmark suite show that the average number of SSFs detected per pattern of CSA is 97.9% that of traditional scan architecture (TSA) with random-fill, while that of TSA with repeat-fill is 83.0%. If a pattern detects more faults, more fault sites are observed by the pattern, improving the detection of un-modeled faults, such as bridging faults. The power consumption of CSA is 17.8% that of TSA with random-fill while the power consumption of TSA with repeat-fill is 10.6%. Simulation results using the ELF18 circuit also demonstrate that CSA provide a good trade-off between test quality and power consumption during test application.

The three presented techniques can be combined to improve the efficiency of semiconductor chip testing. The benefits are 1) improved test quality, 2) reduced test cost, and 3) reduced power consumption during test application.

ACKNOWLDGMENTS

I express my deepest gratitude to my teacher, Professor Edward J. McCluskey. Without Professor McCluskey's help, I could not have completed my Ph.D. program. During the period at Center for Reliable Computing (CRC), I had to resolve several challenging issues that I could not have overcome without Professor McCluskey's advice, help, and encouragement. Thanks to Professor McCluskey, the period I spent at CRC was the happiest time in my life. I am very proud to be one of Professor McCluskey's Ph.D. students.

I would like to express my thanks to Professor H.-S. Philip Wong for being my associate advisor and a member of my dissertation reading committee. Although I only studied under Professor Wong for a short time period, he was kind to me and gave me valuable advice on my dissertation and my future. I also thank Dr. Samy Makar for being a member of my dissertation reading committee. Dr. Makar's advice on ATPG tools and test pattern generation was invaluable for my research; he provided me a good role model for a DFT engineer. I also want to express my gratitude to Professor John Gill for being the chairman of my oral defense committee. When I asked Professor Gill to be my oral chairman, he kindly told me that he was interested in my dissertation.

One of the most invaluable aspects of the time at CRC was the interaction, discussion, and relationship with great people: Professor Subhasish Mitra, Professor Nirmal Saxena, Professor Erik Volkerink, Professor Erik Chmelar, Dr. Siyad Ma, Francois-Fabien Ferhani, Dongwhi Lee, Intaik Park, and Jaekwang Lee. I also thank Arden King for her excellent administrative support.

I want to express thanks to my family for their support and encouragement during my study. I also thank other friends who prayed for me and helped me enjoy living at Stanford.

Finally and most importantly, I express my deepest thanks and warmest love to my wife, Hee Jin Hong. She has been with me throughout the long journey of my

Ph.D. study. Sometimes the journey was harsh and difficult, but she was always beside me and encouraged me to continue my study.

I acknowledge that my research was supported by the National Science Foundation (NSF) under contract number CCR-0098128 and LSI Logic. The ELF35 test chip experiments were supported by LSI Logic and Advantest. The ELF18 test chip experiments were supported by Philips Semiconductors (now NXP Semiconductors) and Verigy.

I dedicate this dissertation to my God,

my wife,

and my parents

TABLE OF CONTENTS

LIST OF FIUGRES

LIST OF TABLES

Chapter 1
Introduction

1.1 Background

Semiconductor technologies have followed Moore's law [Moore 65]: "the complexity for minimum component costs has increased at a rate of roughly a factor of two per year." Later, the law was modified and quoted as "the scale of integrated circuits or the number of transistors has doubled every 18 to 24 months" [Bushnell 00]. One of the factors that have made this trend possible is the shrinking feature size of new semiconductor technologies. The reduction of feature size, however, increases the probability that a chip will become defective as the result of manufacturing defects [Wang 06]. A *defect* in semiconductor chips is a physical imperfection created during the manufacturing process of the chips; it is usually the addition of extra material or the omission of intended material [Bushnell 00]. It is impossible to remove all the defects in the current semiconductor fabrication process; as a result, testing is required to detect the defective chips.

A *fault model* is a representation of the effects of defects on circuit behavior. Several fault models have been proposed, such as the single stuck-at fault (SSF) model [Abramovici 90], the bridging fault model [Mei 74], the path delay fault model [Smith 85], the transition fault model [Waicukauski 87], the gate delay fault model [Carter 87], and the segment delay fault model [Heragu 96]. A fault model may be described at the logic, circuit, or physical level.

The *SSF model* is a fault model in which only one node is stuck at logic-0 (a stuck-at-0 fault) or logic-1 (a stuck-at-1 fault) in the circuit under test [Abramovici 90]. A *bridging fault* represents a short between the signal lines that is not intended by the design [Mei 74]. Several bridging fault models have been proposed: the wired-AND bridging fault model, the wired-OR bridging fault model, and the victim-aggressor bridging fault model [Acken 83][Bushnell 00][Wang 06].

A *path delay fault* is defined as a path that cannot propagate a transition from the starting point of the path to the ending point within the clock interval of the circuit [Smith 85]. The starting point is either a primary input of the circuit or the output of a flip-flop; the ending point is either a primary output or the input of a flip-flop. The main issue with the path delay fault model is that too many paths exist in modern VLSI design.

The *transition fault model* represents defects that delay transitions at gate inputs and outputs, consisting of two types: the slow-to-rise type and the slow-to-fall type [Waicukauski 87]. The *gate delay fault model* represents delays lumped at one gate [Carter 87]; this fault model takes the delay size of the faulty gate into account. The manifestation of this fault model depends on the propagation path through the faulty gate. A *segment delay fault model* assumes that the delay of a segment of a path is large enough so that any path through the segment affects system operations [Heragu 96].

A *test metric* is used to measure the thoroughness of test sets; it also guides automatic test pattern generation (ATPG) tools to generate test sets [McCluskey 04a][McCluskey 04b]. When the thoroughness of a test set is measured using a test metric, the result is usually given as the *test coverage* of the test set. Traditionally, test metrics have been defined using fault models. For example, the SSF test metric and the *N*-detect test metric [Ma 95] are defined using the SSF model, whereas the transition fault test metric, the *N*-detect transition fault test metric, and the TARO test metric [Tseng 01b] are defined using the transition fault model.

Test metrics defined using fault models measure the thoroughness of a test set by the percentage of target faults detected by the test set. The *N-detect test metric* measures the thoroughness of a test set by the percentage of target faults detected at least *N* times by the test set [Ma 95]. The effectiveness of *N*-detect test sets in detecting defects is discussed in [Benware 03][Amyeen 04][Venkataraman 04]. A *TARO test set* propagates each transition fault to all the reachable outputs [Tseng 01b]. The effectiveness of TARO test sets is reported using test chip experiments [Tseng 01b][Park 05]. Test chip experiments over several technology generations have clearly

demonstrated the need for new test metrics for grading test sets and for ATPG [McCluskey 00][McCluskey 04a].

In modern VLSI design, a full-scan design is usually used for better controllability and observability of internal states. In a *full-scan design*, all the flip-flops are replaced by scan flip-flops, and the scan flip-flops are connected as shift registers; test patterns are shifted in and circuit responses are shifted out through the shift registers. Several scan designs have been presented: muxed-D scan design [Williams 73]; level-sensitive scan design (LSSD) [Eichelberger 77]; two-port flip-flop design [McCluskey 86]. New scan architectures have been presented to improve fault coverage [Datta 04], to reduce tester cost [Agrawal 95][Shashaani 99], to reduce test application time [Hamzaoglu 99][Rosinger 04], or to decrease power consumption during testing [Nicolici 02][Sinanoglu 03][Yoshida 03].

To reduce test application time and test set size, Illinois scan architecture is presented [Hamzaoglu 99]. In this scan architecture, the same test vector is shifted into multiple scan chains in parallel; to test faults undetectable by the parallel scan shift-in, serial scan shift-in operations are conducted. This scan architecture is improved to reduce the number of serial scan shift-in operations by reconfiguring the scan architecture [Pandey 02] and by controlling each scan chain independently [Al-Yamani 05]. In [Arslan 04], captured data is used as a template for the next test pattern, and only mismatched care bits are scanned in, reducing test application time and test volume. A *care bit* is a specified bit (a logic-0 or logic-1) in a test pattern that is assigned by an ATPG tool.

In this dissertation, when the *effectiveness* of test techniques is compared, only the quality or thoroughness of the techniques is considered; when the *efficiency* of techniques is compared, both the cost and quality of the techniques are considered.

1.2 Contributions

Figure 1.1 illustrates semiconductor chip design and test flow. One goal of this dissertation is to present a new test metric for improving the quality of test sets and for reducing the size of test sets with minimal impact on defect coverage. The other goal

is to develop a design for testability (DFT) technique for reducing power consumption during test application and for enhancing test quality.

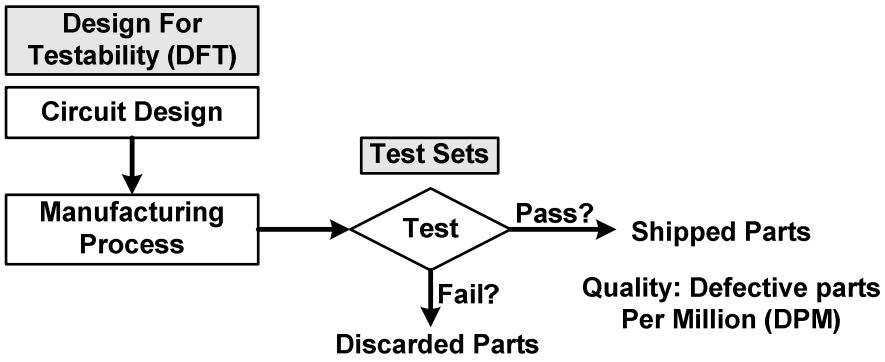


Figure 1.1 Semiconductor chip design flow

The contributions of my Ph.D. research relate to techniques for improving the efficiency of VLSI testing:

- A new test metric for measuring the thoroughness of test sets and for generating effective test sets in detecting defective chips.
- Test set reordering techniques for reducing the number of test patterns with minimal impact on defect detection.
- A new scan architecture for providing a trade-off between defect detection and power consumption for the testing of scan-based designs.

1.3 Outline

This dissertation summarizes my research on improving the efficiency of VLSI testing. Detailed results are presented in the appendices, which are reprints of published papers. This dissertation is organized as follows.

Chapter 2 presents gate exhaustive (GE) testing. Experimental results demonstrate that GE test sets are more efficient than SSF or *N*-detect test sets. The efficiency of test sets is compared using defect coverage and test set size. The correlation between the GE coverage and defect coverage of test sets is also demonstrated by experimental results. This chapter is based on [Cho 05].

Chapter 3 describes test set reordering techniques using the GE test metric discussed in Chapter 2 and the gate super-exhaustive (GSE) test metric which will be presented in Chapter 3. Experimental results demonstrate that the techniques are efficient in reducing test set size with minimal impact on defect detection. This chapter is based on [Cho 07a].

Chapter 4 presents California scan architecture. The architecture provides a trade-off between defect detection and power consumption for the testing of scan-based designs. Simulation results demonstrate the efficiency of California scan architecture. This chapter is based on [Cho 07b].

Finally, Chapter 5 concludes this dissertation.

Chapter 2
Gate Exhaustive Testing

This chapter presents gate exhaustive (GE) testing and demonstrates the efficiency of GE test sets over single stuck-at fault (SSF) and *N*-detect test sets. Experimental results using the Stanford ELF35 test chips reveal the correlation between the GE coverage and the effectiveness of a test set in detecting defective chips.

This chapter is organized as follows: Section 2.1 introduces related work; Section 2.2 presents the GE test metric and the calculation of GE coverage; Section 2.3 illustrates the effectiveness of GE testing using example circuits; Section 2.4 reports experimental results demonstrating the efficiency of GE testing; Section 2.5 presents the *N*-detect SSF simulation results of GE test sets; Section 2.6 summarizes this chapter.

This chapter is based on the paper published in the Proceedings of International Test Conference (ITC) 2005. The published paper is reprinted in Appendix A.

2.1 Background

Traditionally, fault models, such as the SSF model and transition fault model, have been used for test set generation and fault grading. Experimental results using test chips, however, have demonstrated that *N*-detect test sets are more effective than 100% SSF test sets in detecting defective chips [Ma 95][McCluskey 00][McCluskey 04a]. An *N-detect test set* detects each SSF by either *N* "different" test patterns or the maximum number of different patterns if it is impossible to find such *N* different test patterns [Ma 95][McCluskey 00]. *N*-detect test sets are effective in detecting defects because they activate fault signals with different logic states around the fault sites, increasing the possibility to detect un-modeled faults, such as bridging faults.

There are three fundamental issues related to the use of the *N*-detect test metric. First, what value of *N* should be used? In the Murphy test chip experiments, a 5-detect test set detected all of the defective cores [Ma 95][McCluskey 00]. On the other hand, in the ELF35 [McCluskey 04a] experiments, a 15-detect test set failed to detect all of the defective cores. The second issue with the *N*-detect test metric is the fact that the number of patterns of *N*-detect test sets grows approximately linearly with *N* [Pomeranz 03][Venkataraman 04]. Finally, the open question with the *N*-detect test metric is the diversity of the test patterns that detect a fault *N* times; this issue has been addressed in [Tseng 01a][Blanton 03][Dworak 04].

Exhaustive testing of a combinational circuit applies all possible patterns to the inputs of the circuit under test, and it ensures the detection of all irredundant combinational defects (defects that do not introduce additional states) in the circuit. The major problem with an exhaustive test set is the exponential increase in the number of patterns with the increase in the number of circuit inputs. The test length issue of exhaustive test sets can be alleviated by using pseudo-exhaustive testing [McCluskey 81][McCluskey 84]. Pseudo-exhaustive testing partitions a circuit into segments and tests each segment exhaustively. This technique reduces the number of patterns significantly compared to exhaustive testing, but provides similar effectiveness as exhaustive testing [Archambeau 84]. However, it is difficult and computationally intensive to find the optimum partitions because the problem is NP-complete [Shperling 87].

The above discussion demonstrates that we should continue to search for better test techniques. One candidate test technique is GE testing [McCluskey 93]. A *GE test set* applies all possible input combinations to each gate and observes the gate response at an observation point, such as a primary output or a scan flip-flop. This chapter presents GE testing and compares the efficiency of GE testing with other techniques such as SSF testing, *N*-detect testing, and transition fault testing.

2.2 The Gate Exhaustive Test Metric

The basic idea of GE testing is presented using the circuit in Fig. 2.1. Test patterns are applied to the circuit inputs, such as primary inputs and scan flip-flops; the patterns apply all possible input combinations to each gate in the circuit and sensitize the gate output to an observation point, such as a primary output or a scan flip-flop. In GE testing, gates can be elementary gates (e.g., AND, OR, NAND, NOR, inverter), complex gates (e.g., AOI, OAI, XOR, multiplexer, adder, etc.), or circuit segments (e.g., logical cone).

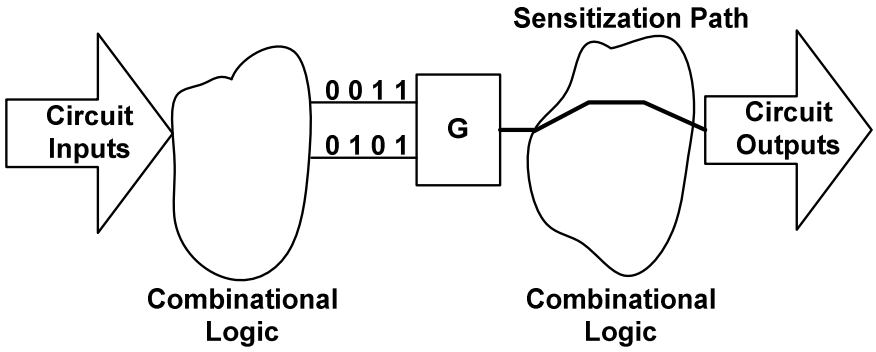


Figure 2.1 Gate exhaustive testing

In general, it may not be possible to apply all input combinations to an internal gate and observe the effects of the combinations at an observation point [McCluskey 93]. The circuits in Figs. 2.2 and 2.3 are used to explain these gate input combinations. Figure 2.2 illustrates an implementation of a multiplexer, which is used to show a gate input combination that cannot be applied to the inputs of a gate. In the circuit, the (J1, J2) = (1, 1) combination cannot be applied to the inputs of gate J; this type of gate input combination is known as a "controllability don't-care" [De Micheli 94]. A *controllability don't-care* is a gate input combination that cannot be applied to the inputs of an internal gate.

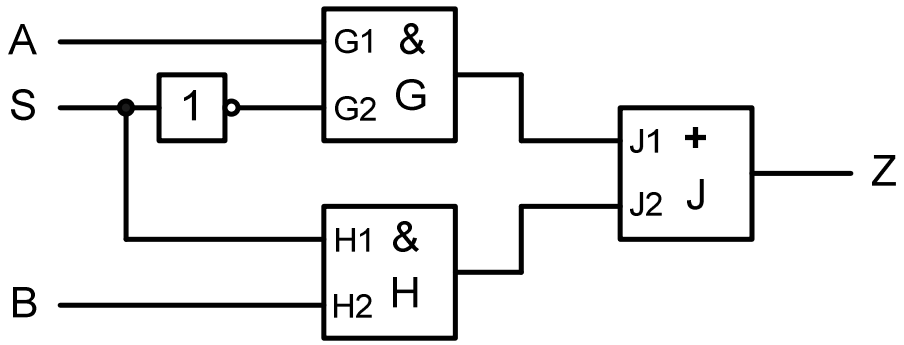


Figure 2.2 Example circuit to explain controllability don't-care

Figure 2.3 illustrates an example circuit to show a gate input combination that cannot be sensitized to any observation point. When the (H1, H2) = (0, 0) combination is applied to the inputs of gate H, the output of gate H cannot be sensitized to the circuit output (Z) because the sensitization path is blocked at gate J. This type of gate input combination is known as an "observability don't-care" [De Micheli 94]. An *observability don't-care* is a gate input combination that can be applied to an internal gate, but the effect of the gate input combination cannot be propagated to any observation point.

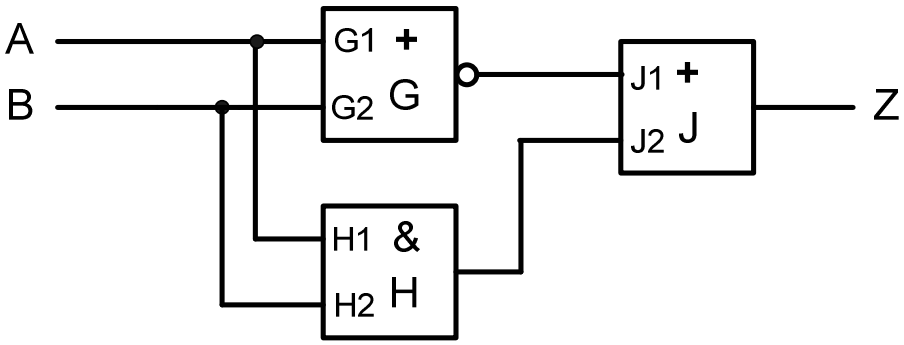


Figure 2.3 Example circuit to explain observability don't-care

The following definitions are used to define the GE test metric.

Definition 2.1: An *observed input combination* of a gate is a logic combination that is applied to the gate inputs with the gate output being sensitized to at least one observation point, such as a primary output or a scan flip-flop.

Definition 2.2: An *observable input combination* of a gate is a logic combination that can be applied to the gate inputs with the gate output being sensitized to at least one observation point.

Definition 2.3: A *nonobservable input combination* of a gate is a logic combination that cannot be applied to the gate inputs with the gate output being sensitized to at least one observation point.

The *GE test metric* measures the thoroughness of a test set by the ratio of the number of observed gate input combinations to the total number of observable gate input combinations. The *GE coverage* of a test set is defined as the value calculated using the GE test metric.

The concept of GE testing and the calculation of GE coverage are illustrated using the circuit in Fig. 2.4. In the circuit, the (G1, G2) = (0, 0), (0, 1), and (1, 0) combinations are nonobservable gate input combinations because the effects of these combinations cannot be sensitized to any observation point; the (J1, J2) = (0, 1) combination is nonobservable because the combination cannot be applied to the inputs of gate J. A 100% SSF test set for the circuit is {(A, B, C) | (1, 1, 0), (0, 1, 1), (1, 0, 0), (1, 1, 1)} while a GE test set is {(A, B, C) | (0, 0, 1), (0, 1, 1), (1, 0, 1), (1, 1, 1), (0, 0, 0), (1, 1, 0)}.

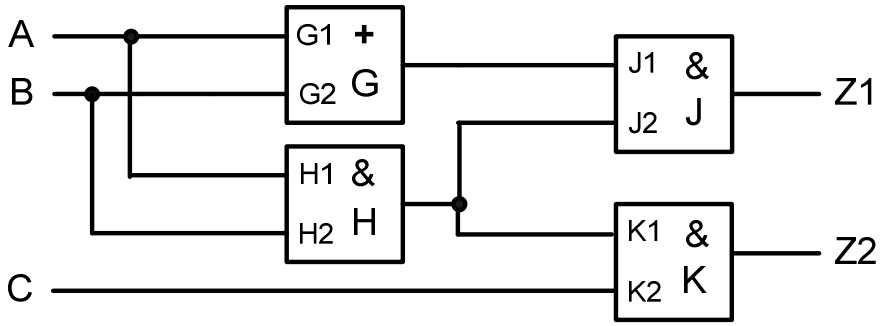


Figure 2.4 Example circuit to illustrate GE testing

Figures 2.5 and 2.6 illustrate the observed gate input combinations of each gate in the circuit for the SSF test set and the GE test set, respectively. In the figures, the nonobservable gate input combinations are marked "N.O.," and the observed gate input combinations are marked "O."

G1 \ G2	0	1
0	N.O.	N.O.
1	N.O.	O

(a)

J1 \ J2	0	1
0		N.O.
1	O	O

(b)

H1 \ H2	0	1
0		O
1	O	O

(c)

K1 \ K2	0	1
0	O	O
1	O	O

(d)

Figure 2.5 Gate input combinations observed by a SSF test set: (a) gate G; (b) gate J; (c) gate H; (d) gate K

G1 \ G2	0	1
0	N.O.	N.O.
1	N.O.	O

(a)

J1 \ J2	0	1
0	O	N.O.
1	O	O

(b)

H1 \ H2	0	1
0	O	O
1	O	O

(c)

K1 \ K2	0	1
0	O	O
1	O	O

(d)

Figure 2.6 Gate input combinations observed by a GE test set: (a) gate G; (b) gate J; (c) gate H; (d) gate K

Table 2.1 reports the number of all gate input combinations and nonobservable gate input combinations for each gate in the circuit illustrated in Fig. 2.4. The last two columns present the number of gate input combinations observed by the SSF and GE test sets, respectively. The table also reports that the GE coverage of the SSF test set is 83.3%, whereas that of the GE test set is 100%.

Table 2.1 Table to calculate GE coverage

Gate	Number of all gate input combinations	Number of nonobservable gate input combinations	Number of observed gate input combinations	
			SSF test set	GE test set
G	4	3	1	1
H	4	0	3	4
J	4	1	2	3
K	4	0	4	4
GE test coverage			83.3%	100.0%

2.3 Effectiveness of Gate Exhaustive Testing

In SSF and *N*-detect test generation, to detect more SSFs, at most one controlling value is applied to the inputs of a gate. A *controlling value* of an elementary gate input is a logic value that determines the gate output logic value independent of the logic value assignments to other inputs [Abramovici 90]. For example, the controlling value to an AND or NAND gate input is logic-0; the controlling value to an OR or NOR gate input is logic-1. A *sensitization path* is a path that propagates a fault effect. If the gate input combination "00" is applied to a two-input NAND gate, all of the sensitization paths through the gate are blocked, which reduces the number of detected SSFs. Therefore, in SSF testing and *N*-detect testing, "01," "10," and "11" combinations are applied to the inputs of AND or NAND gate; these combinations detect all detectable SSFs on the inputs and output of the gate.

Applying some gate input combinations to a gate may identify specific types of defects inside the gate that may not be screened by detecting only SSFs at the inputs and output of the gate. Let us consider the transistor level implementation of a two-input AND gate with a bridging defect between the internal node N and the gate output as illustrated in Fig. 2.7. A *strong logic value* is assigned to a gate output when the inputs of the gate are assigned such that the resistance between VDD or GND and the gate output is minimized [Cusey 97]. When both the PMOS transistors M1 and M2 turn on simultaneously, a strong logic-1 is assigned to internal node N. This strong logic-1 may drive the gate output to a logic-1, even though the NMOS transistor M6

drives the gate output to a logic-0, creating the fault effect of the defect. In GE testing, the fault effect is observed at an observation point, detecting the defect. Although the defect detection depends on the driving strengths of transistors and the resistance of the bridging defect, GE testing improves the possibility of detecting the defect. The fault effect of the defect, however, may not be activated by applying only the "01," "10," and "11" combinations. Note that SSF test sets or *N*-detect test sets may not observe the "00" combination to a two-input AND gate.

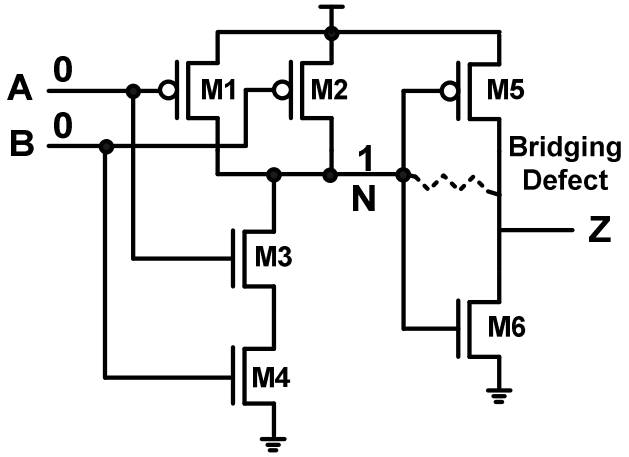


Figure 2.7 A transistor level implementation of a two-input AND gate with a bridging defect

Applying all controlling values to the inputs of a gate may also be effective in detecting bridging defects between nets. Let us consider the circuit illustrated in Fig. 2.8. Applying the "00" combination to the inputs of gate K assigns a strong logic-1 to the gate output; the strong logic-1 may change the logic value on the node connected to the output of gate K by a bridging defect, detecting the defect if the fault effect is propagated to an observation point. On the contrary, the fault effect of the bridging defect may not be activated, if only the "01," "10," and "11" combinations are applied to the inputs of gate K.

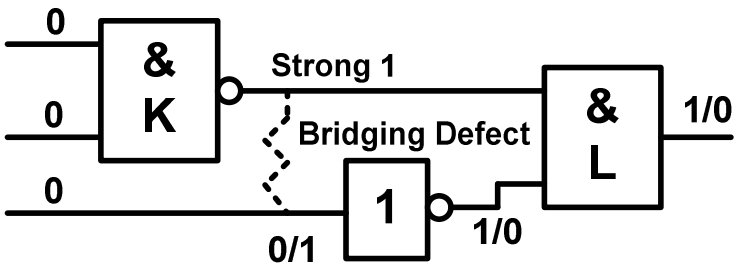


Figure 2.8 Bridging defect detection

Figure 2.9 illustrates another example that explains the effectiveness of GE testing; GE test patterns sensitize a fault site with different conditions around the fault site. Let us assume that there is a bridging defect between the outputs of gates K and L. To activate the effect of the bridging defect, a strong logic-1 must be assigned to the output of gate K, and a logic-0 must be assigned to the output of gate L. To observe the fault effect, the output of gate L must be sensitized to an observation point. The defect may be detected by observing the "001" combination to the inputs of gate L. In GE testing, the "001" combination is assigned to the inputs of gate L and the output of the gate is sensitized to the output Z2, improving the possibility of detecting the defect. However, SSF testing or *N*-detect testing may not observe the "001" combination to the inputs of gate L.

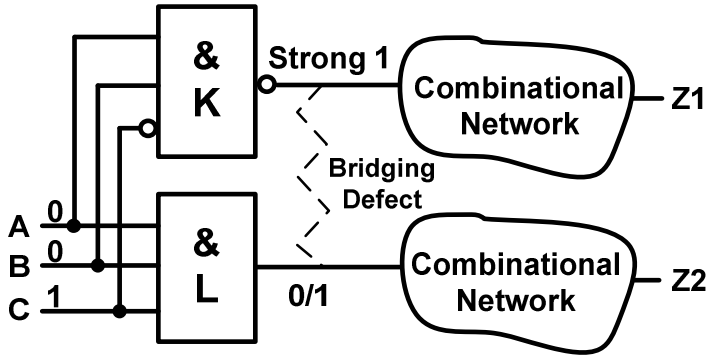


Figure 2.9 Fault effect activation

The examples demonstrate that observing more gate input combinations increases the likelihood of defect detection, although defect detection depends on the relative driving strength of the transistors in the examples. *N*-detect testing may provide similar effects as GE testing; however, *N*-detect testing does not guarantee the

diversity in creating fault effects that is provided by GE testing. Note that GE testing does not guarantee the detection of sequence-dependent defects. If a chip has *sequence-dependent defects*, the test results of the chip depend on the order of the applied patterns [Li 02].

2.4 Experimental Results

In the experiments, the ELF35 test chips [McCluskey 04a] were used. ELF stands for "Early Life Failure" or "ELusive Failure." The ELF35 test chips were fabricated by LSI Logic using the G10P standard cell technology (L_{eff} = 0.35 micron); the nominal supply voltage is 3.3V. Over 10,000 chips were tested, and 324 cores that failed at least one of the 278 test sets applied at two voltages (3.3V and 1.4V) and three test speeds (fast, rated, and slow) were collected. The fast clock cycle time is the minimum clock cycle time at which all of the good cores can operate; this clock cycle time is determined using the shmoo plots of all the good cores. The rated clock cycle time and the slow clock cycle time are 1.3 times the fast clock cycle time and three times the fast clock cycle time, respectively. One ELF35 chip is composed of four combinational cores and two sequential cores. The defects that are present on the chips are only those that occur naturally during fabrication. No artificial defects were inserted.

GE test sets of the ELF35 circuits were generated using the Cadence Encounter Test Design Edition ATPG tool [Cadence 03]. In order to generate a GE test set, the pattern fault model supported by the ATPG tool was utilized. A *pattern fault* is defined as logic value constraints on a set of nets and responses to be observed [Cadence 03]. An example of a pattern fault is given in Fig. 2.10. The logic value constraints are (A) = (0) and (B) = (0), and the response to be observed is the logic-0 on Z. As a result, if the pattern fault is detected, the (A, B) = (0, 0) combination is observed at an observation point. The exhaustive gate input combinations and the corresponding gate output responses for all the technology library cells used in the ELF35 circuits were specified using pattern faults. If a gate input combination is

nonobservable, it is classified as a redundant fault by the ATPG tool. A *redundant fault* is a fault that cannot be tested because of the circuit structure.

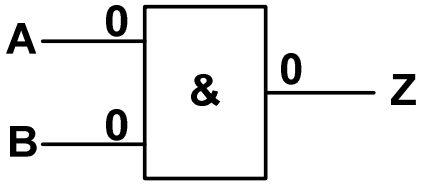


Figure 2.10 Example gate to explain a pattern fault

SSF, *N*-detect (*N* = 2, 3, 5, 10, and 15), transition fault, and GE test sets were applied to the ELF35 test cores. Table 2.2 describes the test sets used in the experiments. The test sets were generated on a Sun-Blade-1000 with the Solaris 8 operating system; the main memory size was 2 GBytes.

Table 2.2 Test sets used in the experiments

Test set	Description
SSF	A test that detects all possible SSFs at least one time
2-detect	A test that detects all possible SSFs at least two times
3-detect	A test that detects all possible SSFs at least three times
5-detect	A test that detects all possible SSFs at least five times
10-detect	A test that detects all possible SSFs at least ten times
15-detect	A test that detects all possible SSFs at least fifteen times
Transition	A test that detects all possible transition faults
GE	A test that observes all observable input combinations to each gate

Table 2.3 reports the number of gates, the number of all gate input combinations, and the number of nonobservable gate input combinations for the six ELF35 cores. The gates are technology library cells used in the implementation of the ELF35 circuits.

Table 2.3 Information to calculate GE coverage: the ELF35 circuit

Core	Number of gates	Number of all gate input combinations	Number of nonobservable gate input combinations
ELF35	54,242	859,766	367,237

In this experiment, 324 defective cores were tested at the rated clock speed, with a test voltage of 3.3V. Table 2.4 reports the experimental results. The number of

patterns for each test set is reported in the second column of Table 2.4. The SSF test coverage, the transition fault test coverage, and the GE test coverage are presented in the subsequent columns, respectively. The SSF (transition fault) test coverage is calculated as the ratio of the number of detected SSFs (transition faults) to the total number of detectable SSFs (transition faults). Some gate input combinations are not observed and are not identified as nonobservable gate input combinations because the detection of the pattern faults is aborted during the ATPG process. Therefore, the GE coverage of the GE test set is lower than 100%. The "Test escapes" column reports the number of defective cores that pass each test set. The results show that the GE test set detected one more defective core than the 15-detect test set and three more defective cores than the transition fault test set. The test generation time is listed in the last column. The test generation time of the GE test set is three times longer than that of the 15-detect test set.

Table 2.4 Experimental results: the ELF35 test chips

Test set	Test length	SSF test coverage	Transition fault test coverage	GE test coverage	Test escapes	Test generation time [sec]
SSF	3,272	99.6 %	-	85.7 %	3	244
2-detect	6,189	99.6 %	-	88.6 %	3	392
3-detect	9,123	99.6 %	-	90.3 %	3	525
5-detect	14,972	99.6 %	-	92.9 %	1	828
10-detect	29,521	99.6 %	-	94.1 %	1	1,526
15-detect	44,227	99.6 %	-	95.9 %	1	2,183
Transition	4,992	-	98.5 %	-	3	506
GE	5,655	99.6 %	-	98.3 %	0	6,843

Figure 2.11 illustrates the relationship between the number of patterns and test escapes of each test set. In this experiment, the GE test set detected all of the defective cores. The figure demonstrates that the GE test set detects more defective cores with considerably fewer test patterns than the *N*-detect test sets.

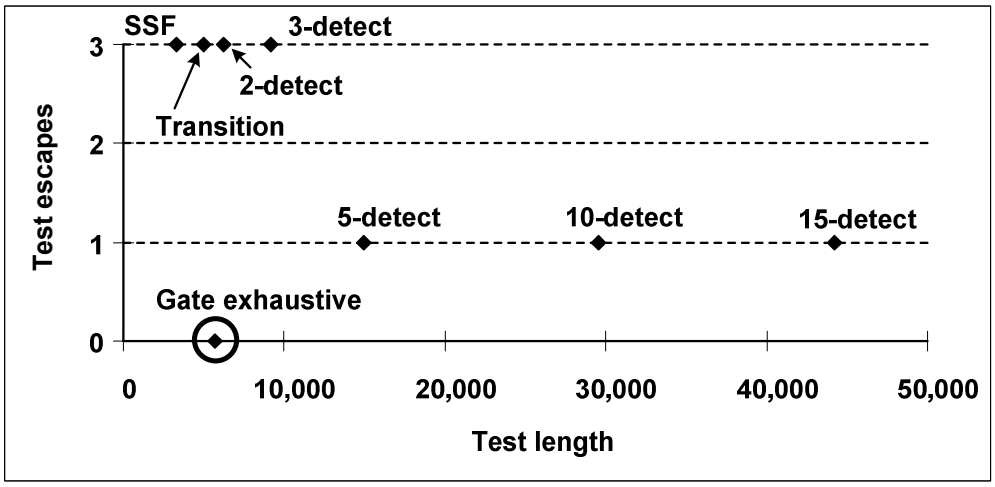


Figure 2.11 Number of test patterns vs. number of test escapes: ELF35

Figure 2.12 illustrates the correlation between the GE coverage and the number of test escapes for each test set. The figure reveals that there is a good correlation between GE coverage and defect detection; i.e., the number of test escapes decreases as the GE coverage of test sets increases. The results also show that the GE coverage of *N*-detect test sets increases with the increase in *N*.

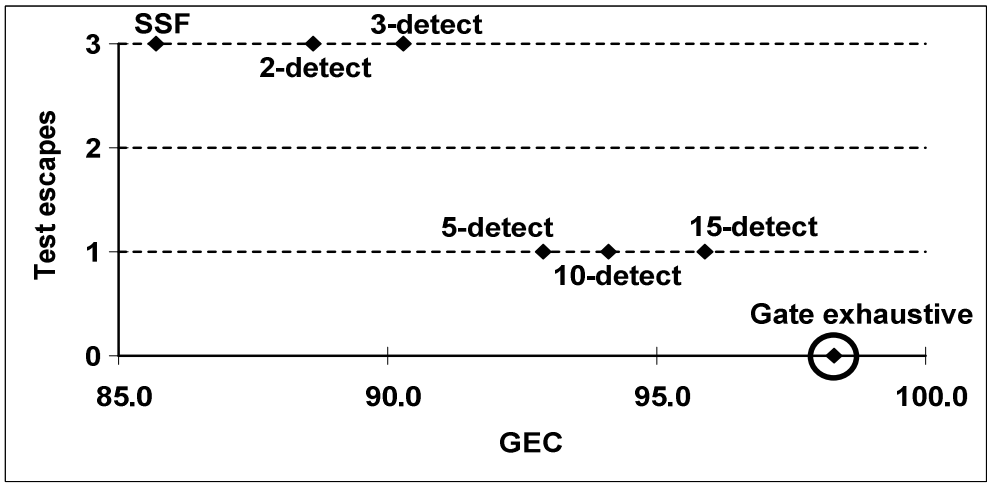


Figure 2.12 Correlation between GE coverage and defect detection: ELF35

2.5 *N*-Detect SSF Simulation

N-detect SSF simulation was conducted on the GE, 5-detect, and 15-detect test sets. Table 2.5 reports the 5-detect and 15-detect SSF simulation results of the three

test sets. *N-detect SSF test coverage* is defined as the ratio of the number of SSFs detected at least *N* times to the total number of detectable SSFs. The results reveal that both the 5-detect and 15-detect test sets are more thorough than the GE test set with respect to either 5-detect or 15-detect test coverage, whereas the GE test set detected more defective cores than the *N*-detect test sets. The results demonstrate that the GE test metric is more effective than the *N*-detect test metric in measuring the thoroughness of test sets.

Table 2.5 Comparison of *N*-detect test coverage

N	Test set		
	GE	5-detect	15-detect
5	76.6%	97.5%	98.9%
15	53.1%	73.5%	96.4%

2.6 Chapter Summary

This chapter presents gate exhaustive (GE) testing; a GE test set applies all the observable gate input combinations to each gate and sensitizes the gate output to an observation point. Experimental results using the Stanford ELF35 test chips demonstrate that the GE test metric is more efficient than either the SSF test metric or the *N*-detect test metric with respect to defect detection and test set size. As outlined in this chapter, the GE test metric avoids several open questions associated with the *N*-detect test metric. Moreover, the GE coverage of test sets shows a good correlation with defect detection; i.e., the number of test escapes decreases with the increase in the GE coverage of test sets.

N-detect simulation results reveal that a GE test set is less thorough than a 5-detect or 15-detect test set with respect to *N*-detect test coverage. The GE test set, however, detects more defective chips than either the 5-detect or 15-detect test set. These results indicate that the GE test metric is more efficient than the *N*-detect test metric in generating test sets and in measuring the thoroughness of test sets.

Chapter 3
Test Set Reordering

This chapter presents test set reordering techniques utilizing the gate exhaustive (GE) and gate super-exhaustive (GSE) test metrics. The efficiency of the presented techniques is compared to that of the traditional techniques using the single stuck-at fault (SSF) and transition fault test metrics.

This chapter is organized as follows: Section 3.1 presents the background of this research; Section 3.2 describes the test set reordering technique utilizing the GE test metric; Section 3.3 presents the test set reordering technique using the GSE test metric; Section 3.4 summarizes this chapter.

This chapter is based on the paper published in the Proceedings of VLSI Test Symposium (VTS) 2007. The published paper is reprinted in Appendix B.

3.1 Background

Production testing cost is closely proportional to test application time and test set size [Hiraide 03]. In order to reduce these values, test set reordering has been studied. The advantage of test set reordering is based on the following: 1) when a test set is too large to fit in a tester memory, the patterns in the later part of the test set can be removed with minimal impact on defect detection [Pomeranz 04]; 2) defective chips fail early on a tester, reducing the test application time for detecting the defective chips [Maly 86][Jiang 01][Pomeranz 04].

Experimental results using industrial chips demonstrate that some SSF test patterns are more effective than others in detecting defects; effective patterns that detect a large number of defective chips are distributed throughout test sets [Nigh 00]. Therefore, test set reordering is an effective technique for reducing test set size with minimal impact on defect detection and for reducing test application time.

An optimal test set reordering technique assumes that the defect occurrence probability and the relationship between defects and faults are available [Maly 86].

However, this technique is not a feasible solution in modern semiconductor designs. A technique that applies all sets of test patterns to a sample set of chips and uses the failure data to reorder the test sets is presented in [Jiang 01]. This technique is not cost efficient because it requires experiments using a tester. The technique also requires a "representative sample" set of chips because it assumes that faulty behavior does not change from lot to lot, which may or may not be true [Madge 04]. In another approach, SSF simulation for each test pattern is performed, and the test patterns are reordered to maximize the SSF coverage of the test set [Lin 01]. The technique presented in this chapter uses a similar technique except that it uses the GE coverage of the test patterns rather than the SSF coverage.

An *original test set* is defined as a test set that has the pattern sequence generated by an ATPG tool. When the pattern sequence of an original test set is changed, the test set is referred to as a *reordered test set*. Test set reordering is conducted to maximize the cumulative test coverage of each test pattern. The *cumulative test coverage of test pattern T* is the coverage calculated for the subset comprising all patterns up to and including test pattern T. Test coverage is calculated using a test metric.

Figure 3.1 illustrates example cumulative fault coverage graphs of an original test set and a reordered test set; the cumulative test coverage is maximized by reordering the original test set. To achieve the same fault coverage, the reordered test set requires fewer test patterns than the original test set. For example, in Fig. 3.1, to achieve 90% coverage, the reordered test set requires 81 test patterns, whereas the original test set requires 414 test patterns. In this chapter, the efficiency of test set reordering techniques is compared by the number of test patterns to be applied to achieve the same defect coverage.

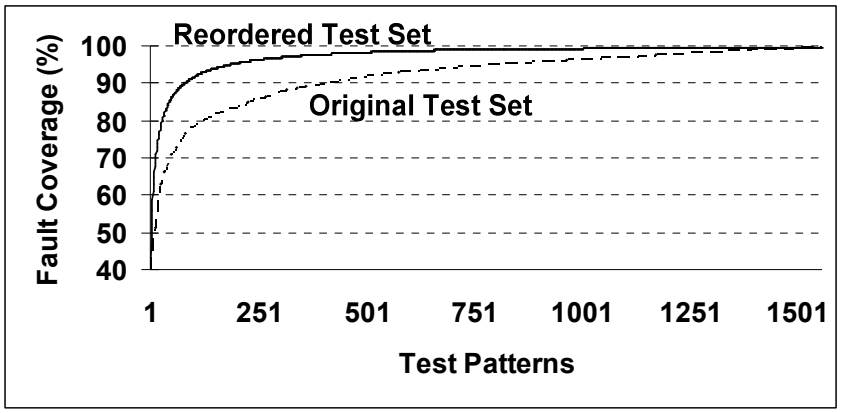


Figure 3.1 Cumulative test coverage

The efficiency of test set reordering depends on the test metric used to calculate the cumulative test coverage. This chapter compares the efficiency of the GE test metric and the SSF test metric in reordering test sets; it also compares the efficiency of the gate super-exhaustive (GSE) test metric and the transition fault test metric.

3.2 Test Set Reordering Using the GE Test Metric

3.2.1 Test Set Reordering

GE simulation was conducted on all the test patterns without dropping observed gate input combinations. After the simulation, the pattern with the highest GE coverage was removed from the original test set and appended to the reordered test set. Then, the gate input combinations observed by the selected pattern were removed from the list of combinations observed by each pattern in the original test set. The process was finished when all the patterns had been reordered. The reordered test set maximizes the cumulative GE coverage for each test pattern.

3.2.2 Experimental Results

The Stanford ELF18 test chips [Brand 04] were used to demonstrate the efficiency of test set reordering techniques. The ELF18 test chips were fabricated using the Philips 0.18 micron Corelib technology; the nominal supply voltage is 1.8V.

More than 70,000 test chips were manufactured; each test chip contains 6 R.E.A.L.™ digital signal processor cores. One ELF18 core contains 13 scan chains; the total number of scan flip-flops is 2,290; the total number of gates for one core is 53,732.

An SSF test set and a GE test set were reordered in the experiments; however, any test set can be reordered. An SSF test set is selected because it is one of the most widely used test sets in VLSI testing; a GE test set is selected because it is the most effective single-pattern test set with respect to defect detection and test set size among the test sets applied to the ELF18 test chips. A *single-pattern test set* consists of patterns that target Boolean or logical defects; each pattern applies one vector to the circuit under test and compares the responses to the expected values. The detailed procedure of GE test set generation is presented in Chapter 2.

Table 3.1 reports the number of patterns, the SSF test coverage, the GE test coverage, and the description of the original test sets.

Table 3.1 Single-pattern test set information: ELF18

Test set	Test length	SSF coverage	GE coverage	Description
SSF	437	99.9%	93.6%	A 100% SSF test set
GE	1,556	99.9%	99.1%	A GE test set

The original test sets were reordered to maximize the cumulative GE test coverage for each additional test pattern. To compare the efficiency of the presented reordering technique, the original test sets were also reordered to maximize the cumulative SSF test coverage. Test set reordering was conducted on a Sun-Fire-V240 workstation running the Solaris 9 operating system. The main memory size was 4 GBytes. Table 3.2 reports the execution time for each test set reordering. The execution time includes the corresponding SSF and GE simulation time.

Table 3.2 Execution time for reordering: single-pattern test sets

Test set	Reordering test metric	Execution time [hour:minute]
SSF	SSF	0:43
	GE	2:45
GE	SSF	7:19
	GE	15:23

The original test sets and reordered test sets were applied to the 140 ELF18 test cores. Each pattern is assigned a number corresponding to its position in the test sequence. The first pattern applied is assigned number 1; the second pattern assigned number 2, etc. The test application was stopped at the first failure, and the first failing pattern number for each defective core was recorded. The first failing pattern number for the i-th defective core is defined as S_i; Table 3.3 reports the maximum value of S_i for the SSF and GE test sets, respectively. The maximum value of S_i represents the number of patterns to be applied to detect all of the defective cores; i.e., the value shows how many patterns could be removed without any impact on defect detection. For the SSF test set, test set reordering techniques using the SSF test metric and the GE test metric reveal similar efficiency. For the GE test set, which is more thorough than the SSF test set, the GE test metric is more efficient than the SSF test metric in reducing the maximum value of S_i.

Table 3.3 Maximum value of S_i: single-pattern test sets

Test set	Number of patterns	Reordering test metric	Maximum $\{S_i\}$
SSF	437	Original	133
		SSF	123
		GE	124
GE	1,556	Original	758
		SSF	739
		GE	286

In combinational circuits, test set reordering may affect the defect detection of test sets due to sequence-dependent defects [McCluskey 04a]. If a chip has *sequence-dependent defects*, the test results of the chip depend on the order of the applied

patterns [Li 02]. The ELF18 core is a full scan circuit, which means that the internal state of the circuit when a test pattern is applied is determined more by the scan shift-in operations than the previous test patterns [Ma 99]. Therefore, sequence dependent behavior was not considered in the experiments.

3.3 Test Set Reordering Using the GSE Test Metric

This section introduces the gate super-exhaustive (GSE) test metric. Thereafter, the test metric is used to reorder two-pattern test sets. A *two-pattern test set* consists of patterns that target delay defects or sequence-dependent defects; each pattern applies two vectors to the circuit under test and compares the responses of the second vector to the expected values.

3.3.1 The GSE Test Metric

In this section, the GSE test metric is introduced because it will be used to reorder two-pattern test sets. The GSE test metric is an extension of the GE test metric to a two-pattern test metric. The following three definitions are used to define the GSE test metric.

Definition 3.1: An *observed pair of gate input combinations* is a pair of logic combinations to a gate, such that the two combinations are applied to the gate inputs on two successive clock pulses with the effect of the second combination being propagated to at least one observation point, such as a primary output or a scan flip-flop.

Definition 3.2: An *observable pair of gate input combinations* is a pair of logic combinations to a gate, such that the two combinations can be applied to the gate inputs on two successive clock pulses with the effect of the second combination being propagated to at least one observation point.

Definition 3.3: A *nonobservable pair of gate input combinations* is a pair of logic combinations to a gate, such that either one or both combinations cannot be applied to the gate inputs on two successive clock pulses or that the effect of the second combination cannot be propagated to any observation point.

In the definitions, note that pairs of logic combinations are considered. Depending on the technique of applying a pair of gate input combinations to a gate, the same pair can be either observable or nonobservable. For example, a pair of gate input combinations can be observable in launch-on-shift [Eichelberger 91][Savir 93], but it may be nonobservable in launch-on-capture [Eichelberger 91][Savir 94], or vice versa.

A pair of gate input combinations can be nonobservable because either one or both combinations cannot be applied to a gate. This is explained using the circuit in Fig. 3.2, which is an implementation of a multiplexer. The (J1, J2) = (1, 1) combination cannot be applied to the inputs of gate J, which is known as a "controllability don't-care" [De Micheli 94]. A *controllability don't-care* is a gate input combination that cannot be applied to the inputs of an internal gate. In this case, the pairs of gate input combinations (00, 11), (01, 11), (10, 11), (11, 11), (11, 00), (11, 01), and (11, 10) are nonobservable pairs to gate J.

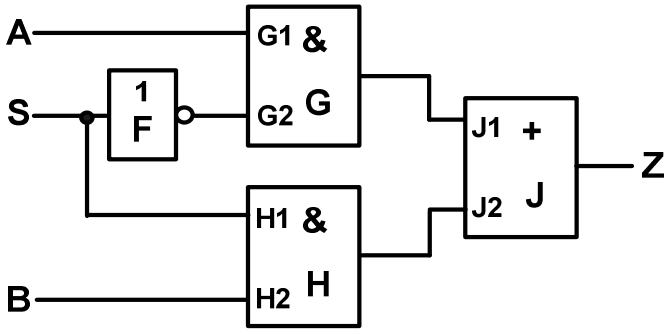


Figure 3.2 Circuit to explain nonobservable pairs of gate input combinations (1)

A pair of gate input combinations can also be nonobservable because the effect of the second combination cannot be propagated to any observation point. Let us consider the circuit illustrated in Fig. 3.3. When the (G1, G2) = (0, 0) combination is applied to the inputs of gate G, logic-1 is assigned to the output of gate H; the logic value blocks the sensitization path from the output of gate G to the circuit output Z. This kind of gate input combination is known as an "observability don't-care" [De Micheli 94]. An *observability don't-care* is a gate input combination that can be applied to an internal gate, but the logic value at the gate output cannot be propagated

to any observation point. In this case, the pairs (00, 00), (01, 00), (10, 00), and (11, 00) are nonobservable pairs of gate input combinations to gate G.

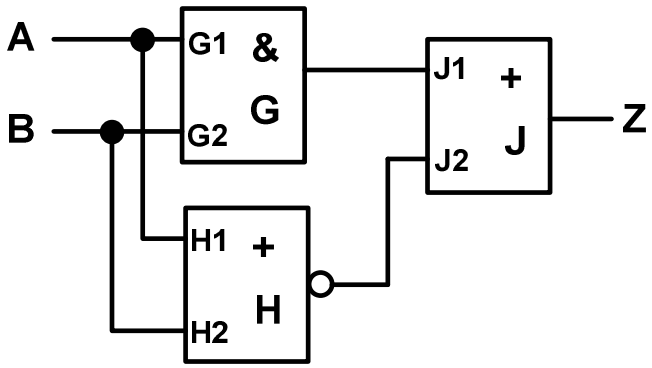


Figure 3.3 Circuit to explain nonobservable pairs of gate input combinations (2)

The *GSE test metric* estimates the thoroughness of a test set by the ratio of the number of pairs of gate input combinations observed by the test set to the total number of observable pairs; the ratio is referred to as the *GSE coverage* of the test set. The number of pairs of gate input combinations increases exponentially with the number of gate inputs. For example, there are 256 (4^4) pairs of gate input combinations to a gate with 4 inputs. A gate can be an elementary gate (e.g., AND, NAND, OR, NOR, or inverter), a complex gate (e.g., AOI, IOA, XOR, Multiplexer, adder, etc.), or a circuit segment (e.g., logical cone); in this research, design library cells are used as gates.

The circuit in Fig. 3.4 is used to illustrate a GSE coverage calculation.

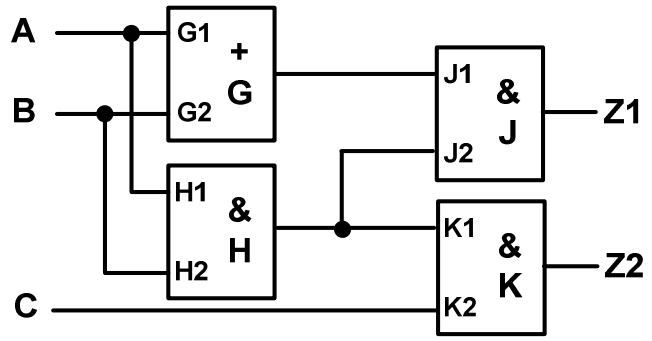


Figure 3.4 Circuit to illustrate a GSE coverage calculation

In the circuit illustrated in Fig. 3.4, there are 19 nonobservable pairs of gate input combinations, which are marked "N.O." in Fig. 3.5. A transition fault test set

that detects all detectable transition faults in the circuit is {(V_1, V_2) | (000, 111), (111, 011), (010, 100), (111, 110)}. In Fig. 3.5, the pairs of gate input combinations observed by the transition fault test set are marked "O" for each gate. The GSE coverage of the transition fault test set is calculated to be 31.1%.

V_1 \ V_2	00	01	11	10
00	N.O.	N.O.	O	N.O.
01	N.O.	N.O.		N.O.
11	N.O.	N.O.	O	N.O.
10	N.O.	N.O.		N.O.

(a)

V_1 \ V_2	00	01	11	10
00			O	
01				O
11		O	O	
10				

(b)

V_1 \ V_2	00	01	11	10
00		N.O.	O	
01	N.O.	N.O.	N.O.	N.O.
11		N.O.	O	O
10		N.O.		O

(c)

V_1 \ V_2	00	01	11	10
00	O		O	
01				
11		O		O
10				

(d)

Figure 3.5 Pairs of gate input combinations observed by a transition fault test set: (a) gate G; (b) gate H; (c) gate J; (d) gate K

3.3.2 Test Set Reordering

GSE simulation was conducted on all the test patterns without dropping observed pairs of gate input combination. After the simulation, the pattern that observes the largest number of pairs of gate input combinations was removed from the original test set and appended to the reordered test set. Thereafter, all the pairs of gate input combinations observed by the selected pattern were removed from the list of the pairs observed by each test pattern in the original test set. This test set reordering process was finished when all the patterns had been reordered. The reordered test set maximizes the cumulative GSE coverage for each test pattern. A similar process was used to reorder test sets utilizing the transition fault test metric, in which the test sets were reordered such that the cumulative transition fault coverage was maximized, rather than the GSE coverage.

3.3.3 Experimental Results

In the experiment, 153 defective ELF18 test chips [Brand 04] were used to compare the efficiency of the test set reordering techniques. A transition fault test set and a 2-detect transition fault test set were reordered. An *N-detect transition fault test set* detects all detectable transition faults at least *N* times using different test patterns. A transition fault test set is selected because it is widely used in production testing. A 2-detect transition fault test set is selected because it is considered as a more thorough test set than a transition fault test set. The test sets were generated using the launch-on-capture technique [Eichelberger 91][Savir 94]. Table 3.4 reports the information of the test sets reordered in this experiment; the test sets were generated using the Synopsys TetraMAX ATPG tool [Synopsys 05] with a maximal compaction option.

Table 3.4 Two-pattern test set information: ELF18

Test set	Transition fault test coverage	Number of patterns	Description
TF	97.24%	1,457	A 100% transition fault test set
TF2det	97.36%	3,060	A 2-detect transition fault test set

Test set reordering was conducted on a Sun-Fire V240 workstation running the Solaris 9 operating system; the main memory size was 4 GBytes. Table 3.5 reports the execution time for test set reordering using the transition fault and GSE test metrics.

Table 3.5 Execution time for reordering: two-pattern test sets

Test set	Reordering test metric	Execution time [hour:minute]
TF	Transition	00:14
	GSE	21:59
TF2det	Transition	00:33
	GSE	43:33

The execution time of the test set reordering technique using the GSE test metric is longer than that using the transition fault test metric. The reason is that the GSE simulation was implemented using a transition fault simulator and PERL scripts. Utilizing better CAD tools and parallel computing can reduce the execution time. In some cases, the long execution time is tolerable because it is a one-time simulation.

The original test sets and reordered test sets were applied to the ELF18 test cores. The launch-capture speed was a speed with a 10% margin from the maximum speed identified from the shmoo plots of defect-free cores. The test application was stopped at the first failure, and the first failing pattern number for each defective core was recorded. The first failing pattern number for the i-th defective core is defined as S_i. Table 3.6 reports the maximum value of S_i. For both test sets, the GSE test metric is more efficient in reducing the maximum value than the transition fault test metric.

Table 3.6 Maximum value of S_i: two-pattern test sets

Test set	Number of patterns	Reordering test metric	Maximum$\{S_i\}$
TF	1,457	Original	618
		Transition fault	398
		GSE	374
TF2det	3,060	Original	1,738
		Transition fault	1,200
		GSE	1,033

3.4 Chapter Summary

Test set reordering techniques using the GE test metric and the GSE test metric are presented; the technique reorders a test set to maximize the cumulative GE coverage or GSE coverage for each pattern. The reordering techniques are applied to test sets generated with maximal compaction and reordering supported by an ATPG tool. Experimental results using the Stanford ELF18 test chips demonstrate that the test sets reordered using the GE test metric and the GSE test metric can be truncated with less impact on defect detection than those reordered utilizing the traditional test metrics, such as the SSF test metric and the transition fault test metric. The presented test set reordering techniques are cost efficient because they can be implemented using computer simulation without the need for any experiment on a tester.

Chapter 4
California Scan Architecture

This chapter presents California scan architecture. The benefits of the scan architecture are 1) improved defect coverage and 2) reduced power consumption during the scan shift operations. This technique is feasible because most of the bits in the test patterns generated by automatic test pattern generation (ATPG) tools are don't-care bits. Simulation results using the ITC'99 benchmark circuits and the ELF18 circuit demonstrate the efficiency of California scan architecture.

This chapter is organized as follows: Section 4.1 presents the background of this chapter; Section 4.2 describes California scan architecture; Section 4.3 reports the implementation and efficiency of the California scan architecture using the ITC'99 benchmark circuits and the ELF18 circuit; Section 4.4 discusses scan shift-out switching activity; Section 4.5 concludes this chapter.

This chapter is based on the paper published in the proceedings of International Test Conference (ITC) 2007. The published paper is reprinted in Appendix C.

4.1 Background

In modern VLSI design, a full-scan design is usually used for better controllability and observability of internal states [Wang 06]. During the scan shift operations, the states of scan flip-flops change, causing many state transitions. These state transitions may cause good chips to fail the applied test set due to abnormal power consumption or excessive power/ground noise compared to normal operation [Nicolici 02][Sinanoglu 03][Yoshida 03][Lee 07]. To reduce switching activity during the scan shift operations, several techniques have been presented: gating logic to prevent the switching activity from propagating to the combinational logic [Gerstendorfer 99][Bhunia 05], activating some of the scan flip-flops at a time [Saxena 01], and modifying the scan path by inserting inverters or XOR gates [Sinanoglu 03].

This chapter presents a cost efficient technique for reducing switching activity during the scan shift operations and for improving the defect coverage of test sets.

ATPG tools generate test patterns with many don't-care bits; it has been reported that 95% to 99% of the bits in compacted test sets for large industrial circuits are don't-care bits [Hiraide 03][Butler 04]. *Don't-care bits* are bits that are not specified in deterministic test patterns generated by ATPG tools. Don't-care bit assignment can be utilized to generate compact test sets [Goel 79][Lambert 96] or to reduce power consumption during testing [Sankaralingam 00]. Examples of don't-care bit assignment are 0-fill, 1-fill, random-fill, repeat-fill, and toggle-fill. The *0-fill* technique assigns logic-0 to each don't-care bit while the *1-fill* technique assigns logic-1. The *repeat-fill* technique assigns don't-care bits using the last care bit. These techniques limit switching activity during the scan shift-in operations, although they may degrade defect coverage. On the other hand, the *random-fill* technique assigns don't-care bits using pseudo-random bits while the *toggle-fill* technique assigns don't-care bits using logic-0 and logic-1 alternately. These techniques can improve defect coverage, although they increase switching activity during the scan shift-in operations.

4.2 California Scan Architecture

California scan architecture (*CSA*) is a minor modification of traditional scan architecture (TSA). Its benefits are 1) reduced power consumption during test pattern scan shift-in and 2) improved defect coverage. These benefits are obtained by applying a modified version, rather than an exact copy, of the scan shift-in pattern to the combinational logic.

This technique is feasible because most of the bits in the test patterns generated by ATPG tools are don't-care bits [Hiraide 03][Butler 04]. To reduce power consumption during the scan shift-in operations, don't-care bits can be assigned using the repeat-fill technique. This assignment strategy, however, may reduce the fortuitous detection of untargeted faults. To enhance fault coverage, don't-care bits can be assigned using the random-fill technique, but this approach increases overall power consumption during the scan shift-in operations. CSA provides a trade-off between

test quality and power consumption by modifying test patterns during the scan shift-in operations.

In CSA, the test patterns that are shifted into scan chains have their don't-care bits assigned using the repeat-fill technique. This technique limits the amount of toggling of scan flip-flops during the scan shift-in operations. The scan shift-in patterns are altered during the scan shift-in operations and converted to toggle-fill patterns when they are applied to the combinational logic. This pattern modification enhances the fault detection of the test patterns [McCluskey 04a]. Table 4.1 shows an example of entering a scan shift-in pattern into a scan chain and of applying an altered pattern to the combinational logic, in which the scan flip-flop 1 is connected to a scan-out pin. In the table, "d" is a don't-care bit. A *test cube* is a deterministic test pattern generated by ATPG tools without assigning don't-care bits [Wang 06].

Table 4.1 Example of test pattern modification

Scan flip-flop	8	7	6	5	4	3	2	1
Test cube	d	1	1	d	d	d	d	0
Scan-in pattern	1	1	**0**	0	0	0	0	0
Applied pattern	0	1	1	0	1	0	1	0

Table 4.2 reports the alteration scheme for applying the patterns. Note that the boldfaced entry in Table 4.1 differs from the value in the test cube since the logic value will be complemented when it is applied to the combinational logic.

Table 4.2 Correspondence between scan shift-in patterns and patterns applied to the combinational logic

Scan-in pattern	Q_8	Q_7	Q_6	Q_5	Q_4	Q_3	Q_2	Q_1
Applied pattern	$\overline{Q}_8$	Q_7	$\overline{Q}_6$	Q_5	$\overline{Q}_4$	Q_3	$\overline{Q}_2$	Q_1

Table 4.3 illustrates the states of scan flip-flops during the scan shift-in operations of the pattern in Table 4.1. The right-most bit in the "Scan-in pattern" column enters the scan chain in each scan clock cycle. During the scan shift-in operations, the repeat-fill pattern is modified to a toggle-fill pattern. Note that the

amount of toggling of each scan flip-flop is limited because don't-care bits are assigned using the repeat-fill technique.

Table 4.3 States of scan flip-flops during the scan shift-in operations

Scan clock	Scan-in pattern	Scan flip-flop							
		8[#]	7	6	5	4	3	2	1[$]
1	1100000 0	1	?	?	?	?	?	?	?
2	110000 0	1	0	?	?	?	?	?	?
3	11000 0	1	0	1	?	?	?	?	?
4	1100 0	1	0	1	0	?	?	?	?
5	110 0	1	0	1	0	1	?	?	?
6	11 0	1	0	1	0	1	0	?	?
7	1 1	0	0	1	0	1	0	1	?
8	1	0	1	1	0	1	0	1	0

[#] Closest to scan-in pin
[$] Closest to scan-out pin

CSA can improve the quality of scan-based test sets, such as SSF and transition fault test sets. Two techniques can be used to detect transition faults in a scan-based circuit: skewed-load, also known as launch-on-shift [Eichelberger 91][Savir 93] and broadside, also known as launch-on-capture [Eichelberger 91][Savir 94]. CSA can be used for any scan flip-flop design, such as muxed-D scan design [Williams 73], level-sensitive scan design (LSSD) [Eichelberger 77], and two-port flip-flop design (clocked-scan design) [McCluskey 86]

4.3 Implementation and Simulation Results

Figure 4.1 illustrates a traditional scan architecture (TSA). Note that the scan-in (SI) input of each scan flip-flop is connected to the Q signal of the previous scan flip-flop. Scan-enable and clock signals are not included in the figure.

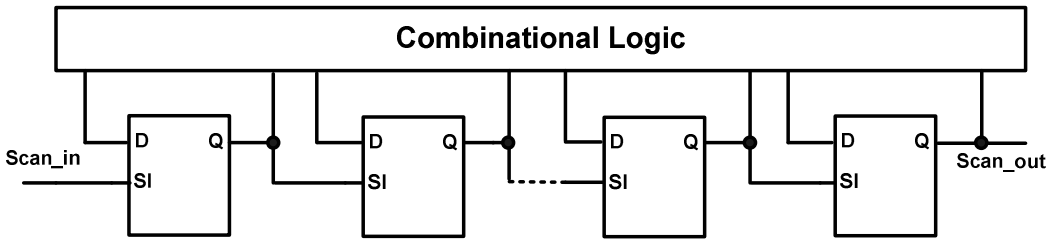


Figure 4.1 A traditional scan architecture

Figure 4.2 illustrates two CSA implementations. Figure 4.2(a) shows a CSA implementation that inserts an inverter at the SI input of each scan flip-flop. Figure 4.2(b) illustrates another implementation that connects the SI input of each scan flip-flop to the $\overline{Q}$ output, rather than the Q output, of the previous scan flip-flop. Both architectures provide similar effectiveness. The selection of the architecture depends on the availability of the $\overline{Q}$ signal of scan flip-flops. CSA does not change the combinational logic of the system.

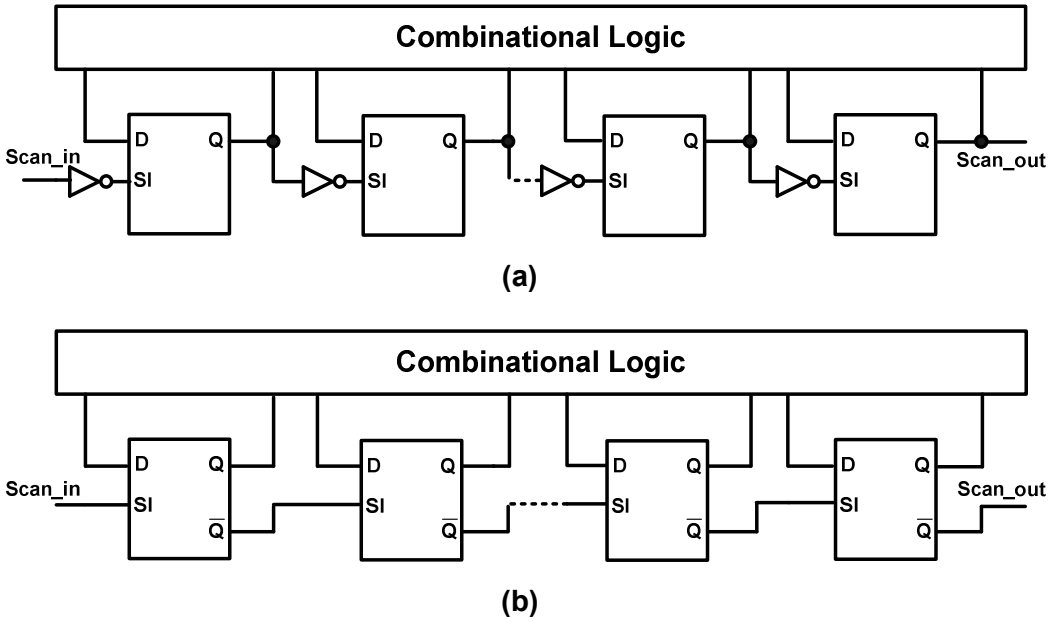


Figure 4.2 California scan architectures: (a) an implementation using inverters; (b) an implementation using the $\overline{Q}$ signals of scan flip-flops

In the CSA implementations, the test patterns applied to the combinational logic differ from the scan shift-in patterns. For example, when the scan shift-in pattern is "000000," the pattern applied to the combinational logic in Fig. 4.2(a) becomes "101010," assuming that the right-most bit is the first bit to enter the scan chain. Similarly, the pattern applied to the combinational logic in Fig. 4.2(b) becomes "010101." One assumption of CSA is that toggle-fill patterns are more effective in detecting defects than repeat-fill patterns.

In this chapter, the b19 circuit, the largest I99T circuit [Corno 00] of the ITC'99 benchmark suite [Basto 00], and the ELF18 circuit [Brand 04] are used to

demonstrate the implementation of CSA and the efficiency of CSA with respect to fault detection and power consumption. The results of other benchmark circuits show similar trends to those of the b19 circuit; they are presented in Appendix C. The synthesis of the benchmark circuits and the scan insertion were conducted using the Synopsys Design Compiler [Synopsys 04]; to simplify the simulation process, only one scan chain was inserted into each circuit. Table 4.4 reports the synthesis results of the b19 circuit: the number of scan flip-flops, primary inputs, and primary outputs. After the circuit is synthesized, a PERL script inserts an inverter at the SI input of each scan flip-flop.

Table 4.4 The b19 circuit information

Circuit	Scan flip-flops	Primary inputs	Primary outputs
b19	6,642	49	31

Table 4.5 defines test sets according to scan architectures and don't-care bit assignment techniques that will be used in this chapter.

Table 4.5 Scan architecture and don't-care bit assignment

Test set	Scan architecture	Don't-care bit assignment
TSA_rpt	Traditional scan	Repeat-fill
TSA_rnd	Traditional scan	Random-fill
CSA_rpt	California scan	Repeat-fill

To demonstrate the efficiency of CSA, power consumption and fault detection are compared. To estimate power consumption, the number of state transitions of scan flip-flops during the scan shift-in and shift-out operations is used; a larger number represents more power consumption. It has been reported that the switching activity of scan flip-flops are closely correlated with that of the combinational logic [Sankaralingam 00]. To estimate capture power consumption, the number of scan flip-flops with different logic values between applied patterns and captured responses is used. The *average switching activity of scan flip-flops per pattern* is defined as the number of state transitions of scan flip-flops during test application divided by the number of patterns. In this research, the average switching activity is used to compare the power consumption during test application.

The defect coverage of a test set is estimated using the average number of SSFs and transition faults detected per pattern and the *N*-detect coverage [Ma 95]. The *average number of faults detected per pattern* is defined as the summation of the number of faults detected by each pattern in a test set divided by the number of patterns. If a pattern detects more faults, more fault sites are observed by the pattern, improving the detection of un-modeled faults, such as bridging faults [Dworak 01]. The use of *N*-detect metric is based on the report that multiple detection of faults improves the defect coverage of the test sets [Ma 95][Grimaila 99] [Amyeen 04]. In this research, only the faults existing in the combinational logic were considered. All the coverage values were calculated as test coverage; i.e., untestable faults were not considered during the fault simulation. In the case of the SSF test sets, the GE coverage of the test sets was also compared. The correlation of GE coverage with defect coverage is presented in Chapter 2.

Figure 4.3 illustrates the simulation flow using an example. The test cube used in the example for a TSA circuit is "00ddddd11," in which "d" represents a don't-care bit. The test cube for the corresponding CSA circuit is "**1**0ddddd**0**1"; the two boldfaced care bits are complemented. In TSA, don't-care bits are assigned using the repeat-fill and random-fill techniques; in CSA, don't-care bits are assigned using only the repeat-fill technique. Assuming that the right-most bit enters the scan chain first, the numbers of state transitions of scan flip-flops during the scan shift-in operations of **TSA_rpt**, **TSA_rnd**, and **CSA_rpt** are 2, 25, and 10, respectively. In this simple example, the switching activity of **CSA_rpt** lies between that of **TSA_rpt** and **TSA_rnd**. Figure 4.3 also presents test patterns applied to the combinational logic. In TSA, all the scan flip-flops are assigned the same logic value as in the scan shift-in pattern; in CSA, every other scan flip-flop is assigned the complemented logic value of the corresponding bit in the scan shift-in pattern. During the test application, the care bits applied to the combinational logic are the same for the three test patterns. Thus, any difference in the fault coverage or in the number of detected faults represents only the effect of don't-care bit assignment.

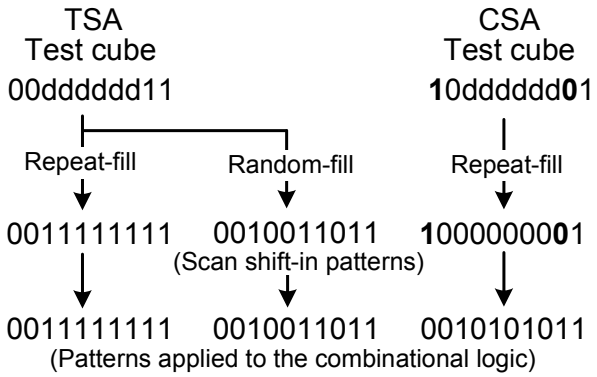


Figure 4.3 Example simulation flow

An SSF test set and a transition fault test set for the b19 circuit were generated using the Synopsys TetraMAX [Synopsys 05] with a maximal compaction option and without assigning don't-care bits. The transition fault test sets were generated using the launch-on-capture technique.

Table 4.6 provides the simulation results of the SSF test sets for the b19 circuit. The first four columns present test set, the number of test patterns, the SSF test coverage, and the GE test coverage, respectively. **CSA_rpt** provides higher GE test coverage than **TSA_rpt**. Thus, **CSA_rpt** is considered more effective than **TSA_rpt** in detecting defective chips. The correlation between GE test coverage and defect coverage is presented in Chapter 2. The "Detected SSFs" column reports the average number of SSFs detected per pattern. The last column of Table 4.6 lists the average switching activity per pattern during the test application. The average activity is calculated from the scan shift-in, capture, and scan shift-out switching activity of scan flip-flops. The "Ratio" column reports the number normalized to that of **TSA_rnd**. In the case of the b19 circuit, the average number of detected SSFs of **CSA_rpt** is 96.7 % that of **TSA_rnd**, whereas the average switching activity is only 15.5 % that of **TSA_rnd**. Figure 4.4 illustrates the average number of SSFs detected per pattern and the average switching activity per pattern in a graph. The switching activity of CSA is close to that of **TSA_rpt**; the average number of SSFs detected per pattern is close to that of **TSA_rnd**. Therefore, the results reveal that CSA can be a good trade-off between fault detection and power consumption.

Table 4.6 Simulation results of SSF test sets: the b19 circuit

Test set	Test length	SSF test coverage	GE test coverage	Detected SSFs		Switching activity	
				Average	Ratio	Average	Ratio
TSA_rpt	821	99.7%	42.6%	20,428	0.823	2,067,661	0.097
CSA_rpt	821	99.7%	43.4%	24,014	0.967	3,298,596	0.155
TSA_rnd	821	99.7%	50.1%	24,834	1.000	21,229,982	1.000
Ratio is the number normalized to that of **TSA_rnd**.							

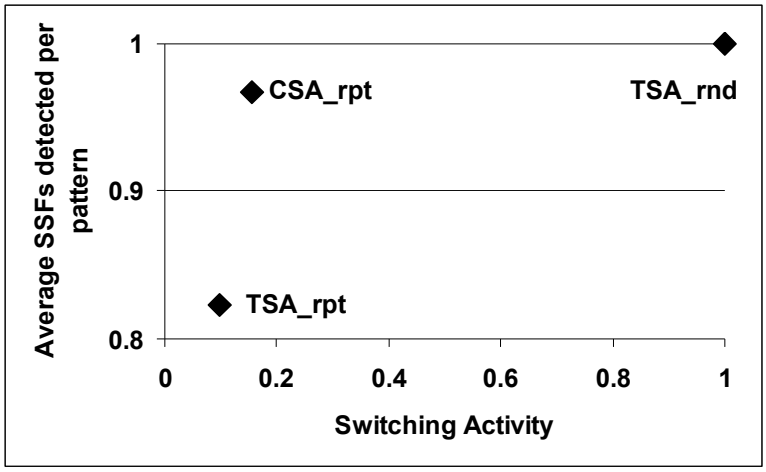


Figure 4.4 Switching activity vs. SSF detection: the b19 circuit

Table 4.7 reports the *N*-detect test coverage of the SSF test sets for the b19 circuit. The overall trends show that the *N*-detect coverage of **CSA_rpt** lies between that of **TSA_rpt** and **TSA_rnd** for the b19 circuit. Therefore, **CSA_rpt** can be considered to be more effective than **TSA_rpt** in detecting defective chips.

Table 4.7 *N*-detect test coverage of SSF test sets: the b19 circuit

Test set	2-det	5-det	10-det	15-det
TSA_rpt	72.4	57.2	45.8	39.4
CSA_rpt	72.4	58.2	46.4	39.8
TSA_rnd	73.9	61.0	50.2	43.7

Table 4.8 provides the simulation results of the transition fault test sets for the b19 circuit. The "Transition fault test coverage" and "Detected transition faults" columns report the transition fault test coverage and the average number of transition faults detected per pattern, respectively. The simulation results demonstrate that CSA is also a good trade-off between power consumption and fault detection for transition

fault test sets. For the b19 circuit, the average number of the detected transition faults of **TSA_rpt** is 85.5% that of **TSA_rnd**, which increases to 94.4% by applying CSA while the power consumption increases from 7.3% to 15.4% that of **TSA_rnd**.

Table 4.8 Simulation results of transition fault test sets: the b19 circuit

Test set	Test length	Transition fault test coverage	Detected transition faults		Switching activity	
			Average	Ratio	Average	Ratio
TSA_rpt	2,714	95.2%	4,114	0.855	1,560,428	0.073
CSA_rpt	2,714	95.2%	4,543	0.944	3,267,048	0.154
TSA_rnd	2,714	95.2%	4,814	1.000	21,234,101	1.000
Ratio is the number normalized to that of **TSA_rnd**.						

Don't-care bits can be assigned during test set generation; this strategy reduces the number of test patterns by detecting untargeted faults fortuitously during the fault simulation of generated patterns. Table 4.9 reports the simulation results of the SSF test sets for the b19 circuit. The numbers normalized to those of **TSA_rnd** are presented in the parentheses. The result reveals that **CSA_rpt** increases the average number of detected SSFs from 83.0% to 97.9% that of **TSA_rnd** while it only increases power consumption from 10.6% to 17.8%. This result demonstrates that CSA is a good trade-off between fault detection and power consumption during test application.

Table 4.9 SSF test generation results with don't-care bits being assigned during test generation: the b19 circuit

Configuration	**TSA_rpt**	**CSA_rpt**	**TSA_rnd**
SSF test coverage	99.7 %	99.7 %	99.7 %
Test length	575	565	591
Average number of SSFs detected per pattern	20,146 (0.830)	23,765 (0.979)	24,283 (1.000)
Average switching activity per pattern	2,246,454 (0.106)	3,772,448 (0.178)	21,188,266 (1.000)

Similar simulations were conducted using the ELF18 circuit. An SSF test set and a transition fault test set for the ELF18 circuit were generated using the Synopsys TetraMAX [Synopsys 05] with a maximal compaction option and without assigning

don't-care bits. The transition fault test sets were generated using the launch-on-capture technique.

Table 4.10 reports the simulation results of the SSF test sets for the ELF18 circuit. For the SSF test sets, the GE test coverage and the bridging fault test coverage of **CSA_rpt** are higher than those of **TSA_rpt**. The bridging fault list was extracted from the ELF18 circuit layout. The trend of the ELF18 simulation results is similar to that of the b19 circuit. One exception is the fact that the average number of the SSFs detected by **CSA_rpt** is larger than that detected by **TSA_rnd**. In the case of the ELF18 circuit, the switching activity of **CSA_rpt** is 47.5% that of **TSA_rnd**. The results show that CSA is a good trade-off between fault detection and test power consumption for the ELF18 circuit.

Table 4.10 Simulation results of SSF test sets: the ELF18 circuit

Test set	Test length	SSF test coverage	GE test coverage	Bridging fault test coverage	Detected SSFs		Switching activity	
					Average	Ratio	Average	Ratio
TSA_rpt	457	99.5%	91.6%	84.8%	9,940	0.821	30,209	0.167
CSA_rpt	457	99.5%	92.6%	86.8%	12,684	1.048	85,737	0.475
TSA_rnd	457	99.5%	93.4%	87.0%	12,102	1.000	180,433	1.000

Table 4.11 reports the *N*-detect test coverage of the SSF test sets for the ELF18 circuit. The *N*-detect coverage of **CSA_rpt** is higher than **TSA_rpt** for the ELF18 circuit. Therefore, **CSA_rpt** can be considered to be more effective than **TSA_rpt** in detecting defective chips.

Table 4.11 *N*-detect test coverage of SSF test sets: the ELF18 circuit

Test set	2-det	5-det	10-det	15-det
TSA_rpt	79.3%	67.7%	55.4%	47.2%
CSA_rpt	79.6%	69.3%	58.7%	52.2%
TSA_rnd	80.2%	70.4%	59.6%	53.3%

Table 4.12 reports the simulation results of the transition fault test sets for the ELF18 circuit. The table reports the transition fault test coverage, average number of transition faults detected per pattern, and average switching activity per pattern. The average number of detected transition faults of **TSA_rpt** is 69.4% that of **TSA_rnd**,

which increases to 99.7% by applying CSA. The power consumption of **CSA_rpt** is 48.5% that of **TSA_rnd**. The results show that **CSA_rpt** provides similar quality as **TSA_rnd**, and the power consumption is less than 50% that of **TSA_rnd**.

Table 4.12 Simulation results of transition fault test sets: the ELF18 circuit

Test set	Test length	Transition fault test coverage	Detected transition faults		Switching activity	
			Average	Ratio	Average	Ratio
TSA_rpt	1,472	97.2%	1,741	0.694	25,765	0.145
CSA_rpt	1,472	97.2%	2,503	0.997	86,166	0.485
TSA_rnd	1,472	97.2%	2,511	1.000	177,501	1.000

4.4 Scan Shift-Out Switching Activity

CSA reduces the switching activity during the scan shift-in operations; the switching activity during the scan shift-out operations depends on the captured patterns. Captured patterns are determined by the applied patterns and the circuit structure. This section investigates the switching activity during the scan shift-out operations.

Table 4.13 reports the average scan shift-in and shift-out switching activity per pattern of the SSF test sets for the b19 circuit. In the case of the b19 circuit, the switching activity of **CSA_rpt** during the scan shift-in operations is 12.3% that of **TSA_rnd** while the switching activity during the scan shift-out operations is 18.8%. Figure 4.5 illustrates the scan shift-in and shift-out switching activity normalized to the total switching activity of **TSA_rnd** during the scan shift operations. The simulation results reveal that the scan shift-out switching activity follows a trend similar to the scan shift-in switching activity for the b19 circuit. In the case of the b19 circuit, a large percentage of scan flip-flops do not change states during the capture period. Therefore, there is a good correlation between scan shift-in switching activity and scan shift-out switching activity.

Table 4.13 Scan shift switching activity of SSF test sets: the b19 circuit

Average switching activity per pattern	**TSA_rpt**	**CSA_rpt**	**TSA_rnd**
Scan-in	782,128 (0.073)	1,328,231 (0.123)	10,764,761 (1.000)
Scan-out	1,284,906 (0.123)	1,969,828 (0.188)	10,464,679 (1.000)
Total	2,067,034 (0.097)	3,298,059 (0.155)	21,229,440 (1.000)

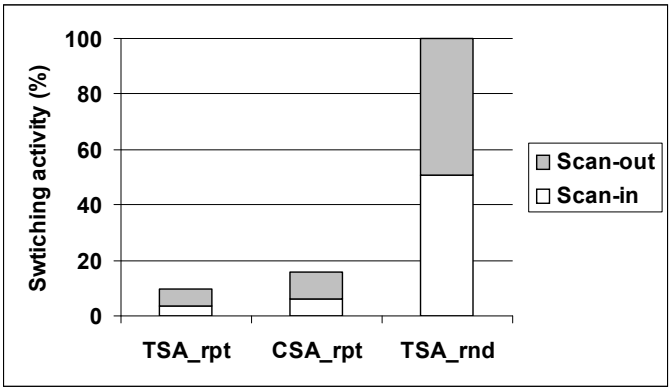


Figure 4.5 Comparison of the switching activity of SSF test sets: the b19 circuit

The switching activity of SSF test sets for the ELF18 circuit is also investigated. Table 4.14 reports the average switching activity per pattern during the scan shift-in and shift-out operations. The numbers normalized to those of **TSA_rnd** are presented in the parentheses. The average switching activity of **CSA_rpt** during the scan shift-out operations is 83.8% that of **TSA_rnd** in the case of the ELF18 SSF test sets. Figure 4.6 illustrates the switching activity normalized to the total switching activity of **TSA_rnd**. In the case of the ELF18 circuit, the switching activity during the scan shift-out operations is not controlled by applying CSA. However, the total scan shift switching activity of **CSA_rpt** is 47.5% that of **TSA_rnd**.

Table 4.14 Scan shift switching activity of SSF test sets: the ELF18 circuit

Average switching activity per pattern	TSA_rpt	CSA_rpt	TSA_rnd
Scan-in	9,428 (0.095)	18,135 (0.182)	99,737 (1.000)
Scan-out	20,781 (0.258)	67,602 (0.838)	80,696 (1.000)
Total	30,209 (0.167)	85,737 (0.475)	180,433 (1.000)

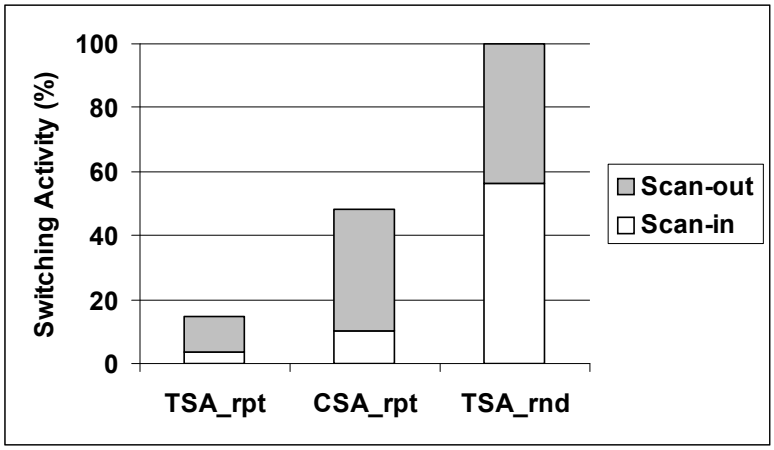


Figure 4.6 Comparison of the switching activity of SSF test sets: the ELF18 circuit

The simulation results of the b19 circuit and the ELF18 circuit show that CSA reduces the scan shift-in switching activity. The results also show that the scan shift-out switching activity depends on the circuit structure. In scan-based testing, the scan shift-in and shift-out operations are conducted concurrently. Therefore, the total switching activity during the scan shift operations can be reduced by reducing the scan shift-in switching activity.

4.5 Conclusions

California scan architecture (CSA) modifies test patterns during the scan shift-in operations to achieve high quality and low power testing. CSA is feasible because most of the bits in the test patterns generated by ATPG tools are don't-care bits. CSA

can be constructed by slightly modifying traditional scan architecture and can be used with only a small modification to current design tools.

Simulation results using ITC'99 benchmark circuits and the ELF18 circuit demonstrate the efficiency of CSA with respect to fault detection and power consumption. The efficiency of CSA may be improved for large industrial circuits because the test sets generated for these circuits contain a higher percentage of don't-care bits than those of the circuits used in this research.

Chapter 5
Concluding Remarks

Test quality and test cost are important issues in VLSI testing. This dissertation presents gate exhaustive (GE) testing, test set reordering, and California scan architecture (CSA). The techniques are more efficient than conventional testing techniques with respect to test quality, test cost, and power consumption during test application.

Techniques to generate compact test sets with high defect coverage have been studied. The SSF test metric has been used to generate compact test sets, whereas the quality of SSF test sets has been shown to be unsatisfactory. To overcome the quality issue of SSF test sets, other techniques can be used, such as *N*-detect and pseudo-exhaustive testing. In Chapter 2, GE testing is demonstrated to be a more efficient technique than the previous ones. Experimental results using the ELF35 test chips reveal that the GE test metric is more efficient than the previous test metrics, such as the SSF test metric and the *N*-detect test metric, in generating high quality test sets and in measuring the thoroughness of test sets. GE test sets can be generated using a commercially available ATPG tool, making GE testing an easier solution than pseudo-exhaustive testing.

As an application of the GE test metric, Chapter 3 presents test set reordering techniques using the GE and gate super-exhaustive test metrics. The gate super-exhaustive test metric is an extension of the GE test metric to a two-pattern test metric. Experimental results using the ELF18 test chips show the efficiency of the techniques in reducing test set size with less impact on defect detection than conventional test metrics, such as the SSF test metric and the transition fault test metric. The techniques can be used in production testing to reduce test cost with minimal impact on test quality.

Power consumption during test application has become an important issue in modern VLSI testing. Chapter 4 presents CSA and demonstrates the efficiency of

CSA with respect to fault detection and power consumption. CSA can be implemented by slightly modifying traditional scan architecture. CSA reduces power consumption and improves fault detection by modifying test patterns during the scan shift-in operations. Moreover, CSA can be used with currently available ATPG tools. Simulation results using benchmark circuits and the ELF18 circuit demonstrate the efficiency of CSA with respect to fault detection and power consumption.

This dissertation presented techniques for improving the efficiency of VLSI testing with respect to test quality, test cost, and power consumption during testing. All the techniques presented in this dissertation can be combined to improve the efficiency of semiconductor chip testing. Experiments and simulation were used to demonstrate the benefit of these techniques.

References

[Abramovici 90] Abramovici, M., M. A. Breuer, and A. D. Friedman, *Digital Systems Testing and Testable Design*, IEEE Press, New York, 1990.

[Acken 83] Acken, J. M., "Testing for Bridging Faults (Shorts) in CMOS Circuits," *Proc. Design Automation Conf.*, pp. 717-718, 1983.

[Agrawal 95] Agrawal, V. D. and T. J. Chakraborty, "High-Performance Circuit Testing with Slow-Speed Testers," *Proc. Intl. Test Conf.*, pp. 302-310, 1995.

[Al-Yamani 05] Al-Yamani, A., E. Chmelar, and M. Grinchuck, "Segmented Addressable Scan Architecture," *Proc. VLSI Test Symp.*, pp. 405-411, 2005.

[Amyeen 04] Amyeen, M. E., S. Venkataraman, A. Ojha, and S. Lee, "Evaluation of the Quality of N-Detect Scan ATPG Patterns on a Processor," *Proc. Intl. Test Conf.*, pp. 669-678, 2004.

[Archambeau 84] Archambeau, E. C., "Fault Coverage of Pseudo-Exhaustive Testing," *Proc. Intl. Symp. Fault-Tolerant Computing*, pp. 141-145, 1984.

[Arslan 04] Arslan, B. and A. Orailoglu, "CircularScan: A Scan Architecture for Test Cost Reduction," *Proc. Design Automation and Test in Europe*, pp. 1290-1295, 2004.

[Basto 00] Basto, L., "First Results of ITC'99 Benchmark Circuits," *IEEE Design & Test of Computers*, vol. 17, no. 3, pp. 54-59, Jul.-Sep. 2000.

[Benware 03] Benware, B., C. Schuermyer, N. Tamarapalli, and K. H. Tsai, "Impact of Multiple-Detect Test Patterns on Product Quality," *Proc. Intl. Test Conf.*, pp. 1031-1040, 2003.

[Bhunia 05] Bhunia, S., H. Mahmoodi, D. Ghosh, S. Mukhopadhyay, and K. Roy, "Low-Power Scan Design Using First-Level Supply Gating," *IEEE Trans. VLSI Systems*, vol. 13, no. 3, pp. 384-395, Mar. 2005.

[Blanton 03] Blanton, R. D., K. N. Dwarakanath, and A. B. Shah, "Analyzing the Effectiveness of Multiple-Detect Test Sets," *Proc. Intl. Test Conf.*, pp. 876-885, 2003.

[Brand 04] Brand, K. A., S. Mitra, E. H. Volkerink, and E. J. McCluskey, "Speed Clustering of Integrated Circuits," *Proc. Intl. Test Conf*., pp. 1128-1137, 2004.

[Bushnell 00] Bushnell, M. L. and V. D. Agrawal, *Essentials of Electronic Testing for Digital, Memory & Mixed-Signal VLSI Circuits*, Kluwer Academic Publishers, 2000.

[Butler 04] Butler, K. M., et al., "Minimizing Power Consumption in Scan Testing: Pattern Generation and DFT Techniques," *Proc. Intl. Test Conf*., 355-364, 2004.

[Cadence 03] Cadence Design Systems, Inc., "Encounter Test Design Edition User Guide," Oct. 2003.

[Carter 87] Carter, J. L., V. S. Iyengar, and B. K. Rosen, "Efficient Test Coverage Determination for Delay Faults," *Proc. Intl. Test Conf*., pp. 418-427, 1987.

[Cho 05] Cho, K. Y., S. Mitra, and E. J. McCluskey, "Gate Exhaustive Testing," *Proc. Intl. Test Conf*., Paper 31.3, 2005.

[Cho 07a] Cho, K. Y. and E. J. McCluskey, "Test Set Reordering Using the Gate Exhaustive Test Metric," *Proc. VLSI Test Symp*., pp. 199-204, 2007.

[Cho 07b] Cho, K. Y., S. Mitra, and E. J. McCluskey, "California Scan Architecture for High Quality and Low Power Testing," *Proc. Intl. Test Conf*., Paper. 25.3, 2007.

[Corno 00] Corno, F., M. S. Reorda, and G. Squillero, "RT-level ITC'99 Benchmarks and First ATPG Results," *IEEE Design & Test of Computers*, vol. 17, no. 3, Jul.-Sep. 2000.

[Cusey 97] Cusey, J. P. and J. H. Patel, "BART: A Bridging Fault Test Generator for Sequential Circuits," *Proc. Intl. Test Conf*., pp. 838-847, 1997.

[Datta 04] Datta, R., R. Gupta, A. Sebastine, J. A. Abraham, and M. d'Abreu, "Tri-Scan: A Novel DFT Technique for CMOS Path Delay Fault Testing," *Proc. Intl. Test Conf*., pp. 1118-1127, 2004.

[De Micheli 94] De Micheli, G., *Synthesis and Optimization of Digital Circuits*, McGraw-Hill, 1994.

[Dworak 01] Dworak, J., et al., "Defect-Oriented Testing and Defective-Part-Level Prediction," *IEEE Design & Test of Computers*, vol. 18, no. 1, pp. 31-41, Jan.-Feb. 2001.

[Dworak 04] Dworak, J., B. Cobb, J. Wingfield, and M. R. Mercer, "Balanced Excitation and its Effect on the Fortuitous Detection of Dynamic Defects," *Proc. Design Automation & Test in Europe*, vol. 2, pp. 1066-1071, Feb. 2004.

[Eichelberger 77] Eichelberger, E. B. and T. W. Williams, "A Logic Design Structure for LSI Testability," *Proc. Design Automation Conf.*, pp. 462-468, 1977.

[Eichelberger 91] Eichelberger, E. B., E. Lindbloom, J. A. Waicukauski, and T. W. Williams, "Delay-Fault Simulation," *Structured Logic Testing*, E. J. McCluskey (Ed.), Prentice-Hall, Inc.: Englewood Cliffs, NJ, 1991.

[Eldred 59] Eldred, R. D., "Test Routines Based on Symbolic Logical Statements," *Journal of the ACM*, vol. 6, no. 1, pp. 33-36, 1959.

[Gerstendorfer 99] Gerstendorfer, S. and H.-J. Wunderlich, "Minimized Power Consumption for Scan-Based BIST," *Proc. Intl. Test Conf.*, pp. 77-84, 1999.

[Goel 79] Goel, P. and B. C. Rosales, "Test Generation & Dynamic Compaction of Tests," *Proc. Intl. Test Conf.*, pp. 189-192, 1979.

[Grimaila 99] Grimaila, M. R., et al., "REDO - Random Excitation and Deterministic Observation – First Commercial Experiment," *Proc. VLSI Test Symp.*, pp. 268-274, 1999.

[Hamzaoglu 99] Hamzaoglu, I. and J. H. Patel, "Reducing Test Application Time for Full Scan Embedded Cores," *Proc. Intl. Symp. Fault-Tolerant Computing*, pp. 260-267, 1999.

[Hamzaoglu 00] Hamzaoglu, I. and J. H. Patel, "Test Set Compaction Algorithms for Combinational Circuits," *IEEE Trans. CAD*, vol. 19, no. 8, pp. 957-963, Aug. 2000.

[Heragu 96] Heragu, K, J. H. Patel, and V. D. Agrawal, "Segment Delay Faults: A New Fault Model," *Proc. VLSI Test Symp.*, pp. 32-39, 1996.

[Hiraide 03] Hiraide, T., et al., "BIST-Aided Scan Test – A New Method for Test Cost Reduction," *Proc. VLSI Test Symp.*, pp. 359-364, 2003.

[Jiang 01] Jiang, W. and B. Vinnakota, "Defect-Oriented Test Scheduling," *IEEE Trans. VLSI Systems*, vol. 9, no. 3, pp. 427-438, Jun. 2001.

[Lambert 96] Lambert, T. J. and K. K. Saluja, "Methods for Dynamic Test Vector Compaction in Sequential Test Generation," *Proc. Intl. Conf. VLSI Design*, pp. 166-169, 1996.

[Lee 02] Lee, S., B. Cobb, J. Dworak, M. R. Grimaila, and M. R. Mercer, "A New ATPG Algorithm to Limit Test Set Size and Achieve Multiple Detections of all Faults," *Proc. Design Automation and Test in Europe*, pp. 94-99, 2002.

[Lee 07] Lee, D., E. Volkerink, I. Park, and J. Rearick, "Empirical Validation of Yield Recovery Using Idle-Cycle Insertion," *IEEE Design & Test of Computers*, vol. 24, no. 4, pp. 362-372, Apr. 2007.

[Li 02] Li, J. C.-M and E. J. McCluskey, "Diagnosis of Sequence-dependent Chips," *Proc. VLSI Test Symp*., pp. 187-192, 2002.

[Lin 01] Lin, X, J. Rajski, I. Pomeranz, and S. M. Reddy, "On Static Test Compaction and Test Pattern Ordering for Scan Designs," *Proc. Intl. Test Conf*., pp. 1088-1097, 2001.

[Ma 95] Ma, S. C., P. Franco, and E. J. McCluskey, "An Experimental Test Chip to Evaluate Test Techniques Experimental Results," *Proc. Intl. Test Conf*., pp. 663-672, 1995.

[Ma 99] Ma, S., I. Shaik, and R. S. Fetherston, "A Comparison of Bridging Fault Simulation Methods," *Proc. Intl. Test Conf*., pp. 587-595, 1999.

[Madge 04] Madge, R., et al., "In Search of the Optimum Test Set – Adaptive Test Methods for Maximum Defect Coverage and Lowest Test Cost," *Proc. Intl. Test Conf*., pp. 203-212, 2004.

[Maly 86] Maly, W., "Optimal Order of the VLSI IC Testing Sequence," *Proc. Design Automation Conf*., pp. 560-566, 1986.

[McCluskey 81] McCluskey, E. J. and S. Bozorgui-Nesbat, "Design for Autonomous Test," *IEEE Trans. Computers*, vol. C-30, no. 11, pp. 866-875, Nov. 1981.

[McCluskey 84] McCluskey, E. J., "Verification Testing – A Pseudoexhaustive Test Technique," *IEEE Trans. Computers*, vol. C-33, no. 6, pp. 541-546, Jun. 1984.

[McCluskey 86] McCluskey, E. J., *Logic Design Principles: With Empahsis on Testable Semicustom Circuits*, Prentice Hall, Inc.: Englewood Cliffs, NJ, 1986.

[McCluskey 93] McCluskey, E. J., "Quality and Single-Stuck Faults," *Proc. Intl. Test Conf*., p. 597, 1993.

[McCluskey 00] McCluskey, E. J. and C.-W. Tseng, "Stuck-Fault Tests vs. Actual Defects," *Proc. Intl. Test Conf*., pp. 336-343, 2000.

[McCluskey 04a] McCluskey, E. J., et al., "ELF-Murphy Data on Defects and Test Sets," *Proc. VLSI Test Symp*., pp. 16-22, 2004.

[McCluskey 04b] McCluskey, E. J., "Digital IC Testing for Art Historians and Test Experts," http://crc.stanford.edu/iccd_ks.pdf (Intl. Conf. Computer Design presentation), 2004.

[Mei 74] Mei, K. C. Y., "Bridging and Stuck-At Faults," *IEEE Trans. Computers*, vol. C-23, no. 7, pp. 720-727, Jul. 1974.

[Moore 65] Moore, G. E., "Cramming More Components onto Integrated Circuits," *Electronics Magazine*, vol. 38, no. 8, pp. 114-117, Apr. 1965.

[Nicolici 02] Nicolici, N. and B. M. Al-Hashimi, "Multiple Scan Chains for Power Minimization during Test Application in Sequential Circuits," *IEEE Trans. Computers*, vol. 51, no. 6, pp. 721-734, Jun. 2002.

[Nigh 00] Nigh, P. and A. Gattiker, "Test Method Evaluation Experiments & Data," *Proc. Intl. Test Conf*., pp. 454-463, 2000.

[Pandey 02] Pandey, A. R. and J. H. Patel, "Reconfiguration Technique for Reducing Test Time and Test Data Volume in Illinois Scan Architecture Based Designs," *Proc. VLSI Test Symp*., pp. 9-15, 2002.

[Park 05] Park, I., A. Al-Yamani, and E. J. McCluskey, "Effective TARO pattern generation," *Proc. VLSI Test Symp*., pp. 161-166, 2005.

[Pomeranz 03] Pomeranz, I. and S. M. Reddy, "On Test Data Compression and n-Detection Test Sets," *Proc. Design Automation Conf*., pp. 748-751, 2003.

[Pomeranz 04] Pomeranz, I. and S. M. Reddy, "On Maximizing the Fault Coverage for a Given Test Length Limit in a Synchronous Sequential Circuit," *IEEE Trans. Computers*, vol. 53, no. 9, pp. 1121-1133, Sep. 2004.

[Rosinger 04] Rosinger, P., B. M. Al-Hashimi, and N. Nicolici, "Scan Architecture with Mutually Exclusive Scan Segment Activation for Shift- and Capture-Power Reduction," *IEEE Trans. CAD*, vol. 23, no. 7, pp. 1142-1153, Jul. 2004.

[Sankaralingam 00] Sankaralingam, R., R. R. Oruganti, and N. A. Touba, "Static Compaction Techniques to Control Scan Vector Power Dissipation," *Proc. VLSI Test Symp*., pp. 35-40, 2000.

[Savir 93] Savir, J. and S. Patil, "Scan-Based Transition Test," *IEEE Trans. CAD*, vol. 12, no. 8, pp. 1232-1241, Aug. 1993.

[Savir 94] Savir, J. and S. Patil, "On Broad-Side Delay Test," *Proc. VLSI Test Symp*., pp. 284-290, 1994.

[Saxena 01] Saxena, J., K. M. Butler, and L. Whetsel, "An Analysis of Power Reduction Techniques in Scan Testing," *Proc. Intl. Test Conf*., pp. 670-677, 2001.

[Shashaani 99] Shashaani, M. and M. Sachdev, "A DFT Technique for High Performance Circuit Testing," *Proc. Intl. Test Conf*., pp. 276-285, 1999.

[Shperling 87] Shperling, I. and E. J. McCluskey, "Circuit Segmentation for Pseudo-Exhaustive Testing via Simulated Annealing," *Proc. Intl. Test Conf*., pp. 58-66, 1987.

[Sinanoglu 03] Sinanoglu, O. and A. Orailoglu, "Modeling Scan Chain Modifications For Scan-in Test Power Minimization," *Proc. Intl. Test Conf*., pp. 602-611, 2003.

[Smith 85] Smith, G. L., "Model for Delay Faults Based upon Paths," *Proc. Intl. Test Conf*., pp. 342-349, 1985.

[Synopsys 04] Synopsys, Inc., "Design Compiler User Guide," Dec. 2004.

[Synopsys 05] Synopsys, Inc., "TetraMAX ATPG User Guide," Jan. 2005.

[Tseng 01a] Tseng, C.-W., S. Mitra, S. Davidson, and E. J. McCluskey, "An Evaluation of Pseudo Random Testing for Detecting Real Defects," *Proc. VLSI Test Symp*., pp. 404-409, 2001.

[Tseng 01b] Tseng, C.-W. and E. J. McCluskey, "Multiple-Output Propagation Transition Fault Test," *Proc. Intl. Test Conf*., pp. 358-366, 2001.

[Venkataraman 04] Venkataraman, S., S. Sivaraj, E. Amyeen, S. Lee, A. Ojha, and R. Guo, "An Experimental Study of N-Detect Scan ATPG Patterns on a Processor," *Proc. VLSI Test Symp*., pp. 23-29, 2004.

[Waicukauski 87] Waicukauski, J. A., E. Lindbloom, B. K. Rosen, and V. S. Iyengar, "Transition Fault Simulation," *IEEE Design & Test Computers*, vol. 4, no. 2, pp. 32-38, Apr. 1987.

[Wang 06] Wang, L.-T., C.-W. Wu, and X. Wen (Eds.), *VLSI Test Principles and Architectures*, Morgan Kaufmann Publishers, San Francisco, CA, 2006.

[Williams 73] Williams, M. J. Y. and J. B. Angell, "Enhancing Testability of Large-Scale Integrated Circuits via Test Points and Additional Logic," *IEEE Trans. Computers*, vol. C-22, no. 1, pp. 46-60, Jan. 1973.

[Yoshida 03] Yoshida, T. and M. Watati, "A New Approach for Low Power Scan Testing," *Proc. Intl. Test Conf*., pp. 480-487, 2003.

Appendix A

Gate Exhaustive Testing

Proceedings of International Test Conference (ITC), Paper 31.3, 2005.

Gate Exhaustive Testing

Kyoung Youn Cho[1], Subhasish Mitra[1,2], and Edward J. McCluskey[1]
[1]Center for Reliable Computing Stanford University, Stanford CA
[2]Intel Corporation, Folsom, CA

Abstract

A gate exhaustive test set applies all possible input combinations to each gate in a combinational circuit, and observes the gate response at an observation point such as a primary output or a scan cell. In this paper, we analyze the effectiveness of the gate exhaustive test metric in detecting defective chips, and compare it with the single stuck-at fault, the N-detect, and the transition fault test metrics. Results from the Stanford CRC ELF35 and ELF18 test experiments show that gate exhaustive test sets are more efficient than single stuck-at and N-detect test sets in terms of the ability to detect defective chips and test length. It is also shown that test sets with higher values of the gate exhaustive coverage have better test quality.

1. Introduction

Test chip experiments over several technology generations have clearly demonstrated the need for new test metrics for grading test sets and for automatic test pattern generation (ATPG) [McCluskey 00][McCluskey 04]. A *test metric* is a measure of the completeness of a test, and is also used to guide test pattern generation. Examples of test metrics include the single stuck-at fault (SSF) test metric, the *N*-detect test metric [Ma 95][McCluskey 00], the transition test metric [Waicukauski 87] and the TARO test metric [Tseng 01a]. A *fault model* is a representation of the effects of defects on chip behaviors. A fault model may be described at logic, circuit, or physical levels of abstraction. Examples of fault models include stuck-at faults, bridging faults, stuck-open faults, and path delay faults.

Traditionally, the SSF test metric has been extensively used because it is easy to use, and many ATPG tools support the SSF model. However, it is demonstrated in [Ma 95][McCluskey 00][McCluskey 04] that *N*-detect test sets are more effective than 100% SSF test sets in detecting defective chips. An *N-detect test set* is defined as a test set in which each stuck-at fault is detected either by *N* "different" tests or by the maximum number of different tests if it is impossible to find such *N* different tests [Ma 95][McCluskey 00]. Recently, several papers have discussed the use of the *N*-detect test metric for industrial designs such as ASICs and microprocessors [Benware 03][Venkataraman 04].

There are three fundamental issues related to the use of the *N*-detect test metric. First, what value of *N* should be used? For the Murphy experiment, a 5-detect test set detected all defective cores [Ma 95][McCluskey 00]. However, in the Stanford CRC ELF35 [McCluskey 04] and ELF18 [Mitra 04] experiments, 15-detect test sets did not detect all defective cores. The second issue with *N*-detect tests is test length. It was observed that the test lengths of *N*-detect test sets grow approximately linearly with *N* [Pomeranz 03][Venkataraman 04]. Finally, the open question with *N*-detect test metric is: how "different" should the different test patterns targeting a fault *N* times be? This issue has been addressed in [Tseng 01b][Blanton 03][Dworak 04].

The above discussion indicates that we should continue to search for better test metrics. One candidate test metric is the gate exhaustive test metric, described in [McCluskey 93]. A *gate exhaustive test set* is defined as a test set that applies all possible input combinations to each gate and observes the gate response at an observation point such as a primary output or a scan cell. The gates can be elementary gates, complex gates, or circuit segments. With, at most, a single bad gate, all Boolean (logical) faults would be detected (unless some gate inputs are redundant) [McCluskey 93]. The effectiveness of a gate exhaustive test set in detecting various kinds of faults such as bridging faults and transistor stuck-on faults is demonstrated by simulation [Blanton 97].

In this work, we compare the test escapes and the test lengths of 100% SSF, *N*-detect, transition, and gate exhaustive test sets for actual chips used in the Stanford

CRC ELF35 and ELF18 test experiments. We also show the correlation between the gate exhaustive coverage metric and the ability of a test set to detect defective cores.

The rest of this paper is organized as follows. Section 2 introduces the background material about gate exhaustive testing. The concept of the gate exhaustive test metric and the calculation of gate exhaustive coverage metric is presented in Sec. 3. In Sec. 4, we report experimental results to support the effectiveness of gate exhaustive test sets. Section 5 presents the *N*-detect stuck-at fault simulation results of gate exhaustive tests, and Sec. 6 concludes the paper.

2. Background

Exhaustive testing of a combinational circuit applies all possible input combinations to the inputs of the circuit under test (CUT), and it ensures detection of all irredundant combinational defects (defects that do not introduce additional states) in the circuit. The major problem with an exhaustive test set is the number of test patterns.

The test length issue of exhaustive test sets can be alleviated by pseudoexhaustive testing technique [McCluskey 84]. Pseudoexhaustive testing partitions a circuit into segments such that the number of inputs of every segment is significantly smaller than the number of primary inputs of the circuit. Exhaustive testing is performed for each segment. However, it is difficult and computationally intensive to find the optimum partitions because the problem is NP-complete [Shperling 87].

In gate exhaustive test set generation, the segment is reduced to each gate in the CUT. The gates can be elementary gates (e.g., AND, OR, NAND, NOR, inverter), complex gates (e.g., XOR, multiplexer, adder, etc.), or circuit segments. Exhaustive input combinations are applied to each gate and the gate response is observed at some observation points. So, the gate exhaustive test metric does not use fault models to generate test patterns.

Note that chips with sequence dependent defects may not be detected by gate exhaustive testing technique. If a chip has sequence dependent defects, the test results of the chip depend on the order of the patterns [Li 02].

3. The gate exhaustive test metric

In order to compare the effectiveness of test sets we develop the gate exhaustive coverage metric. We need the following definitions for that purpose.

Definition 1: An *observed input combination* of a gate is an input combination that is applied to the gate inputs with the gate output being sensitized to at least one observation point such as a primary output or a scan cell.

Definition 2: An *observable input combination* of a gate is an input combination that can be applied to the gate inputs with the gate output being sensitized to at least one observation point.

Definition 3: A *nonobservable input combination* of a gate is an input combination that cannot be applied to the gate inputs with the gate output being sensitized to at least one observation point.

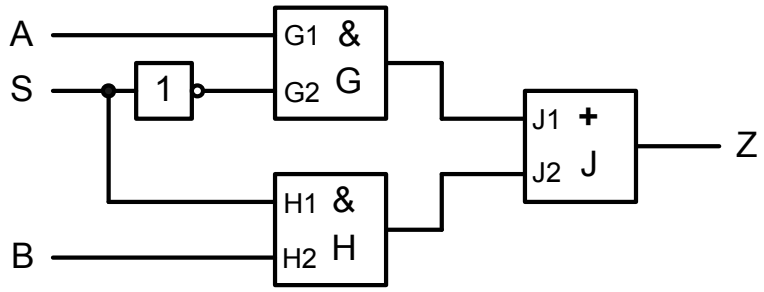
Definition 4: The *gate exhaustive coverage* (GEC) of a test set is defined as the ratio of the total number of all distinct observed input combinations of all gates in the circuit to the total number of all distinct observable input combinations of all gates in the circuit.

In general, we may not be able to apply all possible input combinations to an internal gate and observe its response at some observation points [McCluskey 93]. The circuits in Fig. 1 are used to show examples of nonobservable gate input combinations.

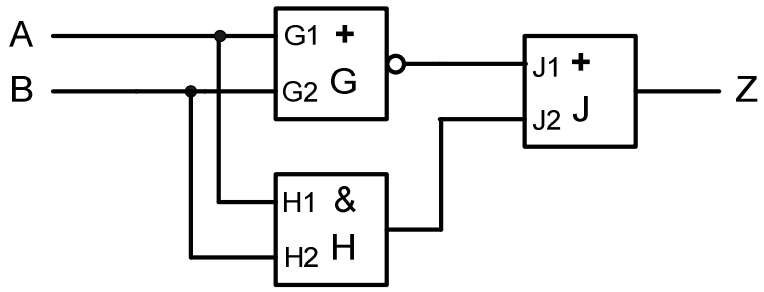
The circuit in Fig. 1 (a) is an implementation of a multiplexer, and it is used to show a gate input combination that cannot be applied to the inputs of a gate. In the circuit in Fig. 1 (a), the (J1, J2) = (1, 1) combination cannot be applied to the inputs of gate J.

Figure 1 (b) is used to show an example of a gate input combination that cannot be sensitized to any observation point. When the (H1, H2) = (0, 0) combination

is applied to the inputs of gate H, the output of gate H cannot be sensitized to the output of the circuit (Z) because the sensitization path is blocked by gate J.



(a) Example circuit to show a gate input combination that cannot be applied to a gate



(b) Example circuit to show a gate input combination that cannot be sensitized to any observation point

Figure 1 Example circuits to show nonobservable gate input combinations

Expression 1 shows the calculation of the GEC. In Expression 1, N is the total number of gates in the CUT. Function $f_i(j)$ is defined as 1 if and only if the binary combination corresponding to the decimal number j is applied to the inputs of gate i and at the same time the response of gate i is observed at some observation points. n_i is the number of the inputs of gate i and NO_i is the number of the nonobservable input combinations of gate i

$$\mathrm{GEC} = \frac{\sum_{i=1}^{N}\sum_{j=0}^{2^{n_i}-1} f_i(j)}{\sum_{i=1}^{N}(2^{n_i} - NO_i)} \times 100 \qquad (1)$$

The circuit in Fig. 2 is used to illustrate the concept of the gate exhaustive testing and the calculation of the GEC.

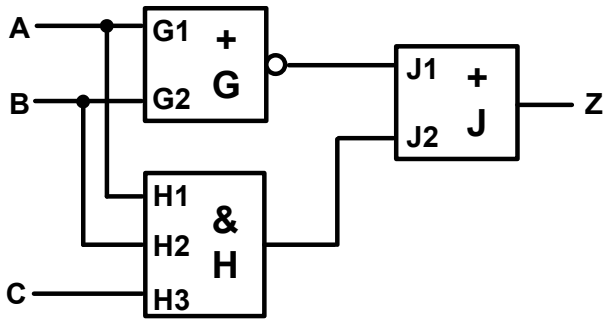


Figure 2 Example circuit to illustrate the gate exhaustive testing and the GEC

For the circuit in Fig. 2, the (J1, J2) = (1, 1) combination and the (H1, H2, H3) = (0, 0, 0) and (0, 0, 1) combinations are nonobservable gate input combinations.

One 100% SSF test set is {(A, B, C) | (0, 0, 0), (1, 1, 1), (1, 0, 1), (1, 1, 0), (0, 1, 0), (0, 1, 1)}. The SSF test set applies all possible input combinations to the inputs of gate G with the output of gate G being sensitized to Z. But it does not apply the (H1, H2, H3) = (1, 0, 0) combination to the inputs of gate H. Table 1 shows the number of all gate input combinations, the number of all nonobservable gate input combinations, and the number of observed gate input combinations in the circuit. From the data in Table 1, the GEC for this SSF test set is calculated to be 92.3% (= (4 + 5 + 3) / (4 + (8 – 2) + (4 – 1)) × 100).

Table 1 Table to calculate the GEC for the circuit in Fig. 2

Gate	Number of all gate input combinations	Number of nonobservable gate input combinations	Number of observed gate input combinations	
			100% SSF	Gate exhaustive
G	4	0	4	4
H	8	2	5	6
J	4	1	3	3

A gate exhaustive test for this circuit is {(A, B, C) | (0, 0, 0), (1, 1, 1), (1, 0, 1), (1, 1, 0), (0, 1, 0), (0, 1, 1), (1, 0, 0)}. In this case, the (A, B, C) = (1, 0, 0) combination is added to the 100% SSF test set to construct the gate exhaustive test set.

From the data in Table 1, the GEC for this gate exhaustive test set is calculated to be 100% (= (4 + 6 + 3) / (4 + (8 – 2) + (4 – 1)) × 100).

Another example to illustrate the concept of gate exhaustive testing and the GEC is shown in Fig. 3. In the circuit shown in Fig. 3, the (G1, G2) = (0, 0), (0, 1), and (1, 0) combinations cannot be sensitized to any observation point. And the (J1, J2) = (0, 1) combination cannot be applied to the inputs of gate J. A 100% SSF test set for the circuit in Fig. 3 is {(A, B, C) | (1, 1, 0), (0, 1, 1), (1, 0, 0), (1, 1, 1)}. From Table 2, the GEC of the 100% SSF test set is calculated to be 83.3% (= (1 + 3 + 2 + 4) / ((4 – 3) + 4 + (4 – 1) + 4)). A gate exhaustive test set for the circuit is {(A, B, C) | (0, 0, 1), (0, 1, 1), (1, 0, 1), (1, 1, 1), (0, 0, 0), (1, 1, 0)}. From Table 2, the GEC of the gate exhaustive test set is calculated to be 100% (= (1 + 4 + 3 + 4) / ((4 – 3) + 4 + (4 – 1) + 4)).

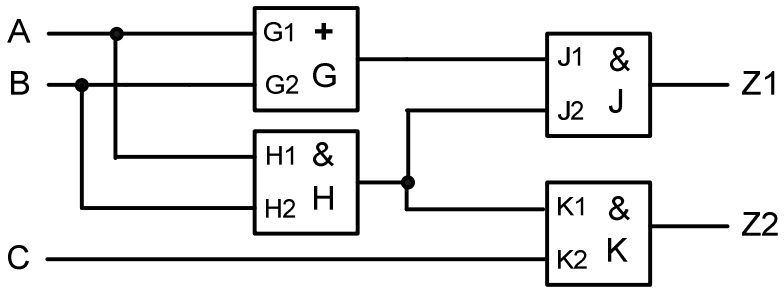


Figure 3 Example circuit to illustrate gate exhaustive testing

Table 2 Table to calculate the GEC for the circuit in Fig. 3

Gate	Number of all gate input combinations	Number of nonobservable gate input combinations	Number of observed gate input combinations	
			100% SSF	Gate exhaustive
G	4	3	1	1
H	4	0	3	4
J	4	1	2	3
K	4	0	4	4

4. Experimental results

Gate exhaustive test sets for the ELF35 [McCluskey 04] and the ELF18 [Mitra 04] cores were generated using the Encounter Test Design Edition tool [Cadence 03]

from Cadence. In order to generate a gate exhaustive test, we used the pattern fault feature available in the tool. A *pattern fault* is defined as a stuck-at fault with logic value constraints on a set of nets [Cadence 03].

An example of a pattern fault is given in Fig. 4. The pattern fault is the Z stuck-at 1 fault with the logic value constraints of (A) = (0) and (B) = (0). So, if the pattern fault is detected, the (A, B) = (0, 0) combination is observed at some observation points. The exhaustive gate input combinations and the corresponding gate output stuck-at faults for all gate types used in the CUTs are specified using pattern faults. If a gate input combination is nonobservable, it is classified as a redundant fault by the ATPG tool.

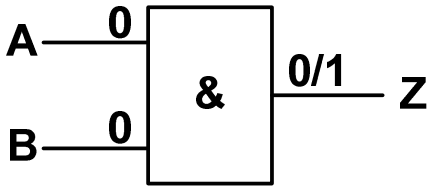


Figure 4 Example circuit to explain a pattern fault

The experiment was conducted on the ELF35 and the ELF18 cores and SSF test sets, *N*-detect test sets (*N* = 2 to 15), transition test sets, and gate exhaustive test sets were tested. Table 3 explains the test sets used in the experiments.

Table 3 Explanation of test sets used in the experiment

Test set	Description
100% SSF	A test that detects all possible SSFs at least one time
2-detect	A test that detects all possible SSFs at least two times
3-detect	A test that detects all possible SSFs at least three times
5-detect	A test that detects all possible SSFs at least five times
10-detect	A test that detects all possible SSFs at least ten times
15-detect	A test that detects all possible SSFs at least fifteen times
Transition	A test that detects all possible transition faults
Gate exhaustive	A test that observes all possible input combinations to each gate
XOR & MUX exhaustive	A test that detects all possible SSFs at least one time and observes all possible input combinations to XOR gates and multiplexers

A transition test pattern consists of two patterns denoted by (V_1, V_2). To detect a slow-to-rise (slow-to-fall) fault on a node, V_1 sets the target node to 0 (1), and V_2 detects the stuck-at 0 (1) faults on the target node.

In a scan-based circuit, the transition tests are applied using skewed-load [Savir 93], also called launch-on-shift, or broadside [Savir 94], also called launch-on-capture method. In this work, the transition test was generated by broadside method. In broadside method, the effect of V_1 is captured on scan cells when launch clock is applied and those captured logic values are used to launch transitions. If the fault effect of V_1 is captured by some scan flip-flops and propagated to some observation points when the next capture clock is applied, then the V_1 can contribute to the detection of defective cores. In order to consider the effect of V_1, the simulation results for V_1 and V_2 are merged to find the GEC for the union of V_1 and V_2 ($V_1 \cup V_2$). So, if one input combination to a gate is observed by either V_1 or V_2, the gate input combination is considered to be observed in the calculation of the GEC for $V_1 \cup V_2$. The GECs for transition tests are presented in Table 4.

Table 4 GEC for transition tests

Core	V_1	V_2	$V_1 \cup V_2$
ELF35	71.3%	83.5%	84.8%
ELF18	85.2%	94.8%	96.5%

The ELF35 chips were fabricated by LSI Logic using their G10P standard cell technology (L_{eff} = 0.35 micron) and V_{dd} is 3.3V. Over ten thousand chips were tested and we collected 324 cores that failed at least one of the 278 test sets applied at 2 voltages (3.3V and 1.4V) and 3 test speeds (fast, rated, and slow). The fast clock cycle time is the minimum clock cycle time at which all the good cores can operate and this clock cycle time was determined using shmoo plot for all good cores. The rated clock cycle time and slow clock cycle time are 1.08 times fast clock cycle time and 3 times fast clock cycle time, respectively. The ELF35 chips consist of 4 combinational cores and 2 sequential cores. The defects that are present on the chips are only those that occurred naturally during fabrication. No artificial defects were inserted [McCluskey

04]. In this experiment, 324 defective cores were tested at the rated clock speed and the operating voltage was 3.3V. In the experiments, the gate exhaustive test set detected all defective cores.

The ELF18 chips were manufactured in the Philips 0.18 micron Corelib technology, and V_{dd} is 1.8V. The R.E.A.L. Digital Signal processor is implemented in the DSP cores, and each chip contains 6 cores [Mitra 04]. More than 70,000 test chips were fabricated and 300 defective cores are used in this experiment. The rated clock speed of ELF18 RDMR core is 20MHz, but in this test the cores were tested at 5MHz. The operating voltage was 1.8V. If a core failed any of the test sets, the core was classified as a defective core.

Table 5 shows the number of fabricated chips and the interesting cores (defective cores) used in the experiment. The interesting cores failed at least one test set that was applied to the cores.

Table 5 The number of test chips and defective cores for ELF35 and ELF18

Core	Total number of fabricated chips	Number of cores used in the experiment
ELF35	> 10,000	324
ELF18	> 70,000	300

Table 6 shows the number of gates, the number of all gate input combinations, and the number of all nonobservable gate input combinations for the ELF35 [McCluskey 04] and the ELF18 [Mitra 04] chips. The numbers are the summations for all cores in a chip.

Table 6 Circuit information to calculate the GEC for the ELF35 and the ELF18 cores

Circuit	Number of gates	Number of all gate input combinations	Number of nonobservable gate input combinations
ELF35	54,242	859,766	367,237
ELF18	322,392	969,852	228,072

The experimental results for the ELF35 cores and the ELF18 cores are shown in Table 7 and Table 8, respectively.

Table 7 Results for the ELF35 cores

Test set	Test Length	SSF test coverage (%)	Transition fault coverage (%)	GEC (%)	Test Escapes	Test generation time [sec]	Test speed
100% SSF	3,272	99.6	-	85.7	3	244	Rated clock (For transition test, rated clock speed was used for launch and capture)
2-detect	6,189	99.6	-	88.6	3	392	
3-detect	9,123	99.6	-	90.3	3	525	
5-detect	14,972	99.6	-	92.9	1	828	
10-detect	29,521	99.6	-	94.1	1	1,526	
15-detect	44,227	99.6	-	95.9	1	2,183	
Transition	4,992	-	98.5	84.8 $^{\alpha}$	3	506	
Gate exhaustive	5,655	99.6	-	98.3	0	6,843	

All tests are generated using complex gate level netlist.
$^{\alpha}$GEC is calculated for $V_1 \cup V_2$

Table 8 Results for the ELF18 cores

Test set	Test Length	SSF test coverage (%)	Transition fault coverage (%)	GEC (%)	Test Escapes	Test generation time [sec]	Test speed
100% SSF	427	100.0	-	93.4	4	198	5 MHz clock (For transition test, 5 MHz clock speed was used for launch and capture)
100% SSF$^{\alpha}$	453	100.0	-	94.2	3	462	
2-detect	795	100.0	-	95.2	3	320	
3-detect	1,152	100.0	-	96.3	3	430	
5-detect	1,841	100.0	-	97.2	2$^{\lambda}$	679	
10-detect	3,554	100.0	-	98.0	1$^{\lambda}$	1,232	
15-detect	5,290	100.0	-	98.2	2$^{\lambda}$	1,811	
Transition	792	-	99.7	96.5$^{\beta}$	0	742	
XOR & MUX exhaustive	856	100.0	-	94.8	3	2,799	
Gate exhaustive	1,528	100.0	-	99.0	2$^{\lambda}$	3,766	

$^{\alpha}$Generated using elementary gate level netlist. Other tests are generated using complex gate level netlist.
$^{\beta}$GEC is calculated for $V_1 \cup V_2$
$^{\lambda}$Classified as chips with sequence dependent defects

The test lengths for each test set are shown in the second column of Table 7 and Table 8. The test length of the gate exhaustive test set in the ELF35 experiment is shorter than that of the 2-detect test set, but the gate exhaustive test set detected all defective cores. In the ELF18 experiment, the gate exhaustive test set showed the same defect coverage as the 5-detect test set. However, the test length of the gate exhaustive test set was shorter than that of the 5-detect test set. The 15-detect test set could not detect more defective cores than the 5-detect test set even if we applied almost 3 times more test patterns.

In the ELF18 experiment, the two defective cores that escaped the 15-detect test set and the gate exhaustive test set were classified to have sequence dependent defects.

In the ELF18 experiment, two 100% SSF test sets were generated; one was generated using a complex gate level netlist and the other was generated using an elementary gate level netlist. The SSF test set generated using an elementary gate level netlist detected one more defective core than the SSF test set generated using a complex gate level netlist, and this test quality difference can be explained by the GECs of the test sets.

A test set, which is SSF test set with the exception that it observes exhaustive input combinations to XOR gates and multiplexers, was tested and the result is shown in XOR & MUX exhaustive row in Table 8. This test set has 3 test escapes.

The SSF test coverage for each test set is given in Table 7 and Table 8. The SSF test coverage is calculated as the ratio of the number of all detected SSFs to the number of all detectable SSFs.

Some gate input combinations are not observed and are not identified as nonobservable gate input combinations because the detection of the pattern faults are aborted. So, the GECs of gate exhaustive test sets are lower than 100%.

From Table 7 and Table 8, it is shown that the ability of a test set to detect defective cores increases as the GEC of the test set increases except the 10-detect test set for the ELF18 experiment.

The test generation times are given under the test generation time column of Table 7 and Table 8. The test generation times of the gate exhaustive test sets are almost 3 times and 2 times longer than the 15-detect test sets for the ELF35 and the ELF18 cores, respectively. The test sets were generated on Sun-Blade-1000 with Solaris 2.8 operating system.

5. *N*-Detect stuck-at fault simulation of gate exhaustive tests

N-detect stuck-at fault simulation is done on gate exhaustive tests, 100% SSF tests, and 15-detect tests. Figure 5 and Fig. 6 show the *N*-detect stuck-at fault simulation results for the ELF35 and the ELF18 test sets, respectively. Along the x-axis we plot the number of times a stuck-at fault is detected. And along the y-axis, we plot the number of stuck-at faults detected by the given number of times. For example, for the 15-detect test sets, most stuck-at faults are detected more than 15 times. For the gate exhaustive test sets, many stuck-at faults are detected less than 15 times. From these results, we can find that the quality of gate exhaustive test sets does not come from the multiple detections of stuck-at faults. In other words, the quality of gate exhaustive test sets is not obtained from the fortuitous detection by adding more patterns that detect stuck-at faults multiple times.

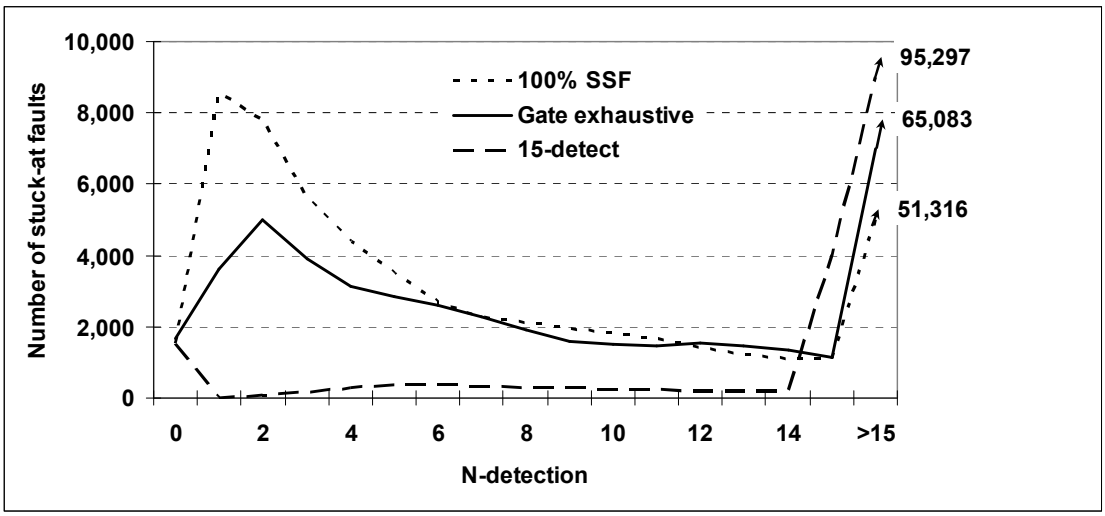


Figure 5 N-detect stuck-at fault simulation results: ELF35

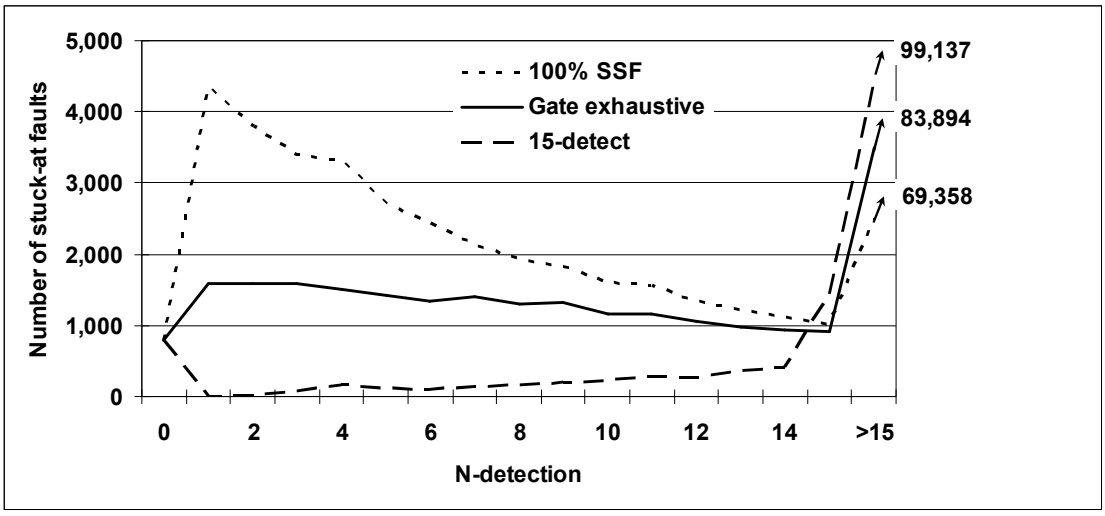


Figure 6 N-detect stuck-at fault simulation results: ELF18

6. Conclusions

Experimental results on actual chips demonstrate the effectiveness of the gate exhaustive test sets in detecting defective chips. Test sets with higher gate exhaustive coverage detect more defective chips. This result makes the gate exhaustive test metric suitable for grading test thoroughness and also for test pattern generation.

Commercial ATPG tools can be used to generate gate exhaustive test sets. Moreover, as outlined in this paper, the gate exhaustive test metric avoids several open questions associated with the *N*-detect test metric.

Acknowledgments

This work was supported by NSF under contract number CCR-0098218. And test chip experiments are supported by LSI Logic and Philips Semiconductors.

The authors would like to thank Bill Price, Stefan Eichenberger, and Fred Bowen of Philips Semiconductors, and Erik Volkerink, Ahmad Al-Yamani, and Intaik Park of Stanford CRC for their support for the ELF18 experiment. And the authors also thank Samy Makar of Azul Systems for his advice on ATPG tools.

References

[Benware 03] Benware, B., C. Schuermyer, N. Tamarapalli, and K. H. Tsai, "Impact of Multiple-Detect Test Patterns on Product Quality," *Proc. Intl Test Conf*., pp. 1031-1040, 2003.

[Blanton 97] Blanton, R. D. and J. P. Hayes, "Properties of the Input Pattern Fault Model," *ICCD*, pp. 372-380, 1997.

[Blanton 03] Blanton, R. D., K. N. Dwarakanath and A. B. Shah, "Analyzing the Effectiveness of Multiple-Detect Test Sets," *Proc. Intl. Test Conf*., pp. 876-885, 2003.

[Cadence 03] Cadence, "Encounter Test Design Edition User Guide," Oct. 2003.

[Dworak 04] Dworak, J., B. Cobb, J. Wingfield, and M. R. Mercer, "Balanced Excitation and its Effect on the Fortuitous Detection of Dynamic Defects," *Design, Automation, and Test in Europe*, pp.1066-1071, vol. 2, Feb. 2004.

[Li 02] Li, James C.-M. and E. J. McCluskey, "Diagnosis of Sequence-dependent Chips," *Proc. VLSI Test Symp*., pp. 187-192, 2002.

[Ma 95] Ma, S. C., P. Franco, and E. J. McCluskey, "An Experimental Test Chip to Evaluate Test Techniques Experimental Results," *Proc. Intl. Test Conf*., pp. 663-672, 1995.

[McCluskey 84] McCluskey, E. J., "Verification Testing – A Pseudoexhaustive Test Technique," *IEEE Trans. Computers*, vol. C-33, no. 6, pp. 541-546, Jun. 1984

[McCluskey 93] McCluskey, E. J., "Quality and Single-Stuck Faults," *Proc. Intl. Test Conf*., p. 597, 1993.

[McCluskey 00] McCluskey, E. J. and Chao-Wen Tseng, "Stuck-Fault Tests vs. Actual Defects," *Proc. Intl. Test Conf*., pp. 336-343, 2000.

[McCluskey 04] McCluskey, E. J., et al., "ELF-Murphy Data on Defects and Test Sets," *Proc. VLSI Test Symp*., pp. 16-22, 2004.

[Mitra 04] Mitra, S., E. H. Volkerink, E. J. McCluskey, and S. Eichenberger, "Delay Defect Screening using Process Monitor Structures," Proc. VLSI Test Symp., pp. 43-52, 2004.

[Pomeranz 03] Pomeranz, I. and S. M. Reddy, "On Test Data Compression and n-Detection Test Sets," *Design Autom. Conf*., pp. 748-751, 2003.

[Savir 93] Savir, J. and S. Patil, "Scan-Based Transition Test," *IEEE Trans. on CAD*, vol. 12, no. 8, pp. 1232-1241, Aug. 1993.

[Savir 94] Savir, J. and S. Patil, "On Broad-Side Delay Test," *Proc. VLSI Test Symp*., pp. 284-290, 1994.

[Shperling 87] Shperling, I. and E. J. McCluskey, "Circuit Segmentation for Pseudo-Exhaustive Testing via Simulated Annealing," *Proc. Intl Test Conf*., pp. 58-66, 1987.

[Tseng 01a] Tseng, C.-W. and E. J. McCluskey, "Multiple-Output Propagation Transition Fault Test," *Proc. Intl Test Conf*., pp. 358-366, 2001

[Tseng 01b] Tseng, C.-W., S. Mitra, S. Davidson, and E. J. McCluskey, "An Evaluation of Pseudo Random Testing for Detecting Real Defects," *Proc*. *VLSI Test Symp.*, pp. 404-409, 2001.

[Venkataraman 04] Venkataraman, S., S. Sivaraj, E. Amyeen, S. Lee, A. Ojha, and R. Guo, "An Experimental Study of N-Detect Scan ATPG Patterns on a Processor," *Proc. VLSI Test Symp.*, pp. 23-29, 2004.

[Waicukauski 87] Waicukauski, J. A., E. Lindbloom, B. K. Rosen, and V. S. Iyengar, "Transition Fault Simulation," *IEEE Design & Test Comput*., vol. 4, no. 2, pp. 32-38, Jan. 1987.

Appendix B

Test Set Reordering Using the Gate Exhaustive Test Metric

Proceedings of VLSI Test Symposium (VTS), pp. 199-204, 2007.

Test Set Reordering Using the Gate Exhaustive Test Metric

Kyoung Youn Cho and Edward J. McCluskey
Center for Reliable Computing
Department of Electrical Engineering
Stanford University, Stanford CA

Abstract

When a test set size is larger than desired, some patterns must be dropped. This paper presents a systematic method to reduce test set size; the method reorders a test set using the gate exhaustive test metric and truncates the test set to the desired size.

To determine the effectiveness of the method, test sets with 1,556 test patterns were applied to 140 defective Stanford ELF18 test cores. The original test set required 758 test patterns to detect all defective cores, while the test set reordered using the presented method required 286 test patterns. The method also reduces the test application time for defective cores.

1. Introduction

Production testing cost is closely related to the test application time and the test set size. In order to reduce these values, test set reordering has been studied. The advantage of test set reordering is based on the following: 1) when a test set is too large to fit in a tester memory the patterns in the later part of the test set can be removed with only minimal impact on defect detection [Pomeranz 04]; 2) the defective chips fail early on a tester, reducing the test application time for the defective chips [Maly 86][Jiang 01][Pomeranz 04].

Experimental results using industrial chips demonstrated that there were big differences in the number of defect detection among single stuck-at fault (SSF) test patterns, and test patterns that detected large number of defective chips were

distributed throughout the test set [Nigh 00]. Therefore test set reordering is important to reduce test set size and to reduce test application time.

An optimal test set reordering method was presented assuming that the defect occurrence probability and the relationship between defects and faults were available [Maly 86]. However, this is not a feasible solution in modern semiconductor designs. Jiang and Vinnakota used the failing data from sample chips; they applied all groups of test patterns to a sample set of ICs, and used the failing data to reorder test sets [Jiang 01]. This method is not cost efficient because it needs a tester. The method also requires a representative sample set of chips because it assumes that faulty behavior doesn't change from lot to lot, which may or may not be true. In another approach, fault simulation for each test pattern is performed, and the number of single stuck-at faults detected by each test pattern is used to reorder test sets [Lin 01]. Other researchers presented their own criteria (or metrics) to select test patterns from an *N*-detect test set [Chao 04][Tian 05]. Another approach to select effective test patterns using probabilistic fault model and output deviations was presented [Wang 06a]. Test type reordering was also studied; functional test, IDDQ test, delay test, and stuck-at test sets were reordered, and the total test application times were compared [Butler 00].

A *test metric* is used to measure the thoroughness of test sets; when a test set is evaluated using a test metric, the thoroughness of the test set is usually calculated as a coverage value [McCluskey 04a, 04b]. It is also used to guide automatic test pattern generation (ATPG) tools to generate test patterns [McCluskey 04a, 04b]. A test metric can also be used to reorder generated test sets, which is addressed in this paper.

This chapter presents a test set reordering method using the gate exhaustive test metric (GEM). Experimental results on actual test chips demonstrate that the reordering method using the GEM is more effective than the previous methods using the SSF test metric. The *GEM* evaluates the thoroughness of a test set by calculating the ratio of the number of gate input combinations observed by the test set to the number of observable gate input combinations; the ratio is defined as the *gate exhaustive coverage (GEC)* of the test set [Cho 05].

An *observed gate input combination* of an internal gate is an input combination applied to the gate inputs with the gate output being sensitized to at least one observation point such as a primary output or a scan flip-flop [Cho 05]. An *observable gate input combination* of an internal gate is an input combination that can be applied to the gate inputs with the gate output being sensitized to at least one observation point [Cho 05]. A *nonobservable gate input combination* of an internal gate is an input combination that cannot be applied to the gate inputs (known as "controllability don't care" [De Micheli 94]) or that cannot be sensitized to any observation point (known as "observability don't care" [De Micheli 94]) [Cho 05]. *Gate exhaustive simulation* identifies the gate input combinations observed by a given test set using a simulation of the circuit operation. A *gate exhaustive test set* is a test set that observes all observable gate input combinations at some observation points in a combinational circuit or a full scan sequential circuit [McCluskey 93][Cho 05].

The *SSF test metric* evaluates the thoroughness of a test set by calculating the percentage of target SSFs detected by the test set; the percentage is defined as the *SSF coverage* of the test set. The *SSF model* is a fault model in which only one node is stuck at logic-0 or logic-1 in the circuit under test [Eldred 59][Abramovici 90]. *SSF simulation* identifies the SSFs detected by a given test set using a simulation of the circuit operation.

An *original test set* is a test set that has the sequence of test patterns generated by an ATPG tool. When the sequence of patterns of an original test set is changed, the test set is called a *reordered test set*.

A *surrogate fault model* has been used to represent defects when chip experiments are not available; a fault model that is not targeted during test set generation or test set reordering is used as a surrogate fault model. When the SSF model is used for test generation or test set reordering, the bridging fault model has been used as a surrogate fault model [Kapur 92][Tian 05]. This paper also discusses the correlation between "surrogate" fault coverage and the defect detection. The bridging fault model is evaluated as a surrogate fault model.

This chapter is organized as follows: Section 2 presents the test set reordering method using the GEM; Section 3 reports the experimental results that support the effectiveness of the test set reordering method presented in this paper; Section 4 evaluates the bridging fault model as a surrogate fault model. Section 5 discusses the effectiveness of the GEM in detecting defective chips; Section 6 concludes the paper.

2. Test Set Reordering

Cho, Mitra, and McCluskey reported that a test set with a higher GEC detected more defective chips than a test set with a lower GEC [Cho 05]. The GEC was also shown to be correlated with silicon fallout better than the SSF coverage or the bridge coverage estimate [Guo 06]. In this paper, we present a test set reordering method using the GEM. The test set reordering flow and an example are presented in this section.

Gate exhaustive simulation is conducted on all test patterns without dropping the gate input combinations observed by other test patterns. After the simulation, one test pattern that yields the highest GEC is removed from the original test set and appended to the reordered test set. Then, all gate input combinations observed by the selected test pattern are removed from the list of gate input combinations observed by each test pattern in the original test set. The test pattern selection process finishes when all test patterns are reordered. The reordered test set maximizes the cumulative GEC for each additional test pattern. The *cumulative GEC of test pattern T* is the GEC calculated for the subset comprising all patterns up to and including test pattern T.

An example of test set reordering using the GEM is presented using the ISCAS85 c17 benchmark circuit illustrated in Fig. 1. A gate exhaustive test set for the benchmark circuit was generated using the procedure presented in [Cho 05]; the test set consists of 9 test patterns.

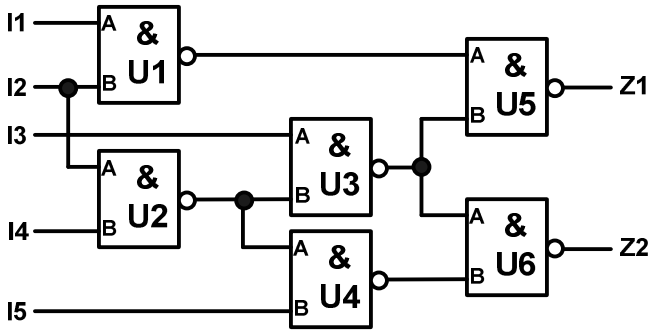


Figure 1 ISCAS85 c17 benchmark circuit

Gate exhaustive simulation was conducted on the generated test patterns, and the gate input combinations observed by each test pattern are presented in Table 1. In Table 1, U1/01 denotes the gate input combination AB = 01 to gate U1.

Table 1 Observed gate input combinations

Test pattern	Observed gate input combinations
t_1 (01000)	U1/01, U3/01, U4/10, U5/11, U6/11
t_2 (00110)	U2/01, U3/11 U5/10, U6/01
t_3 (11001)	U1/11, U2/10, U4/11, U5/01, U6/10
t_4 (00000)	U1/00, U3/01, U4/10, U5/11, U6/11
t_5 (11101)	U2/10, U5/00, U6/00
t_6 (01010)	U1/01, U3/00, U4/00, U5/11, U6/11
t_7 (00100)	U2/00, U3/11, U5/10, U6/01
t_8 (01111)	U1/01, U2/11, U3/10, U4/01, U5/11, U6/11
t_9 (10000)	U1/10, U3/01, U5/11, U6/11

In Table 1, test pattern t_8 observes the largest number of gate input combinations in the test set, so test pattern t_8 is selected first. Then, the gate input combinations observed by test pattern t_8 are removed from the list of the gate input combinations observed by other test patterns. This process is repeated for all remaining test patterns. One reordered test sequence is (t_8, t_3, t_7, t_4, t_6, t_5, t_9, t_2, t_1). During the process we can find that test pattern t_1 can be removed without reducing the GEC of the test set.[1]

[1] If a better compaction is used during the test generation, t_1 may not be included in the test set.

Table 2 shows the SSFs detected by each test pattern, in which U1/Z/1 means the stuck-at 1 fault on the pin Z of gate U1. One sequence of test patterns reordered using the SSF test metric is (t_2, t_8, t_3, t_9, t_1, t_4, t_5, t_6, t_7). During the process we can find that the test patterns t_1, t_4, t_5, t_6, and t_7 can be removed without impacting on the SSF coverage of the test set.

Table 2 Detected single stuck-at faults

Test pattern	Detected single stuck-at faults
t_1 (01000)	Z1/1, I1/1, U3/Z/0, I3/1 Z2/1, I5/1
t_2 (00110)	U5/B/1, Z1/0, I2/1, U3/Z/1, U2/A/1, U2/Z/0, U6/A/1, Z2/0
t_3 (11001)	U1/Z/1, Z1/0, I2/0, U2/Z/0, I4/1, U4/Z/1, Z0/0
t_4 (00000)	Z1/1, U3/Z/0, I3/1, Z2/1, I5/1
t_5 (11101)	Z1/0, U2/Z/0, I4/1, Z2/0
t_6 (01010)	Z1/1, I1/1, U3/Z/0, Z2/1
t_7 (00100)	U5/B/1, Z1/0, U3/Z/1, U2/Z/0, U6/A/1, Z2/0
t_8 (01111)	Z1/1, I1/1, I2/0, U3/B/1, U3/Z/0, U2/Z/1, U4/A/1, Z2/1
t_9 (10000)	U1/B/1, Z1/1, I2/1, U3/Z/0, I3/1, Z2/1, I5/1

The example demonstrates that the sequence of test patterns reordered using the GEM is different from the sequence reordered using the SSF test metric.

The basic procedure of test set reordering is similar to that of test set compaction [Hamzaoglu 00], but they have different goals. The goal of test set compaction is to reduce the number of test patterns without any impact on the target fault coverage; the goal of test set reordering is to maximize the target fault coverage for each subset of test patterns. In the test set compaction using the GEM, only t_1 can be removed because this pattern doesn't observe any unique gate input combination that other test patterns don't observe; the order of test patterns is not important in test set compaction.

3. Experimental Results

In this paper, a gate exhaustive test set is used as an original test set, but any test set can be used as an original test set. This test set was selected because it was the most effective test set we had tested with respect to test escapes and test set size. The

gate exhaustive test set was generated using the static pattern fault model provided by Cadence Encounter Test Design Edition [Cadence 03] with maximal compaction option. The detailed method of gate exhaustive test set generation was presented in [Cho 05].

In this experiment, the Stanford ELF18 test chips [Mitra 04] were used. The ELF18 test chips were fabricated using the Philips 0.18 micron Corelib technology and V_{dd} is 1.8V. More than 70,000 test chips were manufactured; each test chip contains 6 R.E.A.L. digital signal processor cores. One ELF18 core contains 13 scan chains; the total number of scan flip-flops is 2,290; the total number of gates for one core is 53,732.

Table 3 reports the test length, the SSF coverage, the GEC, and the test generation time of the gate exhaustive test set for the ELF18 core. The original gate exhaustive test set is denoted as **ge_org**.

Table 3 Original test set information: ELF18

Test set	Test length	SSF coverage	GEC	Test generation time [sec]
ge_org	1,556	99.9%	99.1%	2,311

The original test set was reordered using the method presented in Sec. 3 so that the cumulative GEC for each additional test pattern is maximized (**ge_gem**). To compare the effectiveness of the presented reordering method, the original test set was also reordered so that the cumulative SSF coverage for each additional test pattern is maximized (**ge_ssf**). Table 4 describes the test sets compared in this paper.

Table4 Description of test sets

Test set	Description
ge_org	A gate exhaustive test set generated by the Cadence Encounter Test Design Edition ATPG tool
ge_ssf	A test set reordered from **ge_org** to maximize the cumulative SSF coverage for each additional test pattern
ge_gem	A test set reordered from **ge_org** to maximize the cumulative GEC for each additional test pattern

Test set reordering was conducted on a Sun-Fire-V240 workstation running the Solaris 9 operating system. The main memory size was 4 GBytes. Table 5 reports the execution time for each test set reordering. The execution time includes SSF and gate exhaustive simulation time for **ge_ssf** and **ge_gem**, respectively.

Table 5 Execution time to reorder test sets

Test set	Execution time [hh:mm]
ge_org	0:00
ge_ssf	7:19
ge_gem	15:23

Figure 2 shows the difference in the cumulative SSF coverage between **ge_ssf** and **ge_gem**. The figure illustrates that **ge_ssf** has higher cumulative SSF coverage than **ge_gem** for almost all test patterns.

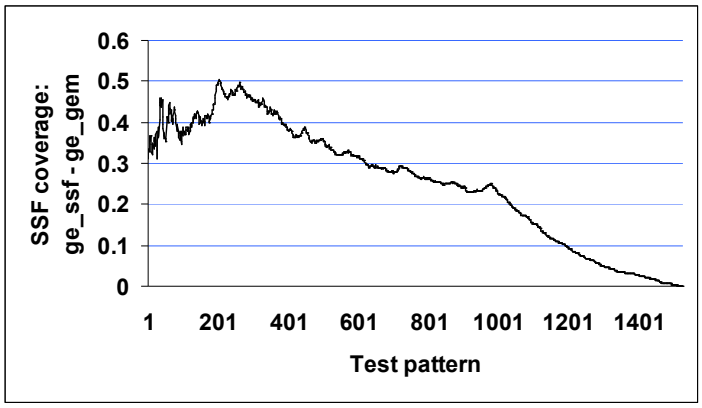


Figure 2 Cumulative SSF coverage comparison: ge_ssf – ge_gem

Figure 3 illustrates the difference in the cumulative GEC between **ge_gem** and **ge_ssf; ge_gem** has higher cumulative GEC than **ge_ssf** for almost all test patterns.

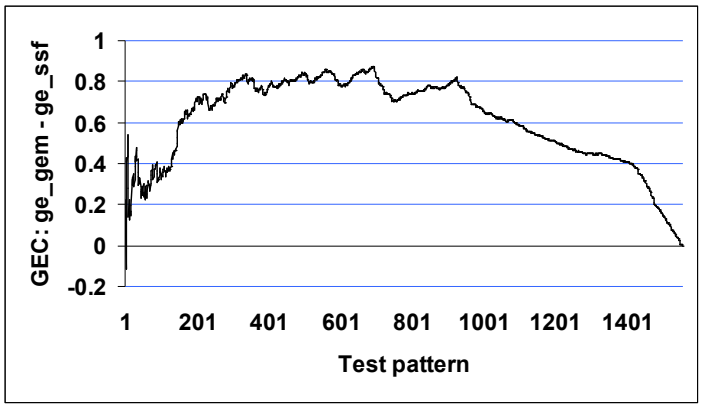


Figure 3 Cumulative GEC comparison: ge_gem – ge_ssf

The test sets (**ge_org**, **ge_ssf**, and **ge_gem**) were applied to the ELF18 test cores. In the experiment, each core was tested independently and 140 defective cores that failed the original test set were tested. The test application stops at the first failure, and the first failing cycle was used to calculate the first failing pattern number for each defective core. The pattern number of the first test pattern is 1.

Figure 4 illustrates the difference in the first failing pattern numbers between **ge_org** and **ge_gem** for each core. The figure demonstrates that **ge_gem** detects many tested cores much earlier than **ge_org**. For some tested cores, **ge_org** detects the cores earlier than **ge_gem**, but the pattern number difference is less than 100.

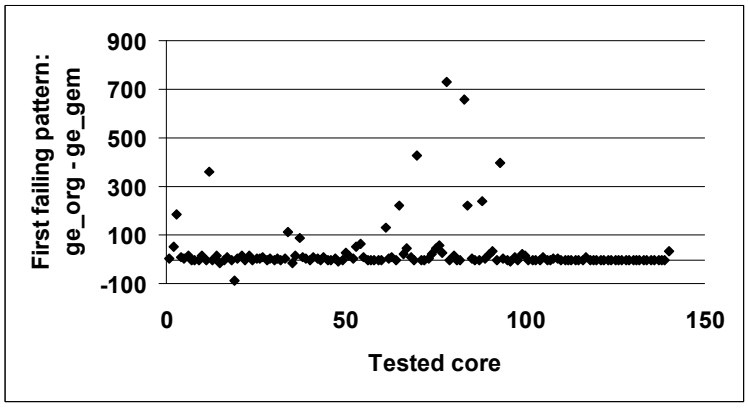


Figure 4 Difference in the first failing pattern numbers: ge_org – ge_gem

Figure 5 describes the difference in the first failing pattern numbers between **ge_ssf** and **ge_gem** for each core. Some tested cores are detected much earlier by **ge_gem** than **ge_ssf**.

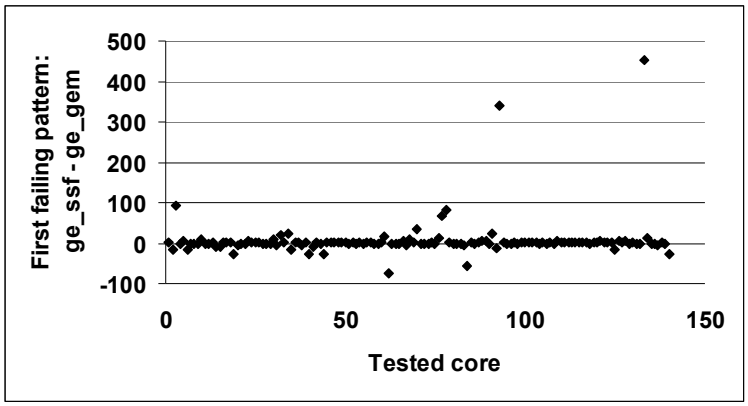


Figure 5 Difference in the first failing pattern numbers: ge_ssf – ge_gem

Table 6 reports the maximum numbers and the average numbers of the first failing patterns for each test set. The maximum number shows how many test patterns can be removed without any impact on defect detection; the average number represents the test application time for the defective cores. The table demonstrates that more patterns can be removed from **ge_gem** than **ge_ssf** with minimal impact on defect detection. This is important when the test set size must be reduced because it doesn't satisfy the required limits; when tester memory size or test application time is limited, test set size must be reduced by dropping some test patterns.

Table 6 The statistics of first failing test pattern number

Test set	**ge_org**	**ge_ssf**	**ge_gem**
Maximum	758	739	286
Average	44.09	18.86	12.69

Figure 6 describes the number of test escapes when only the test patterns up to the number shown in the horizontal axis are applied. In other words, the figure shows the number of defective cores that escape the test set when the latter patterns of the test sets are eliminated. The figure demonstrates that **ge_gem** detects more defective cores than **ge_org** or **ge_ssf** when the test sets are truncated to the same size.

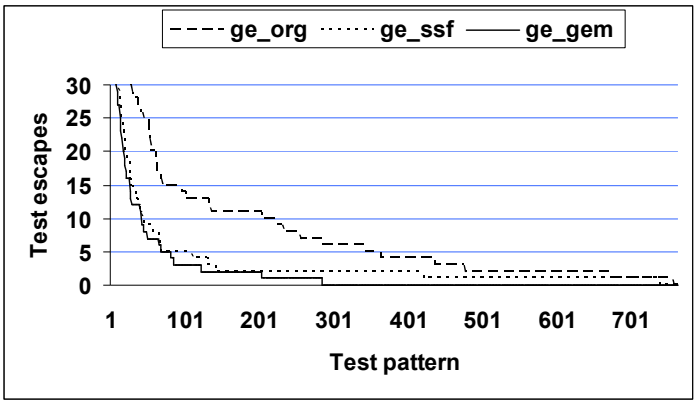


Figure 6 Test escapes for the test sets when the test sets are truncated

In addition to that, the test set reordering method can reduce the test application time. However, the amount of reduction depends on the yield of the fabrication process. Equation 1 calculates the average number of test patterns (*AN*) applied to each chip when test application stops at the first failure. *TL* is the total test length, *Y* is the yield of the fabrication process, and *AF* is the average number of the first failing patterns for defective chips. When the yield is 0.5, the average numbers of test patterns applied to each chip for the **ge_org**, **ge_ssf**, and **ge_gem** are 800, 787, and 784, respectively.

$$AN = TL \times Y + AF \times (1 - Y) \tag{1}$$

In combinational circuits, test set reordering may affect defect detection of test sets because of sequence dependent defects [McCluskey 04a]. The ELF18 core is a full scan circuit, which means that the internal state of the circuit when a test pattern is applied is more determined by the scan shift-in operation than the previous test pattern [Ma 99]. In the application of the three test sets, sequence dependent behavior was not considered.

4. Surrogate Fault Model

In order to compare the effectiveness of the test sets in detecting unmodeled faults or defects, surrogate fault models have been used. In this paper, the bridging fault model is evaluated as a surrogate fault model. The target bridging faults were

extracted from the layout data of the ELF18 design and used when the bridging fault simulation was performed. The bridging fault model used in the simulation was the victim-aggressor model [Acken 83] illustrated in Fig. 7; the logic value on the aggressor node changes the logic value on the victim node. The detection of a bridging fault depends on the driving force of gates. In this paper, however, it is assumed that the bridging fault is detected whenever the stuck-at-0 (1) fault on the victim node is detected with logic-0 (1) being assigned to the aggressor node. Bridging fault simulation was conducted using the Synopsys TetraMAX [Synopsys 05].

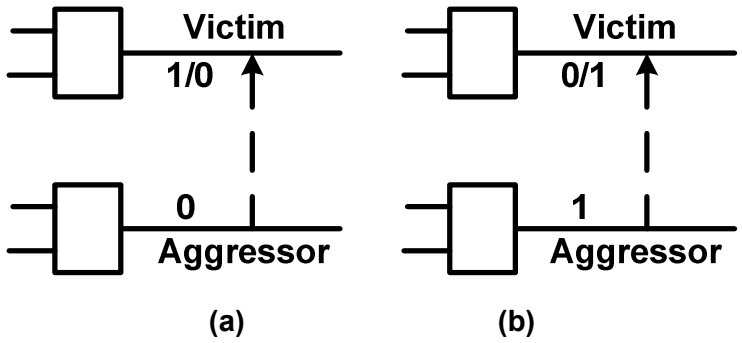


Figure 7 Bridging fault models: (a) Logic-0 on the aggressor node changes logic-1 on the victim node; (b) Logic-1 on the aggressor node changes logic-0 on the victim node

Figure 8 illustrates the difference in the cumulative bridging fault coverage between **ge_ssf** and **ge_gem**. The figure shows that the cumulative bridging fault coverage of **ge_gem** is almost similar to that of **ge_ssf**. Comparing Figs. 6 and 8 demonstrates that there is not a meaningful correlation between the number of detected cores and the cumulative bridging fault coverage in the ELF18 experiments; in other words, the bridging fault model is not a useful surrogate fault model in the ELF18 experiments.

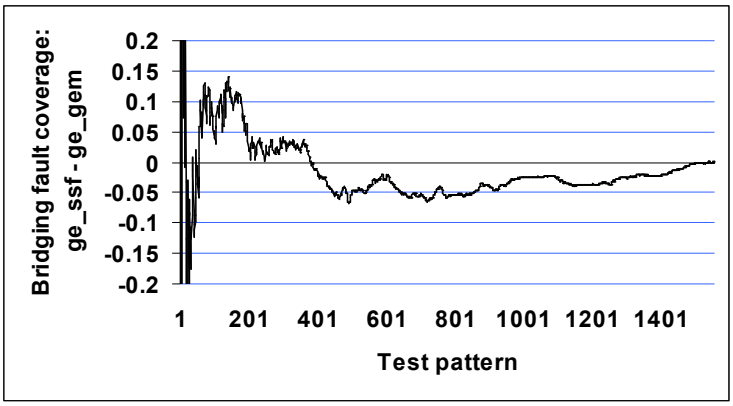


Figure 8 Comparison of cumulative bridging fault coverage: ge_ssf – ge_gem

5. Effectiveness of the the GEM

In this section, the effectiveness of the GEM over the SSF test metric is presented. To detect more SSFs, at most one controlling value is applied to a gate. For example, if the gate input combination (00) is applied to a two-input NAND gate, all the sensitization paths through the gate are blocked; this will reduce the number of detected SSFs. However, this gate input combination may be effective in detecting some types of defects. Let us consider the circuit illustrated in Fig. 9. Applying the (00) combination to NAND gate K assigns a strong logic-1 to the gate output; the strong logic-l may change the logic value on the node connected by bridging defects. The bridging defects may not be detected, if (01) or (10) combination is applied to the inputs of gate K.

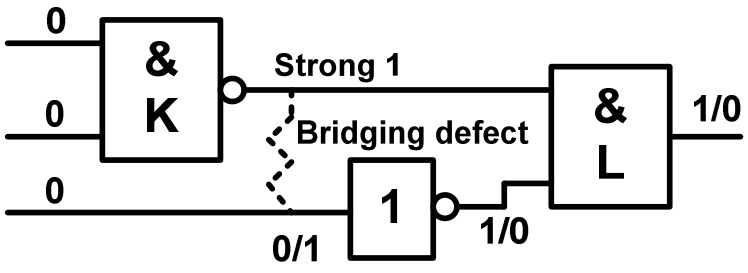


Figure 9 Bridging defect detection

Another example that supports the effectiveness of the GEM is explained using the circuit presented in Fig. 10. In SSF test generation, the stuck-at 1 fault on the

output of gate L will be propagated to Z1 because it is an easy path to propagate the effect of the SSF; however, applying the (001) combination to the inputs of gate L will block the sensitization path to Z1 at gate M, and propagate the fault signal to Z2 through another path, increasing the defect detection probability by detecting the SSF with different conditions from the SSF test generation.

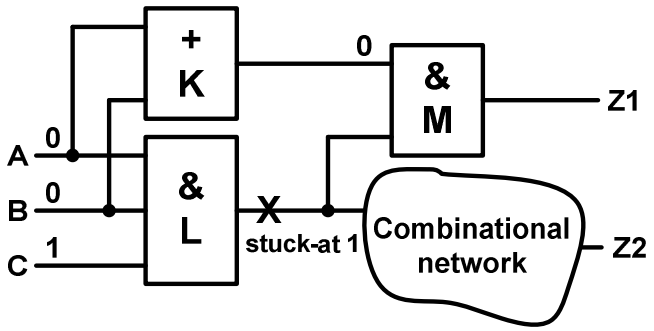


Figure 10 Propagation of fault signal

Applying some gate input combinations to a gate may detect specific types of defects inside the gate that are not guaranteed to be detected by detecting only SSFs. Let us consider the transistor level implementation of a two-input AND gate with the bridging defect between internal node N and the gate output as described in Fig. 11. When the PMOS transistors M1 and M2 turn on simultaneously, logic-1 may be assigned to the gate output in stead of logic-0 because of the strong logic-1 assigned to internal node N, detecting the bridging defect; however, the bridging defect may not be detected by applying (01) or (10) combination because the NMOS transistor M6 may drive the output to logic-0. However, the detection depends on the relative driving strength of transistors. The examples demonstrate that observing more gate input combinations increases the possibility of defect detection.

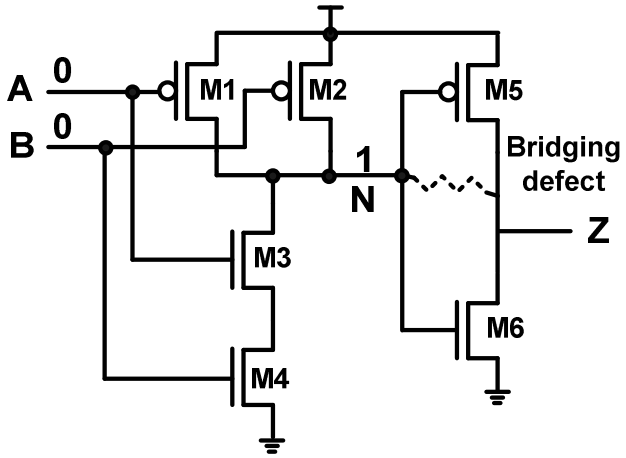


Figure 11 A transistor level implementation of two-input AND gate with a bridging defect

6. Conclusions

This paper presents a test set reordering method; the method reorders a test set using the gate exhaustive test metric and truncates the test set to the desired test set size. The reordering was applied to a test set generated with a maximal compaction and reordering supported by an ATPG tool. Experimental results using the Stanford ELF18 test chips demonstrated that the test set reordered using the gate exhaustive test metric could be truncated with less impact on defect detection than the test set reordered using the SSF test metric. The method also reduces test application time for defective chips more than the method using the SSF test metric.

The test set reordering method presented in this paper is cost efficient because it can be implemented using computer simulation without any experiment on a tester.

The results also demonstrate that the bridging fault model is not a useful surrogate fault model in the ELF18 experiments. We propose the GEM as a test metric that can be used to measure the thoroughness of test sets.

Acknowledgments

The research was supported by NSF under contract number CCR-0098128 and LSI Logic. The ELF18 test chip experiments are supported by Philips Semiconductors.

The authors would like to express their thanks to Bill Price, Stefan Eichenberger, and Fred Bowen of Philips Semiconductors, Erik Volkerink, Intaik Park, and Francois-Fabien Ferhani of Stanford CRC for their support for the ELF18 experiments.

References

[Abramovici 90] Abramovici, M., M. A. Breuer, and A. D. Friedman, *Digital Systems Testing and Testable Design*, IEEE Press, New York, 1990.

[Acken 83] Acken, J. M., "Testing for Bridging Faults (Shorts) in CMOS Circuits," *Proc. Design Automation Conf.*, pp. 717-718, 1983.

[Butler 00] Butler, K. M. and J. Saxena, "An Empirical Study on the Effects of Test Type Ordering on Overall Test Efficiency," *Proc. Intl. Test Conf.*, pp. 408-416, 2000.

[Cadence 03] Encounter Test Design Edition User Guide, Cadence Design Systems, Inc., Oct. 2003.

[Chao 04] Chao, C.-T., L.-C. Wang, and K.-T. Cheng, "Pattern Selection For Testing Of Deep Sub-Micron Timing Defects," *Proc. Design Automation & Test in Europe*, pp. 1060-1065, 2004.

[Cho 05] Cho, K. Y., S. Mitra, and E. J. McCluskey, "Gate Exhaustive Testing," *Proc. Intl. Test Conf.*, Paper 31.3, 2005.

[De Micheli 94] De Micheli, G., *Synthesis and Optimization of Digital Circuits*, McGraw-Hill, 1994.

[Eldred 59] Eldred, R. D., "Test Routines Based on Symbolic Logical Statements," *Journal of the ACM*, vol. 6, no. 1, pp. 33-36, 1959.

[Guo 06] Guo, R., *et al.*, "Evaluation of Test Metrics: Stuck-at, Bridge Coverage Estimate and Gate Exhaustive," *Proc. VLSI Test Symp.*, pp. 66-71, 2006.

[Hamzaoglu 00] Hamzaoglu, I. and J. H. Patel, "Test Set Compaction Algorithms for Combinational Circuits," *IEEE Trans. on CAD*, vol. 19, no. 8, pp. 957-963, Aug. 2000.

[Jiang 01] Jiang, W. and B. Vinnakota, "Defect-Oriented Test Scheduling," *IEEE Trans. VLSI Systems*, vol. 9, no. 3, pp. 427-438, Jun. 2001.

[Kapur 92] Kapur, R., J. Park, and M. R. Mercer, "All Tests for a Fault are Not Equally Valuable for Defect Detection," *Proc. Intl. Test Conf.*, pp. 762-769, 1992.

[Lin 01] Lin, X, J. Rajski, I. Pomeranz, and S. M. Reddy, "On Static Test Compaction and Test Pattern Ordering for Scan Designs," *Proc. Intl. Test Conf.*, pp. 1088-1097, 2001.

[Ma 99] Ma, S., I. Shaik, and R. S. Fetherston, "A Comparison of Bridging Fault Simulation Methods," *Proc. Intl. Test Conf.*, pp. 587-595, 1999.

[Maly 86] Maly, W., "Optimal Order of the VLSI IC Testing Sequence," *Proc. Design Automation Conf.*, pp. 560-566, 1986.

[McCluskey 93] McCluskey, E. J., "Quality and Single-Stuck Faults," *Proc. Intl. Test Conf.*, p. 597, 1993.

[McCluskey 04a] McCluskey, E. J., et al., "ELF-Murphy Data on Defects and Test Sets," *Proc. VLSI Test Symp.*, pp. 16-22, 2004.

[McCluskey 04b] McCluskey, E. J., "Digital IC Testing for Art Historians and Test Experts," ICCD'04 presentation, http://crc.stanford.edu/iccd_ks.pdf.

[Mitra 04] Mitra, S., E. H. Volkerink, E. J. McCluskey, and S. Eichenberger, "Delay Defect Screening using Process Monitor Structures," *Proc. VLSI Test Symp.*, pp. 43-52, 2004.

[Nigh 00] Nigh, P. and A. Gattiker, "Test Method Evaluation Experiments & Data," *Proc. Intl. Test Conf.*, pp. 454-463, 2000.

[Pomeranz 04] Pomeranz, I. and S. M. Reddy, "On Maximizing the Fault Coverage for a Given Test Length Limit in a Synchronous Sequential Circuit," *IEEE Trans. Computers*, vol. 53, no. 9, pp. 1121-1133, Sep. 2004.

[Synopsys 05] TetraMAX ATPG User Guide, Synopsys, Inc., Jan. 2005.

[Tian 05] Tian, Y., M. R. Grimaila, W. Shi, and M. R. Mercer, "An Optimal Test Pattern Selection Method to Improve the Defect Coverage," *Proc. Intl. Test Conf.*, Paper 31.2, 2005.

Appendix C

California Scan Architecture for High Quality and Low Power Testing

Proceedings of International Test Conference (ITC), Paper 25.3, 2007.

CALIFORNIA SCAN ARCHITECTURE FOR HIGH QUALITY AND LOW POWER TESTING

Kyoung Youn Cho, Subhasish Mitra, and Edward J. McCluskey
Center for Reliable Computing (CRC)
Department of Electrical Engineering
Stanford University, Stanford, CA
{kycho, smitra, ejm}@crc.stanford.edu

Abstract

This paper presents a scan architecture—California scan—that achieves high quality and low power testing by modifying test patterns in the test application process. The architecture is feasible because most of the bits in the test patterns generated by ATPG tools are don't-care bits. Scan shift-in patterns have their don't-care bits assigned using the repeat-fill technique, reducing switching activity during the scan shift-in operation; the scan shift-in patterns are altered to toggle-fill patterns when they are applied to the combinational logic, improving defect coverage.

1. Introduction

California scan architecture (CSA) is a minor modification of traditional scan architecture (TSA). Its benefits are (1) reduced power consumption during test pattern scan shift-in and (2) improved defect coverage. These benefits are made possible by applying a modified version, rather than an exact copy, of the scan shift-in pattern to the combinational logic.

This technique is feasible because most of the bits in test patterns generated by automatic test pattern generation (ATPG) tools are don't-care bits. Table 1 reports the percentage of don't-care bits for the largest I99T circuits [Corno 00] from the ITC'99 benchmark suite [Basto 00] and the Stanford ELF18 circuit [Brand 04]. The test sets are generated using a commercial ATPG tool with a maximal compaction option. The results reveal that a large percentage of bits in single stuck-at fault (SSF) and

transition fault (TF) test patterns are don't-care bits. Furthermore, it has been reported that 95% to 99% of the bits in patterns for large industrial circuits are don't-care bits [Hiraide 03][Butler 04]. Examples of don't-care bit assignments are 0-fill, 1-fill, random-fill, and repeat-fill.

Table 1 Percentage of don't-care bits

Circuit	SSF test set	TF test set
b17	86.7 %	88.3 %
b18	88.0 %	91.8 %
b19	92.5 %	95.8 %
b20	71.9 %	70.0 %
b21	74.6 %	71.1 %
b22	71.4 %	74.4 %
ELF18	83.6 %	87.0 %

The *repeat-fill* method assigns don't-care bits using the last care bit. The *0-fill* technique assigns logic-0 to each don't-care bit while the *1-fill* method assigns logic-1. These don't-care bit assignment methods limit switching activity during the scan shift-in operation, although it may reduce defect coverage. On the other hand, the *random-fill* method assigns don't-care bits using pseudo-random bits. The *toggle-fill* technique assigns don't-care bits using logic-0 and logic-1 alternately. These assignments can improve defect coverage while they increase switching activity during the scan shift-in operation

In CSA, the test patterns that are shifted into scan chains have their don't-care bits assigned using the repeat-fill technique. These assignments limit the amount of toggling of scan cells during the scan shift-in operation. The patterns applied to the combinational logic are altered in the test application process so that they correspond to toggle-fill patterns. This modification of patterns enhances the defect coverage of test patterns. Table 2 shows an example of entering a scan shift-in pattern into a scan chain and applying an altered pattern to the combinational logic, in which the scan cell 1 is connected to a scan-out pin. A *test cube* is a deterministic test pattern generated by ATPG tools without assigning don't-care bits [Wang 06].

Table 2 Example of test pattern modification

Scan cell	8	7	6	5	4	3	2	1
Test cube	d	**1**	**1**	d	d	d	d	**0**
Scan-in pattern	1	**1**	**0**	0	0	0	0	**0**
Applied pattern	0	**1**	**1**	0	1	0	1	**0**

Table 3 illustrates the alteration scheme for applying the patterns. Note that the underlined entry in Table 3 differs from the value in the test cube since it will be complemented when applied to the combinational logic.

Table 3 Example of correspondence between scan shift-in patterns and patterns applied to the combinational logic

Scan-in pattern	Q_8	Q_7	Q_6	Q_5	Q_4	Q_3	Q_2	Q_1
Applied pattern	$\overline{Q}_8$	Q_7	$\overline{Q}_6$	Q_5	$\overline{Q}_4$	Q_3	$\overline{Q}_2$	Q_1

Table 4 shows the states of scan cells during the scan shift-in operation of the pattern in Table 2. The right-most bit in the "Scan-in pattern" column enters the scan chain. During the scan shift-in operation, the repeat-fill pattern is modified to toggle-fill pattern. Note that the amount of toggling of each scan cell is limited because don't-care bits are assigned using the repeat-fill technique.

Table 4 States of scan cells during scan shift-in

Scan clock	Scan-in pattern	Scan cell							
		8	7	6	5	4	3	2	1
1	1100000 0	1	?	?	?	?	?	?	?
2	110000 0	1	0	?	?	?	?	?	?
3	11000 0	1	0	1	?	?	?	?	?
4	1100 0	1	0	1	0	?	?	?	?
5	110 0	1	0	1	0	1	?	?	?
6	11 0	1	0	1	0	1	0	?	?
7	1 1	0	0	1	0	1	0	1	?
8	1	0	1	1	0	1	0	1	0

CSA can improve the quality of scan-based test sets, such as SSF and TF test sets. Two techniques can be used to detect TFs in a scan-based circuit: one is skewed-load, also known as launch-on-shift [Eichelberger 91][Savir 93]; the other is broadside, also known as launch-on-capture [Eichelberger 91][Savir 94]. The TF testing in this

paper uses the launch-on-capture technique to detect TFs. CSA can be used for any scan cell design, such as muxed-D scan design [Williams 73], two-port flip-flop design (clocked-scan design) [McCluskey 86], and level-sensitive scan design (LSSD) [Eichelberger 77].

This paper is organized as follows: Section 2 presents CSA and simulation results that show the effectiveness of CSA; Section 3 discusses scan shift-out switching activity; Section 4 reports experimental results that support the effectiveness of CSA; Section 5 concludes this paper; Section 6 summarizes related research.

2. California Scan Architecture

As presented in Sec. 1, most of the bits in deterministic test patterns are don't-care bits. To reduce power consumption during scan shift-in operation, don't-care bits can be assigned using the repeat-fill technique. This assignment strategy, however, may reduce the fortuitous detection of untargeted faults or defect coverage. To enhance defect coverage, don't-care bits can be assigned using the random-fill technique. But this approach increases overall power consumption during the scan shift-in operation. Therefore, in scan based testing, reducing power consumption during test application and improving defect coverage are conflicting goals. CSA provides a trade-off between test quality and power consumption by modifying test patterns during the scan shift-in operation. CSA is feasible because most of the bits in deterministic test patterns generated by ATPG tools are don't-care bits.

In CSA, scan shift-in patterns have their don't-care bits assigned using the repeat-fill method, limiting the amount of toggling of scan cells during scan shift-in operation. In the test application process, the scan shift-in patterns are modified so that the patterns applied to the combinational logic become toggle-fill patterns. Figure 1 illustrates two CSA implementations. Scan enable and clock signals are not included in the figures.

Figure 1(a) shows a CSA implementation that inserts an inverter at the scan-in (SI) input of each scan flip-flop. Figure 1(b) illustrates another implementation that connects the SI input of each scan flip-flop to the $\overline{Q}$ output, rather than Q output, of

the previous scan flip-flop. CSA does not change the combinational logic part of the system. In the CSA implementations, the test patterns applied to the combinational logic are different from the scan shift-in patterns. For example, when the scan shift-in pattern is "000000," the pattern applied to the combinational logic in Fig. 1(a) becomes "101010" assuming the right-most bit is the first bit to enter the scan chain. Similarly, the pattern applied to the combinational logic in Fig. 1(b) becomes "010101." One assumption of CSA is that toggle-fill patterns are more effective in detecting defects than repeat-fill patterns. Experimental results that support this assumption will be presented in Sec. 4 using the Stanford ELF18 test chips [Brand 04].

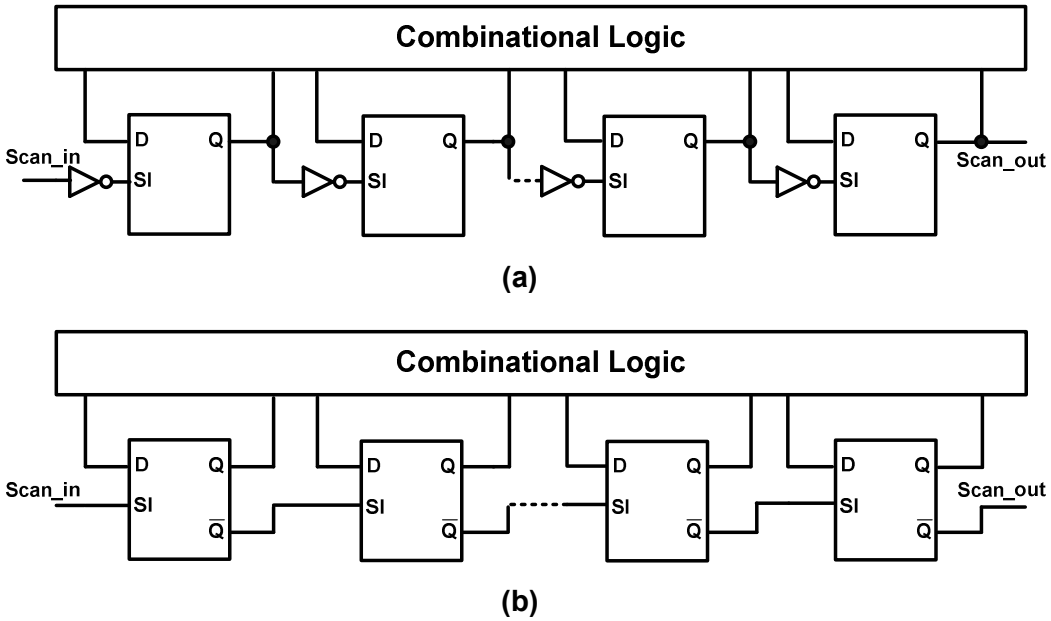


Figure 1 California scan architectures:
(a) An implementation using inverters; (b) An implementation using $\overline{Q}$ signals of scan flip-flops

In this paper, the six largest I99T benchmark circuits [Corno 00] are used to show the implementation and effectiveness of CSA. The synthesis of the benchmark circuits and scan insertion are conducted using the Synopsys Design Compiler [Synopsys]; only one scan chain is inserted into each circuit to simplify the simulation process. After the circuits are synthesized, a PERL script inserts an inverter at the SI

input of each scan flip-flop. Table 5 reports the synthesis results of the benchmark circuits: the number of scan flip-flops, primary inputs, and primary outputs.

Table 5 Benchmark circuits used in this research

Circuit	Scan flip-flops	Primary inputs	Primary outputs
b17	1,315	41	97
b18	3,016	40	24
b19	6,642	49	31
b20	430	36	23
b21	490	36	23
b22	613	36	23

Figure 2 illustrates the overall simulation flow. Test sets are generated using the Synopsys TetraMAX [Synopsys] with a maximal compaction option and without don't-care bit assignments; TSA implementations of the benchmark circuits are used to generate test sets. To apply the test sets generated for TSA netlists to corresponding CSA netlists, some care bits are complemented so that the same care bits are applied to the combinational logic (note the mapping in Table 3).

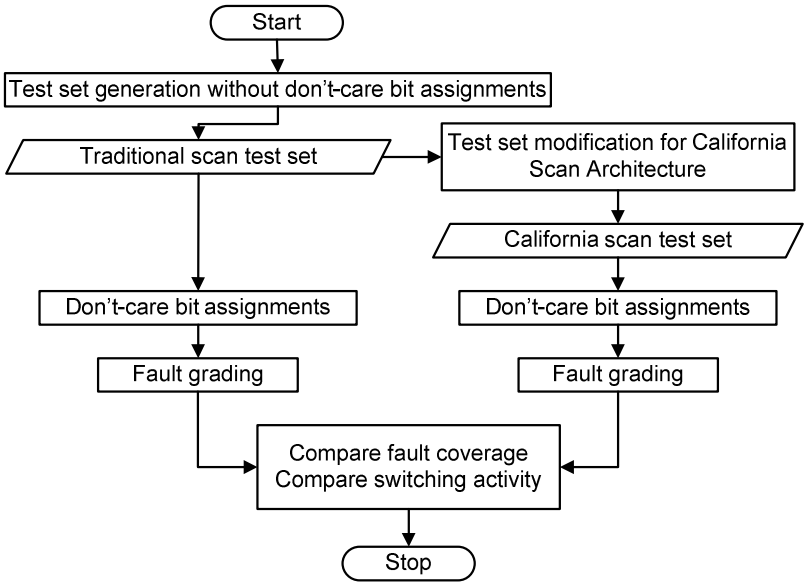


Figure 2 Simulation flow

Power consumption and defect detection are compared to show the effectiveness of CSA. Table 6 shows the definitions for scan architectures and don't-care bit assignments.

Table 6 Scan architecture and don't-care bit assignments

Definition	Scan architecture	Don't-care bit assignments
TSA_rpt	Traditional scan	Repeat-fill
TSA_rnd	Traditional scan	Random-fill
CSA_rpt	California scan	Repeat-fill

To estimate power consumption during the scan shift operation, the number of weighted transitions (WTs) [Sankaralingam 00] is used. The method counts the switching activity of scan flip-flops during the scan shift-in and shift-out operations; a larger number represents more power consumption. It is shown that there is a close correlation between WTs and state transitions in the combinational logic [Sankaralingam 00]. Assuming the scan chain illustrated in Fig. 3, the numbers of WTs of the scan flip-flops during scan shift-in and shift-out operations are calculated using Expressions 1 and 2, respectively. In Expressions 1 and 2, x_i is a scan shift-in or scan shift-out logic value corresponding to scan flip-flop $\mathbf{x_i}$, and $\oplus$ is an exclusive-OR operation. The expressions can be applied to both TSA and CSA. To estimate capture power consumption, the number of scan flip-flops that have different logic values between applied patterns and captured responses is used.

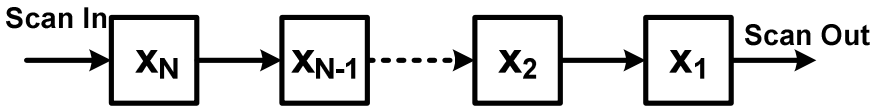


Figure 3 Example scan chain

$$WT_{\text{scan–in}} = \sum_{i=1}^{N-1} (N-i)(x_i \oplus x_{i+1}) \quad (1)$$

$$WT_{\text{scan–out}} = \sum_{i=1}^{N-1} (i)(x_i \oplus x_{i+1}) \quad (2)$$

The defect coverage of a test set is estimated using the average number of detected SSFs and TFs per pattern and *N*-detect coverage [Ma 95]. The use of the average number of detected faults per pattern as a metric is based on the report that multiple detection of faults improves the defect detection of test sets [Ma 95][Grimaila 99] [Amyeen 04]. Each test pattern is fault simulated without dropping detected faults to find the average number per pattern. In this research, only the faults existing in the

combinational logic are considered. All the coverage values are calculated as test coverage; i.e., untestable faults are not considered during the fault simulation.

In the case of the SSF test sets, the numbers of observed gate input combinations are also compared. An *observed gate input combination* of an internal gate is a logic combination applied to the gate inputs with the gate output being sensitized to at least one observation point, such as a primary output or a scan flip-flop [Cho 05]. The number of observed gate input combinations shows a better correlation with defect detection than the SSF coverage or the bridge coverage estimate [Cho 05][Guo 06].

Figure 4 illustrates the simulation flow using an example. The test cube used in the example for a TSA circuit is "00dddddd11," in which "d" is a don't-care bit. The test cube for the corresponding CSA circuit is "10dddddd01"; two care bits underlined are complemented. In TSA, don't-care bits are assigned using the repeat-fill and random-fill methods; in CSA, don't-care bits are assigned using only the repeat-fill technique. Figure 4 also presents scan shift-in patterns and test patterns applied to the combinational logic. In TSA, all the scan flip-flops are assigned the same logic value as in scan shift-in patterns; in CSA, alternate scan flip-flops are assigned complemented logic values of the corresponding bits in scan shift-in patterns. During test application, the care bits applied to the combinational logic are the same for the three configurations. Thus, the differences in the fault coverage or the number of detected faults are only the effect of don't-care bit assignments.

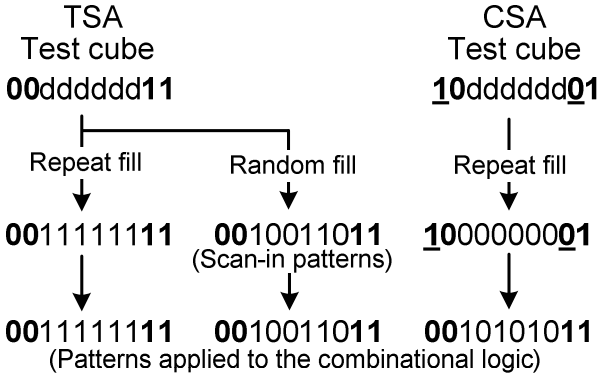


Figure 4 An example simulation flow

The switching activity for care bits can be different. For example, in Fig. 4, the scan shift-in patterns for TSA_rpt and CSA_rpt are "0011111111" and "1000000001," respectively. Assuming that the right-most bit enters a scan chain first, the numbers of WTs for TSA_rpt and CSA_rpt are 2 and 10, respectively.

Table 7 provides the simulation results for the SSF test sets. The first four columns in Table 7 present the circuit name, the configuration, the number of test patterns, and the SSF test coverage, respectively. The "Detected SSFs" column reports the average number of detected SSFs per pattern. The last column of Table 7 provides the average switching activity per pattern during test process. The average number is calculated from scan shift-in, capture, and scan shift-out switching activity. In this paper, the "Ratio" column shows the number normalized to that of **TSA_rnd**. In the case of the largest benchmark circuit (b19), the average number of detected SSFs of **CSA_rpt** is 96.7 % that of **TSA_rnd** while the average switching activity is only 15.5 % that of **TSA_rnd**.

Table 7 Simulation results: SSF test sets

Circuit	Configuration	Test length	SSF test coverage	Detected SSFs		Observed GIC		Switching activity	
				Average	Ratio	Total	Ratio	Average	Ratio
b17	**TSA_rpt**	438	99.6	3,268	0.834	67,687	0.882	98,884	0.121
	CSA_rpt	438	99.6	3,527	**0.900**	69,643	**0.918**	195,284	**0.238**
	TSA_rnd	438	99.6	3,918	1.000	76,039	1.000	820,487	1.000
b18	**TSA_rpt**	718	99.7	7,771	0.806	186,033	0.894	493,848	0.114
	CSA_rpt	718	99.7	8,965	**0.930**	190,276	**0.914**	930,717	**0.214**
	TSA_rnd	718	99.7	9,637	1.000	208,197	1.000	4,349,002	1.000
b19	**TSA_rpt**	821	99.7	20,428	0.823	361,223	0.849	2,067,661	0.097
	CSA_rpt	821	99.7	24,014	**0.967**	368,376	**0.866**	3,298,596	**0.155**
	TSA_rnd	821	99.7	24,834	1.000	425,298	1.000	21,229,982	1.000
b20	**TSA_rpt**	692	100	1,752	0.915	36,310	0.956	22,330	0.260
	CSA_rpt	692	100	1,808	**0.944**	36,793	**0.969**	36,455	**0.424**
	TSA_rnd	692	100	1,914	1.000	38,003	1.000	86,049	1.000
b21	**TSA_rpt**	473	99.9	2,324	0.984	29,985	0.952	23,391	0.216
	CSA_rpt	473	99.9	2,234	**0.946**	30,395	**0.965**	49,156	**0.454**
	TSA_rnd	473	99.9	2,362	1.000	31,504	1.000	108,237	1.000
b22	**TSA_rpt**	711	100	2,633	0.918	55,079	0.953	45,955	0.264
	CSA_rpt	711	100	2,712	**0.945**	56,128	**0.971**	76,158	**0.437**
	TSA_rnd	711	100	2,869	1.000	57,792	1.000	174,414	1.000
The "Ratio" column gives the number normalized to that of **TSA_rnd**.									

The "Observed GIC" column presents the number of observed gate input combinations obtained by gate exhaustive simulation [Cho 05]. For each benchmark circuit, **CSA_rpt** observes more gate input combinations than **TSA_rpt**, although the number is smaller than that of **TSA_rnd**.

Figure 5 compares the average number of detected SSFs per pattern and the average switching activity per pattern for each benchmark circuit. In the graphs, each number is normalized to that of corresponding **TSA_rnd**. Except the b21 circuit, the number of **CSA_rpt** is between **TSA_rpt** and **TSA_rnd**. The results show that CSA can be a good trade-off between power consumption and defect detection.

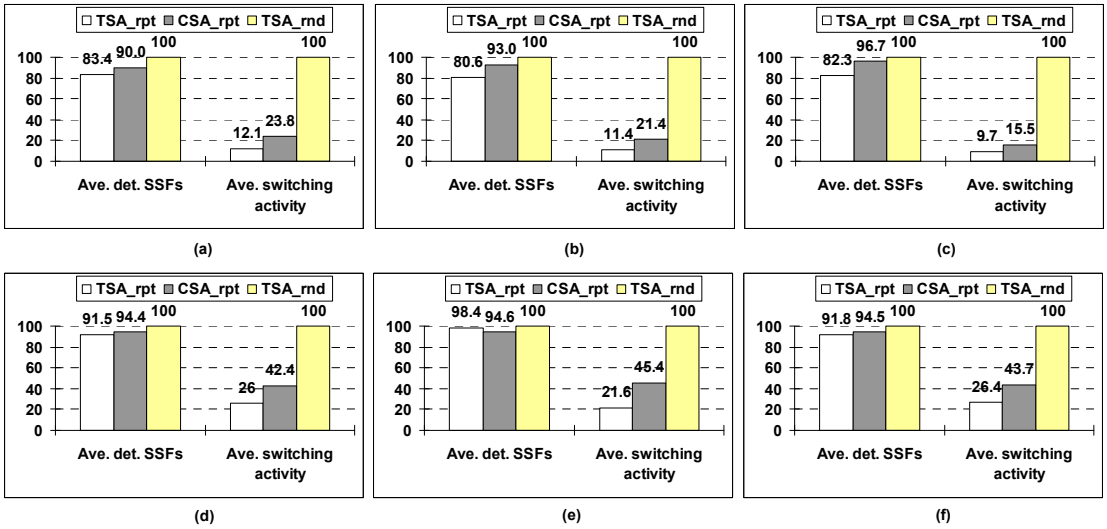


Figure 5 Comparison of average number of detected SSFs per pattern and average switching activity per pattern; the number of TSA_rnd is used as a reference: (a) b17; (b) b18; (c) b19; (d) b20; (e) b21; (f) b22

Table 8 provides the simulation results for the TF test sets. The "TF test coverage" and "Detected TFs" columns report the TF test coverage and the average number of detected TFs per pattern, respectively. Except the b22 circuit, the average number of the detected TFs per pattern of **CSA_rpt** is larger than that of **TSA_rpt**. Figure 6 demonstrates that the increases in the number of detected TFs of **CSA_rpt** from **TSA_rpt** are higher than the increases in power consumption. For example, in the case of the b19 circuit, the average number of the detected TFs of **TSA_rpt** is

85.5% that of **TSA_rnd**, which increases to 94.4% by applying CSA; on the other hand, the power consumption increases from 7.3% to 15.4% that of **TSA_rnd**.

Table 8 Simulation results: TF test sets

Circuit	Configuration	Test length	TF test coverage	Detected TFs		Switching activity	
				Average	Ratio	Average	Ratio
b17	**TSA_rpt**	991	91.0	822	0.951	96,545	0.116
	CSA_rpt	991	91.0	888	**1.028**	162,873	**0.196**
	TSA_rnd	991	91.0	864	1.000	832,254	1.000
b18	**TSA_rpt**	1,378	91.7	2,193	0.890	494,510	0.112
	CSA_rpt	1,378	91.7	2,406	**0.977**	866,492	**0.197**
	TSA_rnd	1,378	91.7	2,463	1.000	4,396,664	1.000
b19	**TSA_rpt**	2,714	95.2	4,114	0.855	1,560,428	0.073
	CSA_rpt	2,714	95.2	4,543	**0.944**	3,267,048	**0.154**
	TSA_rnd	2,714	95.2	4,814	1.000	21,234,101	1.000
b20	**TSA_rpt**	1,104	97.5	642	0.982	25,691	0.301
	CSA_rpt	1,104	97.5	649	**0.992**	40,473	**0.474**
	TSA_rnd	1,104	97.5	654	1.000	85,464	1.000
b21	**TSA_rpt**	677	96.5	745	0.949	30,310	0.282
	CSA_rpt	677	96.5	775	**0.988**	56,357	**0.525**
	TSA_rnd	677	96.5	785	1.000	107,379	1.000
b22	**TSA_rpt**	1,333	97.7	943	0.992	48,196	0.275
	CSA_rpt	1,333	97.7	924	**0.973**	74,356	**0.425**
	TSA_rnd	1,333	97.7	950	1.000	175,145	1.000
The "Ratio" column gives the number normalized to that of **TSA_rnd**.							

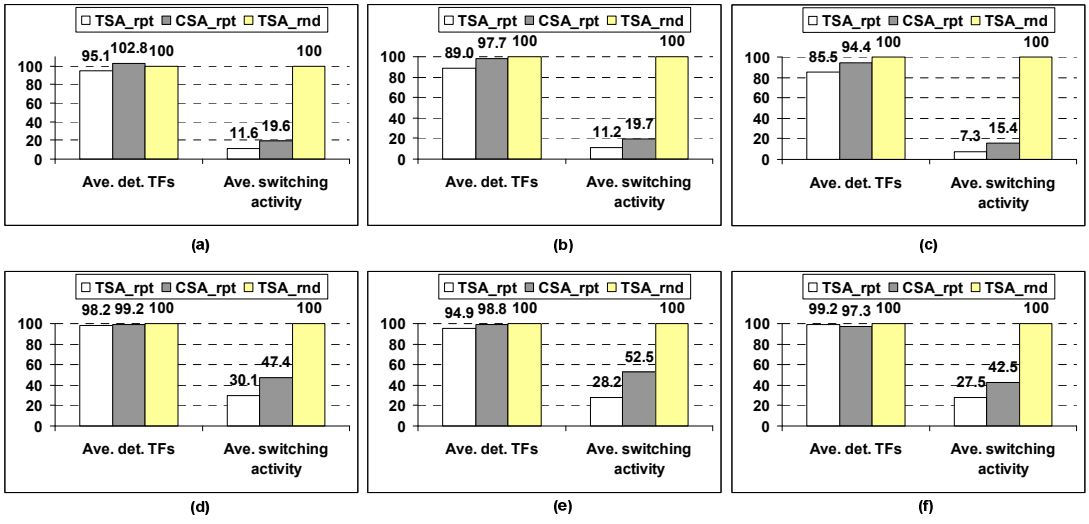


Figure 6 Comparison of average number of detected TFs per pattern and switching activity per pattern; the number of TSA_rnd is used as a reference: (a) b17; (b) b18; (c) b19; (d) b20; (e) b21; (f) b22

The switching activity of **CSA_rpt** is larger than that of **TSA_rpt** because many care bits are clustered, and the same logic value is assigned to the clusters of care bits. In the CSA circuits, alternate care bits are complemented in scan shift-in patterns, which may increase switching activity during the scan shift-in operation.

Table 9 reports the *N*-detect test coverage [Ma 95] for the SSF test sets of I99T benchmark circuits. The overall trends show that the *N*-detect coverage of **CSA_rpt** is between that of **TSA_rpt** and **TSA_rnd**.

Table 9 *N*-detect coverage: SSF test sets

Circuit	Conf.	2-det	5-det	10-det	15-det
b17	**TSA_rpt**	73.5	57.5	45.2	35.3
	CSA_rpt	73.4	57.7	45.9	36.5
	TSA_rnd	73.6	59.3	47.5	37.5
b18	**TSA_rpt**	75.8	59.5	45.7	36.8
	CSA_rpt	75.9	59.8	46.2	37.3
	TSA_rnd	77.4	63.6	49.7	40.8
b19	**TSA_rpt**	72.4	57.2	45.8	39.4
	CSA_rpt	72.4	58.2	46.4	39.8
	TSA_rnd	73.9	61.0	50.2	43.7
b20	**TSA_rpt**	77.8	53.4	34.7	29.3
	CSA_rpt	78.4	53.0	35.4	29.7
	TSA_rnd	78.3	53.8	35.5	29.5
b21	**TSA_rpt**	78.2	57.1	42.6	38.3
	CSA_rpt	78.1	58.2	44.2	39.0
	TSA_rnd	78.5	58.2	44.3	39.5
b22	**TSA_rpt**	78.4	54.5	36.0	30.7
	CSA_rpt	79.0	55.2	36.1	30.4
	TSA_rnd	79.0	54.8	36.3	30.4

Don't-care bits can be assigned using care bits during test set generation; this strategy will reduce the number of test patterns by detecting untargeted faults fortuitously. Table 10 reports the results of b19 SSF test sets for the three configurations. The number in parenthesis represents the ratio to the number in **TSA_rnd**. Figure 7 demonstrates that CSA increases the average number of detected SSFs from 83% to 97.9% that of **TSA_rnd** while it increases power consumption only from 10.6% to 17.8%.

Table 10 SSF test generation results with don't-care bits being assigned during test generation (b19)

	TSA_rpt	CSA_rpt	TSA_rnd
SSF test coverage	99.7 %	99.7 %	99.7 %
Test length	575	565	591
Ave. det. SSFs	20,146 (0.830)	23,765 (**0.979**)	24,283 (1.000)
Ave. Scan-in activity	2,246,454 (0.106)	3,772,448 (**0.178**)	21,188,266 (1.000)

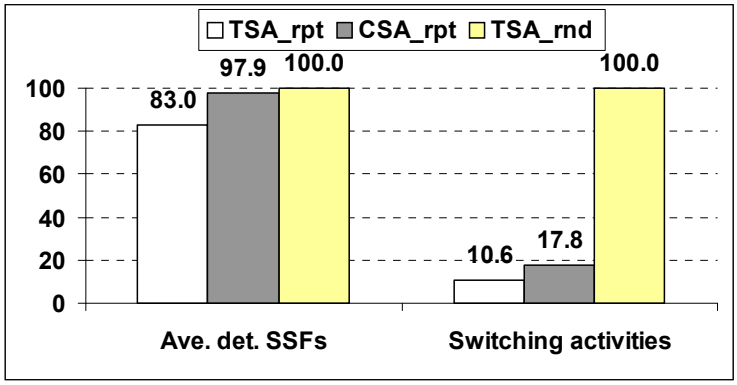


Figure 7 Average number of detected SSFs per pattern and average switching activity per pattern (b19); the number is normalized to that of TSA_rnd

The simulation results show that CSA is more effective in the b19 circuit than other benchmark circuits. The b19 circuit is the largest among the benchmark circuits, and the test sets for the circuit contain the largest percentage of don't-care bits. This shows that the effectiveness of CSA may increase when it is applied to circuits that contain more don't-care bits in their patterns than the small benchmark circuits used in this research.

3. Scan Shift-Out Switching Activity

CSA controls the power consumption during the scan shift-in operation, whereas the power consumption during the scan shift-out operation depends on the captured patterns, which are determined by the applied patterns and the circuit structure. This section further investigates power consumption during the scan shift-out operation.

Table 11 reports scan shift-in and shift-out switching activity of the SSF test sets for the I99T benchmark circuits. In the case of the b19 circuit, the percentages of switching activity of TSA_rpt and CSA_rpt during the scan shift-in operation are 7.3% and 12.3% that of TSA_rnd, respectively; the percentages of switching activity of TSA_rpt and CSA_rpt during the scan shift-out operation are 12.3% and 18.8%, respectively.

Table 11 Scan-in and scan-out switching activity: SSF test sets

Cir.	Config.	Scan-in		Scan-out	
		Average	Ratio	Average	Ratio
b17	**TSA_rpt**	38,587	0.093	60,180	0.149
	CSA_rpt	70,329	**0.169**	124,819	**0.309**
	TSA_rnd	416,901	1.000	403,464	1.000
b18	**TSA_rpt**	209,924	0.096	283,649	0.131
	CSA_rpt	386,293	**0.177**	544,163	**0.252**
	TSA_rnd	2,184,849	1.000	2,163,898	1.000
b19	**TSA_rpt**	782,128	0.073	1,284,906	0.123
	CSA_rpt	1,328,231	**0.123**	1,969,828	**0.188**
	TSA_rnd	10,764,761	1.000	10,464,679	1.000
b20	**TSA_rpt**	9,106	0.209	13,164	0.310
	CSA_rpt	14,458	**0.332**	21,942	**0.517**
	TSA_rnd	43,544	1.000	42,450	1.000
b21	**TSA_rpt**	9,403	0.168	13,902	0.266
	CSA_rpt	17,523	**0.313**	31,570	**0.604**
	TSA_rnd	55,944	1.000	52,231	1.000
b22	**TSA_rpt**	18,309	0.208	27,555	0.319
	CSA_rpt	30,327	**0.345**	45,750	**0.530**
	TSA_rnd	87,953	1.000	86,379	1.000

Figure 8 illustrates the scan shift-in and shift-out switching activity normalized to the total switching activity of TSA_rnd. The simulation results reveal that switching activity during the scan shift-out operation follows a trend similar to switching activity during the scan shift-in operation.

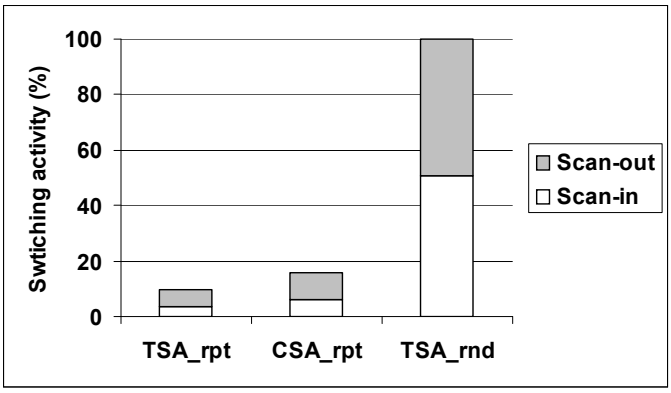


Figure 8 Comparison of switching activity: b19 SSF test sets

One explanation of the trend is that not all scan flip-flops change states between applied patterns and captured patterns. Table 12 reports the percentage of scan flip-flops that have different logic values between the applied patterns and the captured patterns for the SSF test sets. The percentage is less than 17.6%. Therefore there is a correlation between scan shift-in patterns and scan shift-out patterns in the benchmark circuits.

Table 12 Percentage of the time that scan flip-flops have different logic values between the applied patterns and captured patterns: SSF test sets

Circuit	TSA_rpt	CSA_rpt	TSA_rnd
b17	8.9	10.3	9.3
b18	9.1	8.7	8.5
b19	9.4	8.1	8.2
b20	14.0	12.8	12.8
b21	17.6	12.7	12.7
b22	14.8	13.2	13.4

To show the dependency of scan shift-out switching activity on the synthesis library, the b19 circuit is synthesized using a different technology library. Table 13 reports the switching activity of an SSF test set. The results show that the switching activity of **CSA_rpt** during scan shift-out operation is 27.8% that of **TSA_rnd**. The switching activity normalized to the total switching activity of **TSA_rnd** is presented in Fig. 9. The percentage of scan shift-out switching activity is a little higher than the result in Table 11; however, the overall trend in switching activity during the scan shift operation is similar as presented in Figs. 8 and 9.

Table 13 Scan shift switching activity: b19 SSF test sets

Average switching activity	TSA_rpt	CSA_rpt	TSA_rnd
Scan-in	750,392	1,643,218	8,657,669
Scan-out	1,123,262	2,393,407	8,597,605
Total	1,873,654	4,036,625	17,255,274

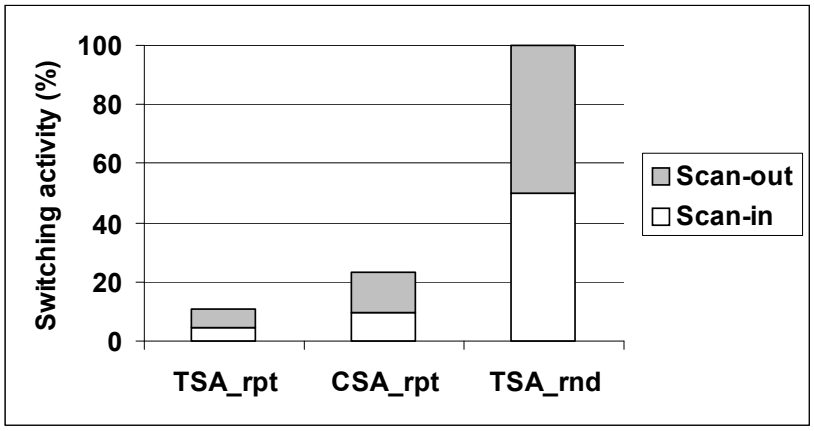


Figure 9 Comparison of switching activity: b19 SSF test sets

Table 14 provides the average number of detected SSFs per pattern for the circuits synthesized using the different technology library. In this case, the average number of **CSA_rpt** is larger than that of **TSA_rnd**.

Table 14 Average number of detected SSFs per pattern

Configuration	Detected SSFs per pattern	
	Average	Ratio
TSA_rpt	27,409	0.895
CSA_rpt	30,719	1,003
TSA_rnd	30,633	1.000

The switching activity of SSF test sets for the ELF18 core is also compared. The percentages of scan flip-flops that change states during the capture period are 15.0%, 22.1%, and 22.3% for **TSA_rpt**, **CSA_rpt**, and **TSA_rnd**, respectively. Table 15 shows that the average switching activity of **CSA_rpt** during the scan shift-out operation is 83.8% that of **TSA_rnd** for the ELF18 SSF test sets. In the case of the ELF18 SSF test sets, the average numbers of switching activity of **TSA_rpt** and **CSA_rpt** during the scan shift operation are 15% and 48% that of **TSA_rnd**,

respectively. Figure 10 illustrates the relative switching activity normalized to the total switching activity of **TSA_rnd**.

Table 15 Scan shift switching activity: an ELF18 SSF test set

Average switching activity	**TSA_rpt**	**CSA_rpt**	**TSA_rnd**
Scan-in	9,428	18,135	99,737
Scan-out	20,781	67,602	80,696
Total	30,209	85,737	180,433

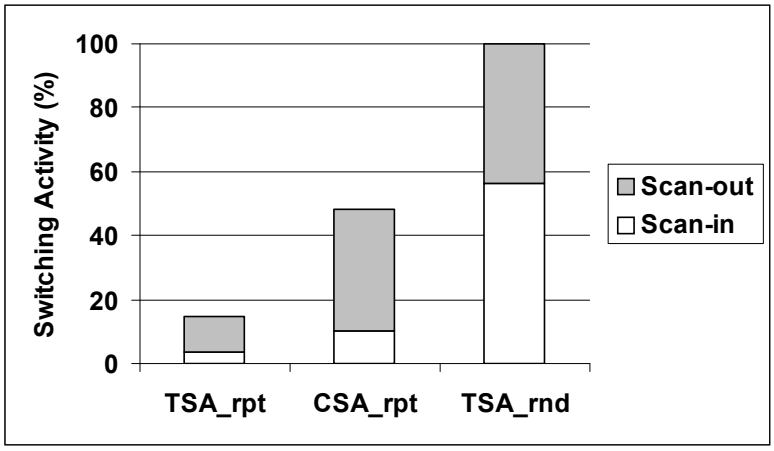


Figure 10 Comparison of switching activity: ELF18 SSF test sets

The simulation results reveal that CSA reduces switching activity during the scan shift-in operation. Even though switching activity during the scan shift-out operation depends on the circuit structure, the simulation results show that CSA also reduces scan shift-out switching activity. In scan-based testing, the scan shift-in and shift-out operations are conducted concurrently. Therefore total switching activity during the scan shift operation can be reduced by reducing scan shift-in switching activity, even though the reduction in scan shift-out switching activity is not significant.

4. Experimental Results

The Stanford ELF18 test chips [Brand 04] are used to compare the correlation between defect detection and don't-care bit assignments. The ELF18 test chips were fabricated using the Philips 0.18 micron Corelib technology, and the nominal supply

voltage is 1.8V. More than 70,000 test chips were manufactured; each test chip contains 6 R.E.A.L.^{TM} digital signal processor cores. One ELF18 core contains 13 scan chains; the total number of scan flip-flops is 2,290; the total number of gates for one core is 53,732.

A SSF test set and a TF test set are generated with a maximal compaction option and without don't-care bit assignments using the Synopsys TetraMAX [Synopsys]. Then, the test sets have their don't-care bits assigned using three techniques: repeat-fill, toggle-fill, and random-fill. Table 16 describes the test sets.

Table 16 Test sets: ELF18

Test set	Test set type	Don't-care bit assignments
SSF_rpt	Single stuck-at fault	Repeat-fill
SSF_tgl	Single stuck-at fault	Toggle-fill
SSF_rnd	Single stuck-at fault	Random-fill
TF_rpt	Transition fault	Repeat-fill
TF_tgl	Transition fault	Toggle-fill
TF_rnd	Transition fault	Random-fill

Tables 17 and 18 report the simulation results of the SSF test sets and the TF test sets, respectively. The "Ratio" column shows the average number of detected faults per pattern normalized to that of **SSF_rnd** or **TF_rnd**. The trend of the ELF18 simulation results is similar to that of I99T benchmark circuit results. One exception is the fact that the average number of SSFs detected by **SSF_tgl** is larger than that of SSFs detected by **SSF_rnd**.

Table 17 SSF test set simulation results: ELF18

Test set	Test length	SSF coverage [%]	Detected SSFs	
			Average	Ratio
SSF_rpt	457	99.5	9,940	0.821
SSF_tgl	457	99.5	12,684	**1.048**
SSF_rnd	457	99.5	12,102	1.000

Table 18 TF test set simulation results: ELF18

Test set	Test length	TF coverage [%]	Detected TFs	
			Average	Ratio
TF_rpt	1,472	97.2	1,741	0.694
TF_rnd	1,472	97.2	2,503	**0.997**
TF_tgl	1,472	97.2	2,511	1.000

All test sets are applied at nominal voltage (1.8V). In the case of the TF test sets, the launch/capture clock speed is a speed with 10% margin from the maximum operation speed determined from the shmoo plots of defect-free cores. In this experiment, 1,604 cores are tested.

Figure 11 illustrates the fallout of defective cores. The experiments reveal that the random-fill and toggle-fill assignments are more effective in detecting defects than the repeat-fill assignments. Note that the difference in the defect detection among the test sets is only the effect of don't-care bit assignments. The experimental results using the ELF18 test chips demonstrate that CSA is more effective in detecting defects than TSA_rpt because CSA applies toggle-fill patterns to the combinational logic.

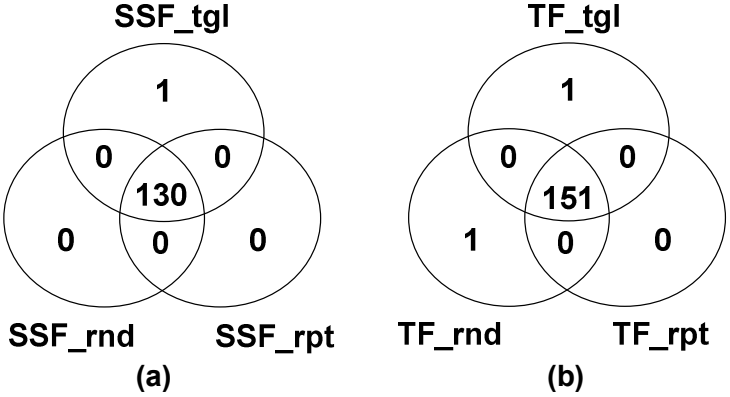


Figure 11 Detection of defective cores:
(a) SSF test sets; (b) TF test sets

5. Conclusions

This paper presents California scan architecture (CSA) that achieves high quality and low power testing by modifying test patterns during the scan shift-in

operation. CSA is feasible because most of the bits in test patterns generated by ATPG tools are don't-care bits.

CSA can be constructed by slightly modifying traditional scan architecture. Furthermore, CSA can be used without changing current design tools. The effectiveness of CSA with respect to power consumption and defect detection may increase when it is applied to circuits that contain more don't-care bits in their patterns than the benchmark circuits used in this paper.

Appendix: Review of Related Research

In modern VLSI design, full-scan design is usually used for better controllability and observability of internal states. Widely used scan designs are the muxed-D scan design [Williams 73] and the level-sensitive scan design (LSSD) [Eichelberger 77].

During the scan shift operation, the states of scan flip-flops change, causing many state transitions; these state transitions may cause good chips to fail the applied test set because of abnormal power consumption or excessive power/ground noise compared to normal operation [Nicolici 02][Sinanoglu 03][Yoshida 03]. To overcome the abnormal switching activity during the scan shift operation, an "enhanced" scan architecture [DasGupta 81] can be used to prevent the switching activity from propagating to the combinational logic while the architecture increases hardware cost. Cost-efficient implementations of gating logic to mask the switching activity during the scan shift operation are presented [Gerstendorfer 99][Bhunia 05]. The insertion of gating logic, however, introduces an additional delay in the normal operation. To reduce the maximum switching activity, scan chains are divided into multiple segments, and one segment is activated at a time [Saxena 01].

Scan chains are modified by inserting inverter and XOR gates into specific locations in scan chain paths to reduce switching activity during the scan shift operation; the scan-in data are transformed according to the modified structure [Sinanoglu 03]. This method changes scan chain paths for a specific test set that

contains only care bits while CSA utilizes don't-care bit assignments to improve defect detection and to reduce power consumption during testing.

ATPG tools generate test patterns with many don't-care bits; the percentage of don't-care bits for compacted test sets is between 95% and 99% [Hiraide 03][Butler 04]. Don't-care bit assignments can be utilized to compact test sets or to reduce power consumption during testing. Examples of don't-care bit assignments are 0-fill, 1-fill, random-fill, and repeat-fill. The repeat-fill method is an efficient method with respect to test set size and power consumption during test application [Butler 04]. On the other hand, don't-care bit assignments also affect the defect detection of test sets [McCluskey 04].

Some previous methods are test set dependent, i.e., scan chains are modified to reduce power consumption for a specific test set. In production testing, many test sets are applied; sometimes different test sets are generated after the design is completed. Moreover, scan flip-flops cannot be reordered freely because of the layout constraints [Makar 98][Bonhomme 03]. Therefore test set dependent methods are not practical solutions in production testing. CSA does not depend on a specific test set or scan flip-flop reordering. Furthermore, CSA can be used with other scan architectures and scan design methods such as scan cell stitching and reordering.

Acknowledgment

This research was supported partially by NSF under contract number CCR-0098128 and partially by LSI Logic.

The authors would like to express their thanks to Bill Price, Stefan Eichenberger, and Fred Bowen of NXP for providing the ELF18 test chips. The authors also thank Erik Volkerink of Verigy for providing tester time.

References

[Amyeen 04] Amyeen, M. E., S. Venkataraman, A. Ojha, and S. Lee, "Evaluation of the Quality of N-Detect Scan ATPG Patterns on a Processor," *Proc. Intl. Test Conf.*, pp. 669-678, 2004.

[Basto 00] Basto, L., "First Results of ITC'99 Benchmark Circuits," *IEEE Design & Test of Computers*, vol. 17, no. 3, pp. 54-59, Jul.-Sep. 2000.

[Bhunia 05] Bhunia, S., H. Mahmoodi, D. Ghosh, S. Mukhopadhyay, and K. Roy, "Low-Power Scan Design Using First-Level Supply Gating," *IEEE Trans. on VLSI Systems*, vol. 13, no. 3, pp. 384-395, Mar. 2005.

[Bonhomme 03] Bonhomme, Y., P. Girard, L. Guiller, C. Landrault, and S. Pravossoudovitch, "Efficient Scan Chain Design for Power Minimization During Scan Testing Under Routing Constraint," *Proc. Intl. Test Conf.*, pp. 488-493, 2003.

[Brand 04] Brand, K. A., S. Mitra, E. H. Volkerink, and E. J. McCluskey, "Speed Clustering of Integrated Circuits," *Proc. Intl. Test Conf* ., pp. 1128-1137, 2004.

[Butler 04] Butler, K. M., et al., "Minimizing Power Consumption in Scan Testing: Pattern Generation and DFT Techniques," *Proc. Intl. Test Conf.*, pp. 355-364, 2004.

[Cho 05] Cho, K. Y., S. Mitra, and E. J. McCluskey, "Gate Exhaustive Testing," *Proc. Intl. Test Conf.*, Paper 31.3, 2005.

[Corno 00] Corno, F., M. S. Reorda, and G. Squillero, "RT-level ITC'99 Benchmarks and First ATPG Results," *IEEE Design & Test of Computers*, vol. 17, no. 3, pp. 44-53, Jul.-Sep. 2000.

[DasGupta 81] DasGupta, S., R. G. Walther, T. W. Williams, and E. B. Eichelberger, "An Enhancement to LSSD and Some Applications of LSSD in Reliability, Availability, and Serviceability," *Proc. Intl. Symp. on Fault-Tolerant Computing*, pp. 32-34, 1981.

[Eichelberger 77] Eichelberger, E. B. and T. W. Williams, "A Logic Design Structure for LSI Testability," *Proc. Design Automation Conf.*, pp. 462-468, 1977.

[Eichelberger 91] Eichelberger, E. B., E. Lindbloom, J. A. Waicukauski, and T. W. Williams, "Delay-Fault Simulation," *Structured Logic Testing*, E. J. McCluskey (Ed.), Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1991.

[Gerstendorfer 99] Gerstendorfer, S. and H.-J. Wunderlich, "Minimized Power Consumption for Scan-Based BIST," *Proc. Intl. Test Conf.*, pp. 77-84, 1999.

[Grimaila 99] Grimaila, M. R. et al., "REDO - Random Excitation and Deterministic Observation – First Commercial Experiment," *Proc. VLSI Test Symp*., pp. 268-274, 1999.

[Guo 06] Guo, R., et al., "Evaluation of Test Metrics: Stuck-at, Bridge Coverage Estimate and Gate Exhaustive," *Proc. VLSI Test Symp*., pp. 66-71, 2006.

[Hiraide 03] Hiraide, T., et al., "BIST-Aided Scan Test—A New Method for Test Cost Reduction," *Proc. VLSI Test Symp*., pp. 359-364, 2003.

[Ma 95] Ma, S. C., P. Franco, and E. J. McCluskey, "An Experimental Chip to Evaluate Test Techniques Experiment Results," *Proc. Intl. Test Conf*., pp. 663-672, 1995.

[Makar 98] Makar, S., "A Layout-Based Approach for Ordering Scan Chain Flip-Flops," *Proc. Intl. Test Conf*., pp. 341-347, 1998.

[McCluskey 86] McCluskey, E. J., *Logic Design Principles: With Emphasis on Testable Semicustom Circuits*, Prentice Hall, Englewood Cliffs, NJ, 1986.

[McCluskey 04] McCluskey, E. J., et al., "ELF-Murphy Data on Defects and Test Sets," *Proc. VLSI Test Symp*., pp. 16-22, 2004.

[Nicolici 02] Nicolici, N. and B. M. Al-Hashimi, "Multiple Scan Chains for Power Minimization during Test Application in Sequential Circuits," *IEEE Trans. on Computers*, vol. 51, no. 6, pp. 721-734, Jun. 2002.

[Sankaralingam 00] Sankaralingam, R., R. R. Oruganti, and N. A. Touba, "Static Compaction Techniques to Control Scan Vector Power Dissipation," *Proc. VLSI Test Symp*., pp. 35-40, 2000.

[Savir 93] Savir, J. and S. Patil, "Scan-Based Transition Test," *IEEE Trans. on CAD*, vol. 12, no. 8, pp. 1232-1241, Aug. 1993.

[Savir 94] Savir, J. and S. Patil, "On Broad-Side Delay Test," *Proc. VLSI Test Symp*., pp. 284-290, 1994.

[Saxena 01] Saxena, J., K. M. Butler, and L. Whetsel, "An Analysis of Power Reduction Techniques in Scan Testing," *Proc. Intl. Test Conf*., pp. 670-677, 2001.

[Sinanoglu 03] Sinanoglu, O. and A. Orailoglu, "Modeling Scan Chain Modifications for Scan-in Test Power Minimization," *Proc. Intl. Test Conf*., pp. 602-611, 2003.

[Synopsys] Synopsys, Inc., http://www.synopsys.com.

[Wang 06] Wang, L.-T., C.-W. Wu, and X. Wen (Eds.), *VLSI Test Principles and Architectures*, Morgan Kaufmann Publishers, San Francisco, CA, 2006.

[Williams 73] Williams, M. J. Y. and J. B. Angell, "Enhancing Testability of Large-Scale Integrated Circuits via Test Points and Additional Logic," *IEEE Trans. on Computers*, vol. C-22, no. 1, pp. 46-60, Jan. 1973.

[Yoshida 03] Yoshida, T. and M. Watai, "A New Approach for Low Power Scan Testing," *Proc. Intl. Test Conf.*, pp. 480-487, 2003.