Gate Exhaustive Testing

Kyoung Youn Cho[1], Subhasish Mitra[1,2], and Edward J. McCluskey[1]

[1] Center for Reliable Computing
Departments of Electrical Engineering and Computer Science
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 88] 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

0-7803-9039-3/$20.00 © 2005 IEEE

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.

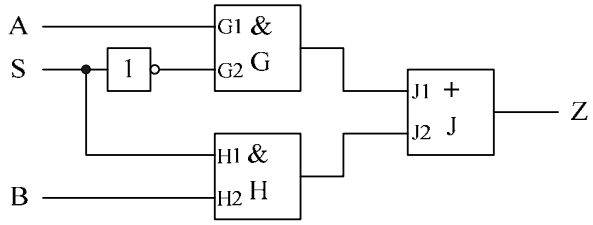
Definition 3: A *nonobservable input combination* of a gate is an input combination that cannot be applied to the gate inputs with the gate output being sensitized to at least one observation point.

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.

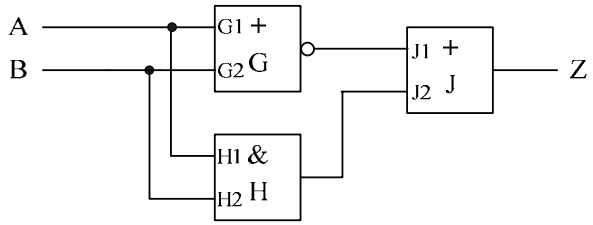
In general, we may not be able to apply all possible input combinations to an internal gate and observe its response at some observation points [McCluskey 93]. The circuits in Fig. 1 are used to show examples of nonobservable gate input combinations.

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

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

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

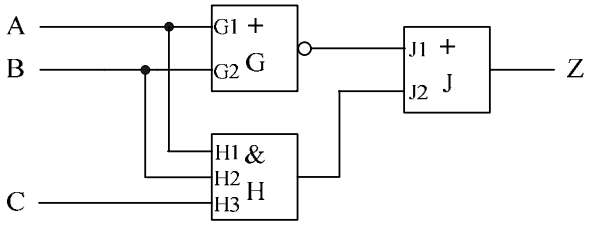


Figure 2 Example circuit to illustrate the gate exhaustive testing and the GEC

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).

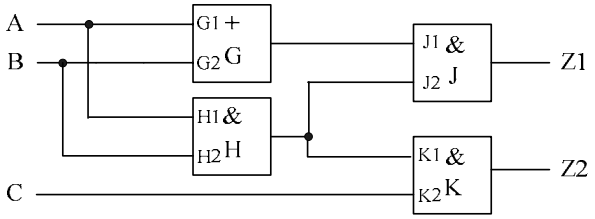


Figure 3 Example circuit to illustrate gate exhaustive testing

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 as 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.

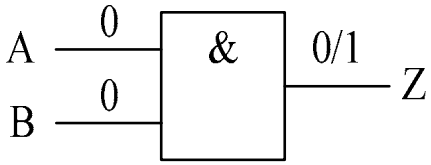


Figure 4 Example circuit to explain a pattern fault

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.

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 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

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.

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

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 [α]	3	506	
Gate exhaustive	5,655	99.6	-	98.3	0	6,843	

All tests are generated using complex gate level netlist.
[α] 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[α]	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[λ]	679	
10-detect	3,554	100.0	-	98.0	1[λ]	1,232	
15-detect	5,290	100.0	-	98.2	2[λ]	1,811	
Transition	792	-	99.7	96.5[β]	0	742	
XOR & MUX exhaustive	856	100.0	-	94.8	3	2,799	
Gate exhaustive	1,528	100.0	-	99.0	2[λ]	3,766	

[α] Generated using elementary gate level netlist. Other tests are generated using complex gate level netlist.
[β] GEC is calculated for $V_1 \cup V_2$
[λ] Classified as chips with sequence dependent defects

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

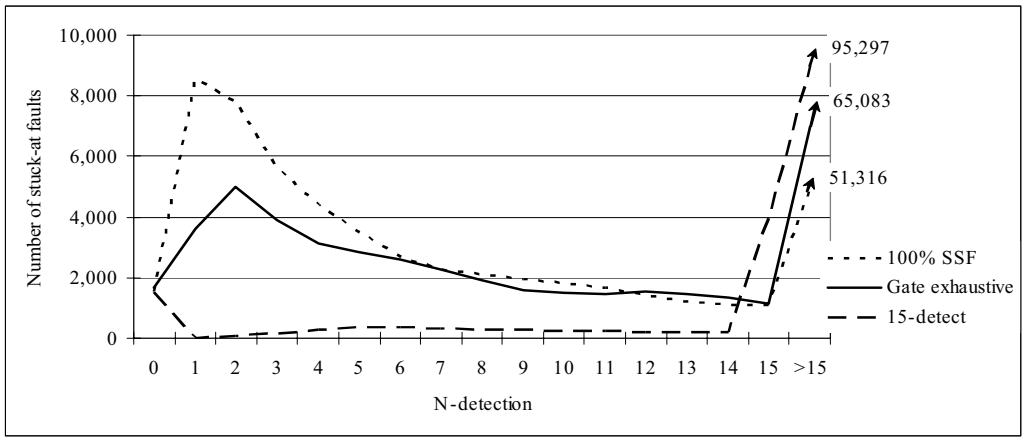


Figure 5 ***N*-detect stuck-at fault simulation results: ELF35**

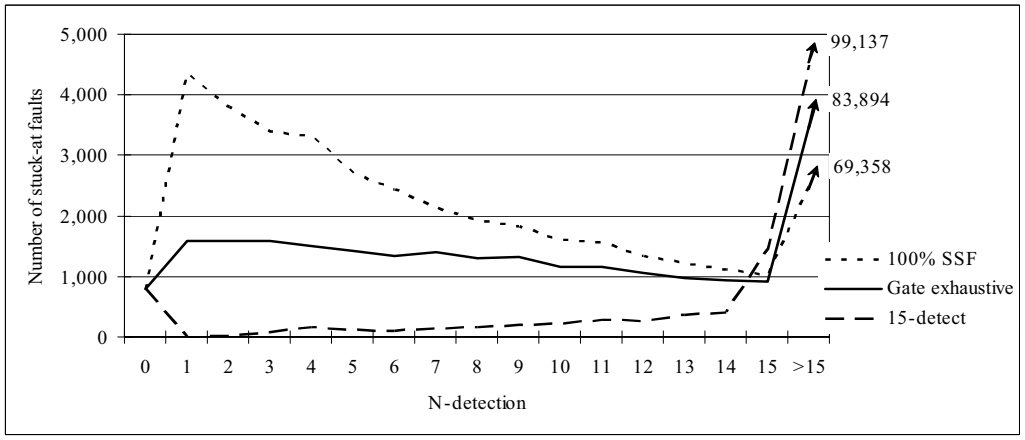


Figure 6 ***N*-detect stuck-at fault simulation results: ELF18**

fortuitous detection by adding more patterns that detect stuck-at faults multiple times.

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

[Blanton 03] R. D. Blanton, 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] J. Dworak, 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] James C.-M. Li and E. J. McCluskey, "Diagnosis of Sequence-dependent Chips," *Proc. VLSI Test Symp*., pp. 187-192, 2002.

[Ma 95] S. C. Ma, 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] E. J. McCluskey, "Verification Testing – A Pseudoexhaustive Test Technique," *IEEE Trans. Computers*, vol. C-33, no. 6, pp. 541-546, Jun. 1984

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

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

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

[Mitra 04] S. Mitra, 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] I. Pomeranz and S. M. Reddy, "On Test Data Compression and n-Detection Test Sets," *Design Autom. Conf*., pp. 748-751, 2003.

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

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

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

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

[Tseng 01b] Chao-Wen Tseng, 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] S. Venkataraman, 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 1988] J. A. Waicukauski, E. Lindbloom, B. K. Rosen, and V. S. Iyengar, "Transition Fault Simulation," *IEEE Design & Test Comput*., vol. 4, pp. 32-38, Jan. 1988.