Center for Reliable Computing

TECHNICAL REPORT

Test Point Insertion for Non-Feedback Bridging Faults

Nur A. Touba and Edward J. McCluskey

96-3 (CSL TR # 96-703) August 1996	**Center for Reliable Computing** Gates Building 2A, Room 236 Computer Systems Laboratory Departments of Electrical Engineering and Computer Science Stanford University Stanford, California 94305

Abstract:

This paper studies pseudo-random pattern testing of bridging faults. Although bridging faults are generally more random pattern testable than stuck-at faults, examples are shown to illustrate that some bridging faults can be much less random pattern testable than stuck-at faults. A fast method for identifying these random-pattern-resistant bridging faults is described. It is shown that state-of-the-art test point insertion techniques, which are based on the stuck-at fault model, are inadequate. Data is presented which indicates that even after inserting test points that result in 100% single stuck-at fault coverage, many bridging faults are still not detected. A test point insertion procedure that targets both single stuck-at faults and non-feedback bridging faults is presented. It is shown that by considering both types of faults when selecting the location for test points, higher fault coverage can be obtained with little or no increase in overhead. Thus, the test point insertion procedure described here is a low-cost way to improve the quality of built-in self-test. While this paper considers only non-feedback bridging faults, the techniques that are described can be applied to feedback bridging faults in a straightforward manner.

Funding:

This work was supported in part by the Ballistic Missile Defense Organization, Innovative Science and Technology (BMDO/IST) Directorate and administered through the Department of the Navy, Office of Naval Research under Grant No. N00014-92-J-1782, by the National Science Foundation under Grant No. MIP-9107760, and by the Advanced Research Projects Agency under prime contract No. DABT63-94-C-0045.

Imprimatur: Vincent Lo and Sanjay Wattal

Test Point Insertion for Non-Feedback Bridging Faults

Nur A. Touba and Edward J. McCluskey

CENTER FOR RELIABLE COMPUTING
Computer Systems Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, California 94305-4055

ABSTRACT

This paper studies pseudo-random pattern testing of bridging faults. Although bridging faults are generally more random pattern testable than stuck-at faults, examples are shown to illustrate that some bridging faults can be much less random pattern testable than stuck-at faults. A fast method for identifying these random-pattern-resistant bridging faults is described. It is shown that state-of-the-art test point insertion techniques, which are based on the stuck-at fault model, are inadequate. Data is presented which indicates that even after inserting test points that result in 100% single stuck-at fault coverage, many bridging faults are still not detected. A test point insertion procedure that targets both single stuck-at faults and non-feedback bridging faults is presented. It is shown that by considering both types of faults when selecting the location for test points, higher fault coverage can be obtained with little or no increase in overhead. Thus, the test point insertion procedure described here is a low-cost way to improve the quality of built-in self-test. While this paper considers only non-feedback bridging faults, the techniques that are described can be applied to feedback bridging faults in a straightforward manner.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

A common physical defect in MOS technologies is a short between two signal lines which results in a bridging fault [Shen 85], [Ferguson 88]. Detecting bridging faults during the test process is very important for achieving high quality levels. Bridging faults can be detected with either IDDQ testing [Levi 81], [Acken 83], or conventional voltage testing.

IDDQ testing involves monitoring the quiescent power supply current in CMOS circuits. If two shorted nodes are driven to opposite values, an increase in the static current results. If this increase in the static current can be measured, then the corresponding bridging fault can be detected. There are two well-known drawbacks to IDDQ testing. The first is that current measurements take longer than voltage measurements. The second is that subthreshold ("leakage") current can mask the effect of a bridging fault. As feature sizes continue to shrink, leakage current will increase making it increasingly difficult to differentiate good and defective devices using IDDQ measurements [Williams 96]. Projected data presented by Williams, *et. al*, in [Williams 96] is not encouraging for the future quality of IDDQ testing. This paper focuses on conventional voltage testing for bridging faults.

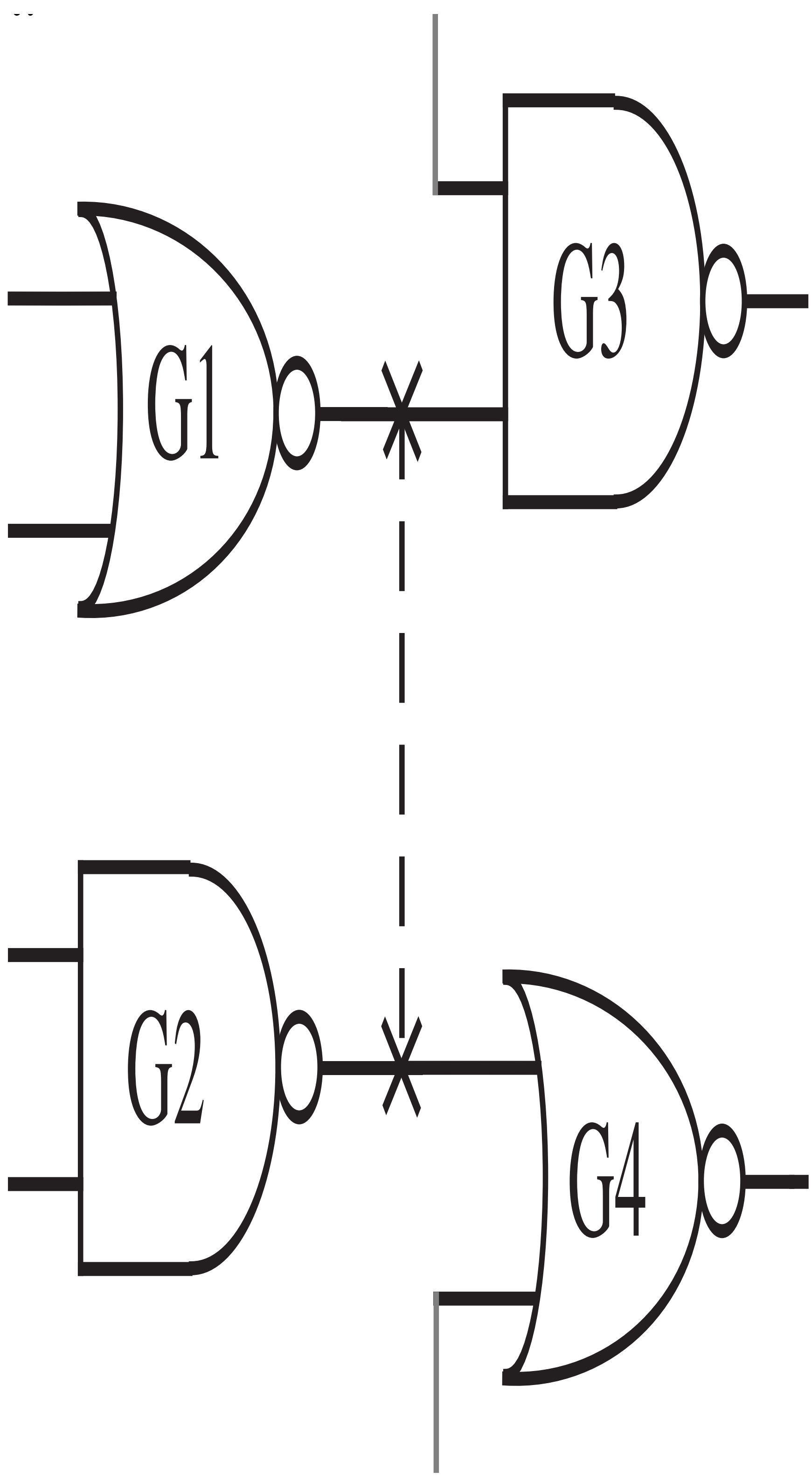
While test sets for single stuck-at faults guarantee detection of some bridging faults (e.g., bridging faults between the inputs of an elementary gate [Mei 74]), they do not guarantee detection of the vast majority of bridging faults. It has been shown that a significant number of bridging faults are generally not detected by single stuck-at tests sets [Millman 88], [Storey 90], [Butler 92], [Chess 94]. Empirical data confirms the limits of single stuck-at testing [Pancholy 90], [Maxwell 91], [Storey 91], [Perry 92], [Gayle 93], [Ma 95]. In order to achieve the quality levels now required for digital integrated circuits, research has been done on deterministic test pattern generation and fault simulation techniques that explicitly target bridging faults [Abramovici 85], [Acken 91], [Millman 91], [Lee 91], [Ferguson 91], [Hajj 92], [Greenstein 92], [Chess 93, 94], [Rearick 93], [Maxwell 93].

While previous research has focused on deterministic testing of bridging faults, this paper studies pseudo-random testing of bridging faults. Pseudo-random testing is an attractive approach because of its suitability for built-in self-test (BIST). A simple compact circuit such as a linear feedback shift register (LFSR) or cellular automaton (CA) can be used to generate the patterns thereby minimizing BIST overhead. Although bridging faults are generally more random pattern testable than stuck-at faults, examples are shown to illustrate that some bridging faults can be much less random pattern testable than stuck-at faults. A fast method for identifying these random-pattern-resistant bridging faults is described. It is shown that state-of-the-art test point insertion techniques, which are based on the stuck-at fault model, are inadequate. Data is presented which indicates that even after inserting test points that result in 100% single stuck-at fault coverage, many bridging faults are still not detected. A test point insertion procedure that targets both single stuck-at faults and non-feedback bridging faults is presented. It is shown that by considering both types of faults when selecting the location for test points, higher fault coverage can be obtained with little or no increase in overhead.

The paper is organized as follows: In Sec. 2, the bridging fault model that is used in this paper is explained. In Sec. 3, a fast method for identifying random-pattern-resistant bridging faults is described. In Sec. 4, a test point insertion procedure which targets both single stuck-at and non-feedback bridging faults is presented. In Sec. 5, experimental results are shown for the test point insertion procedure. Section 6 is a conclusion.

2. BRIDGING FAULT MODELS

Three gate level bridging fault models are wired-and, wired-or, and dominant driver. These models are illustrated in Figure 1. In the wired-and (wired-or) model, if the output of either gate *G1* or gate *G2* is a 0 (1), then the shorted node is a 0 (1). This situation occurs in CMOS when the n-network pull-down (p-network pull-up) is stronger than the p-network pull-up (n-network pull-down). In the dominant driver model, it is assumed that the output of gate *G1* (gate *G2*) is stronger than the output of gate *G2* (gate *G1*), and hence the shorted output is always equal to the output of gate *G1* (gate *G2*). This situation occurs in CMOS when gate *G1* (gate *G2*) is scaled such that it has a larger load driving capability than gate *G2* (gate *G1*).


G1
G3
G2
G4

G1 Output	G2 Output	Wired-and	Wired-or	G1 Dominant	G2 Dominant
0	0	0	0	0	0
0	1	0	1	0	1
1	0	0	1	1	0
1	1	1	1	1	1

Figure 1. Example of Gate Level Bridging Fault Models

If the layout of the circuit is known, inductive fault analysis techniques can be used to identify a set of probable bridging faults [Ferguson 88], [Jee 93]. Circuit level models can then be used to more accurately predict the behavior of each bridging fault. Several different methods have been proposed with various tradeoffs between accuracy and simulation time [Acken 88, 91, 92], [Lee 91], [Hajj 92], [Greenstein 92], [Maxwell 93], [Rearick 93]. The drawback of using layout dependent fault modeling is that if the layout changes, then the results are no longer valid.

Since test point insertion involves modifying the circuit and hence changing the layout, layout dependent fault modeling is not feasible. For this reason, gate level bridging faults models are used in this paper. All three of the gate level bridging fault models previously described are targeted to ensure a very thorough test that is layout independent.

Bridging faults can be divided into two classes. *Feedback bridging faults* are those in which there is a path in the fault-free circuit from one of the shorted lines to the other thereby creating feedback in the faulty circuit. *Non-feedback bridging faults* are those for which no feedback is introduced when the two lines are shorted together. Feedback bridging faults may add state causing the circuit to no longer be combinational, and thus they are more complicated to simulate. Since feedback bridging faults have been found to be easier to detect than non-feedback bridging faults [Millman 88], this paper will consider only non-feedback bridging faults. However, the techniques described in this paper can be applied to feedback bridging faults in a straightforward manner. The only difference is the added complexity for simulation.

3. RANDOM-PATTERN-RESISTANT BRIDGING FAULTS

Detection of the gate level bridging faults described in the previous section can be related to single stuck-at fault detection using the theorems shown below [Williams 73]. Note that in these theorems, "node" refers to primary inputs and gate outputs. Stems and fanout branches are not distinguished because a bridging fault will never cause a stem and its branches to have different values. Stuck-at 1 and stuck-at 0 are abbreviated *s-a-1* and *s-a-0*, respectively.

Theorem 1: A test t detects a wired-AND non-feedback bridging fault between node x and node y if and only if either t detects x s-a-0 and sets $y = 0$, or t detects y s-a-0 and sets $x = 0$.

Theorem 2: A test t detects a wired-OR non-feedback bridging fault between node x and node y if and only if either t detects x s-a-1 and sets $y = 1$, or t detects y s-a-1 and sets $x = 1$.

Theorem 3: A test t detects a node x dominant non-feedback bridging fault between node x and node y if and only if either t detects y s-a-0 and sets $x = 0$, or t detects y s-a-1 and sets $x = 1$.

The *detection probability* of a fault is equal to the number of input patterns that detect the fault divided by the total number of inputs patterns, 2^n, where n is the number of primary inputs. Faults with very low detection probabilities are said to be *random-pattern-resistant (r.p.r.)* because they are hard to detect with random patterns [Eichelberger 83]. The detection probability for bridging faults is generally higher than that for stuck-at faults because there are two possible sites from which the effects of the fault can be observed, whereas there is only one site from which the effects of a single stuck-at fault can be observed. However, examples will be shown to illustrate that the detection probability for some bridging faults can be much lower than that for any single stuck-at faults.

3.1 Examples of Random-Pattern-Resistant Bridging Faults

Figure 2 shows an example of a bridging fault whose detection probability is much lower than that for any single stuck-at fault in the circuit. All of the stuck-at faults in the circuit have a detection probability of 2^{-6} or greater, whereas the wired-and bridging fault has a detection probability of 2^{-10}. The reason for this is that ANDing the two inputs of gate *G2* will never change the output of gate *G2*, so the only way to observe the wired-and bridging fault is through the fanout line to gate *G3*. This type of situation occurs anytime a bridging fault between two inputs lines to a gate mimics the logic function of the gate. Such a bridging fault can only be observed through a fanout line from one of the gate inputs and therefore can have a low detection probability. Note that it is likely that two input lines to a gate will be routed near each other and thus an unintentional short between them is quite possible.

Figure 3 shows another example of a bridging fault whose detection probability is much lower than that for any single stuck-at faults in the circuit. Again, all of the stuck-at faults have a detection probability of 2^{-6} or greater, whereas the dominant driver bridging fault has a detection probability of 2^{-10}. The reason for this is that the logic values of the two lines involved in the bridging fault are correlated. If the common input to gate G2 and gate G3 is a 0, then the fault-free output of both gates is a 1. The only time that the bridging fault is provoked is when the common input to gate G2 and gate G3 is a 1 and the output of gate G1 is a 0, then the output of gate G2 is a 1 which forces a faulty value on the output of gate G3. The type of situation can occur when the two lines involved in a bridging fault share inputs. Note again that it is likely that lines that share inputs will be routed near each other and thus an unintentional short between them is quite possible.

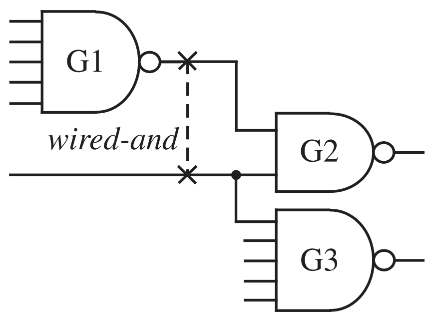


Figure 2. Example of a Random-Pattern-Resistant Bridging Fault Between Input Lines of a Gate

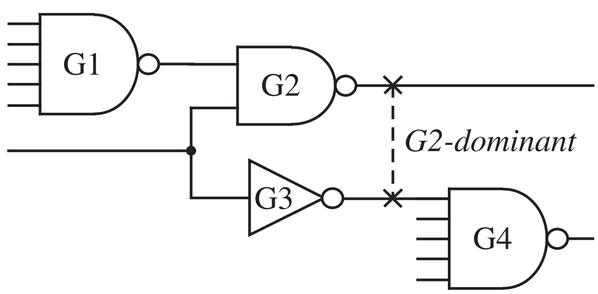


Figure 3. Example of a Random-Pattern-Resistant Bridging Fault Between Correlated Lines

3.2 Identifying Random-Pattern-Resistant Bridging Faults

There are two general approaches for identifying which bridging faults are random-pattern-resistant. The first approach is to analytically compute the detection probabilities. Computing fault detection probabilities is an NP-hard problem [Krishnamurthy 86]. For small circuits, an exact method for computing detection probabilities for bridging faults was outlined in [Kapur 91]. For larger circuits, many methods exist for estimating detection probabilities for single stuck-at faults, but the accuracy of applying these techniques to computing detection probabilities for bridging faults can be greatly reduced when the controllability of the two shorted nodes is not independent.

The second approach for identifying random-pattern-resistant bridging faults is to perform simulation experiments. This involves doing fault simulation for several different random pattern test sets and keeping statistics on which bridging faults are not detected.

In the case of pseudo-random pattern testing where the patterns that are going to be used during testing are known a priori, fault simulation can be done to find the exact set of bridging faults that are not detected. Given this set of undetected bridging faults, a test point insertion procedure will be presented in the next section for modifying the circuit so that all of the bridging faults are detected.

One issue is the amount of time that is required for fault simulation of the bridging faults. One advantage of using gate level bridging fault models is that a stuck-at fault simulator can be used for fault simulation of bridging faults as described by Abramovici and Menon [Abramovici 85]. This is done by using the theorems listed above to relate bridging fault detection to stuck-at fault detection. Each time a stuck-at fault is detected, the theorems above are used to determine which bridging faults to mark as detected. Normally a stuck-at fault is dropped from the fault list as soon as it is detected, but when considering bridging faults, a stuck-at fault is not dropped from the fault list until all of the bridging faults associated with it have been detected. As a result, there is an increase in the fault simulation time for bridging faults compared with stuck-at faults due to the fact that the fault list is not reduced as quickly. One speedup technique that was suggested by Chess and Larrabee [Chess 93] is to check if any of the bridging faults associated with a stuck-at fault are provoked by a pattern before simulating the stuck-at fault for that pattern; this reduces the number of stuck-at faults that need to be simulated for each pattern.

Note that the number of bridging faults associated with each single stuck-at fault is proportional to the size of the circuit. So the larger the circuit, the slower the fault list will be reduced during fault simulation. One way to reduce the fault simulation time is to only consider bridging faults between nodes with low stuck-at detection probabilities. While this will not guarantee that all random-pattern-resistant bridging faults are found, data in Table 1 suggests that it will find most of them.

In Table 1, results are shown for fault simulation of non-feedback bridging faults for 32,000 pseudo-random patterns. The first two columns are for simulating all non-feedback bridging faults, and the last three columns are for only simulating non-feedback bridging faults between nodes where single stuck-at faults were detected less than 5 times by the 32,000 patterns. The fault simulation times are expressed as a multiple of the time required for single stuck-at fault simulation. The number of undetected non-feedback bridging faults that were found in each case is shown.

Table 1. Results for Fault Simulation of Bridging Faults

Circuit Name	Simulate Faults Between All Nodes		Simulate Faults Between Nodes Detected ≤ 5 Times		
	Undetected Bridging Total	Multiple of SSA Simulation Time	Undetected Bridging Found	Undetected Bridging Missed	Multiple of SSA Simulation Time
s420	256	9	256	0	5
s838	1527	5	1527	0	4
s1196	29	12	29	0	5
s1423.s	5	3	3	0	2
C2670.s	2197	14	2190	7	7
C3540.s	211	19	186	25	9
C5314.s	17	7	16	1	4
C7552.s	3487	15	3470	17	8

4. TEST POINT INSERTION FOR BRIDGING FAULTS

Test point insertion involves adding control and observation points to the circuit-under-test in a way that the system function remains the same, but the testability is improved. An *observation point* is an additional primary output that is inserted in the circuit to increase the observability of nodes in the circuit. In the example in Fig. 4, an observation point is inserted at the output of gate *G1* such that nodes are observable regardless of the logic value at node *y*. A *control point* is inserted in the circuit such that when it is activated, it fixes the logic value at a particular node to increase the controllability of some nodes in the circuit. In the example in Fig. 5, a control point is inserted to fix the logic value at the output of gate *G1* to a ‘1’ when the control point is activated. This is accomplished by placing an OR gate at the output of gate *G1*. During system operation, the control points are not activated and thus don't affect the system function. However, control points do add an extra level of logic to some paths in the circuit which can increase the delay through the circuit.

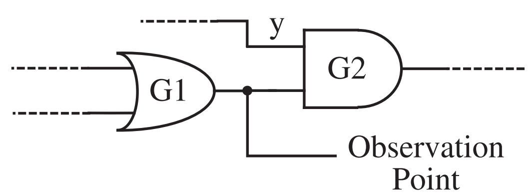


Figure 4. Example of Observation Point

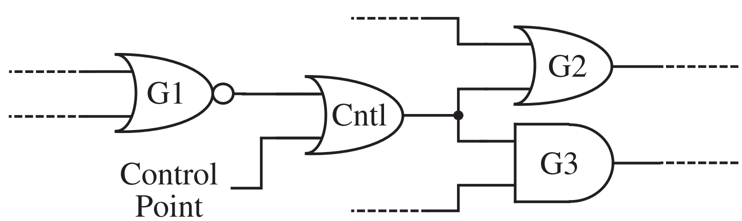


Figure 5. Example of Control-1 Point

Since test points add both area and performance overhead, it is important to try to minimize the number of test points that are inserted to achieve the desired fault coverage. This is accomplished by carefully selecting the location of each test point. There are two general approaches for test point placement. One approach is to select the location of the test points based on testability measures [Seiss 91], [Savaria 91], [Youssef 93], [Cheng 95]. The other approach is to select the location of the test points based on data collected during simulation [Briers 86], [Iyengar 89], [Touba 96]. All of these techniques target single stuck-at faults only. The focus of this paper is to target bridging faults. Because of the added complexity in controlling and added flexibility in observing bridging faults, the effectiveness of testability measures in predicting bridging fault testability is questionable. In this section, a simulation based test point insertion technique will be described for targeting both single stuck-at and bridging faults.

The test point insertion technique described here uses the path tracing method introduced in [Touba 96]. For each undetected stuck-at fault and bridging fault, a path tracing procedure is used to identify the set of test points that will enable the fault to be detected, i.e., the set of *test point solutions* for the fault. Given the set of test points solutions for each undetected fault, a minimal set of test points to achieve the desired fault coverage is selected using a set covering procedure.

4.1 Computing Observation Point Solutions

Given the set of pseudo-random patterns that are applied to the circuit during testing, the set of observation point solutions for each undetected bridging fault can be computed. Fault-free simulation is performed for each pseudo-random pattern, and a check is made to see if the pattern places opposite values on the two shorted nodes of an undetected bridging fault thereby provoking the fault. If the fault is provoked, then path tracing is performed to identify the set of nodes that the effect of the fault propagates to. An observation point placed at any of the nodes that the fault propagates to will enable the fault to be detected and thus is a solution for the fault.

An example is shown in Fig. 6. Fault-free simulation is performed for a pattern that provokes the wired-or bridging fault, and path tracing is used to identify the propagation path for the fault. The fault propagates through gates *G3* and *G5*, but is blocked at gates *G6* and *G8* and therefore doesn't propagate to a primary output. Inserting an observation point at node *a* or node *b* would enable the fault to be detected, so those two nodes form the set of observation point solutions for the fault for that pattern. The union of the set of observation point solutions for each pseudo-random pattern that provokes a particular fault gives the full set of observation point solutions for the fault.

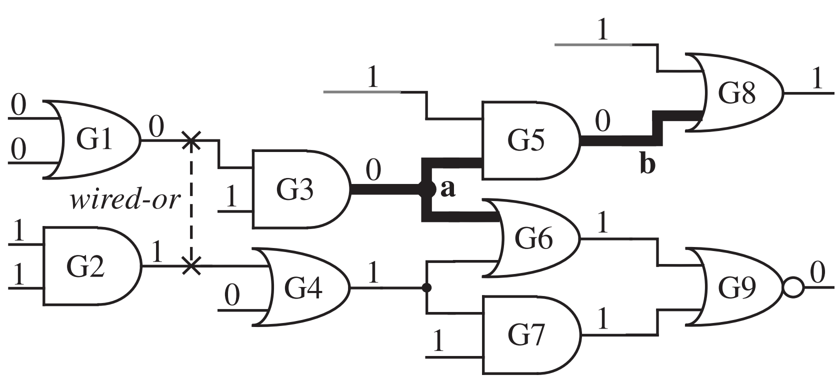


Figure 6. Example: Observation Point at Node *a* or Node *b* is a Solution.

4.2 Computing Control Point Solutions

For computing control point solutions, fault-free simulation is performed for each pseudo-random pattern, and a check is made to see if a sensitized path exists from an undetected bridging fault site to a primary output. If so, then backwards path tracing is performed to identify the set of nodes *S* that have a sensitized path to the line of the bridging fault whose value needs to be complemented in order to provoke the bridging fault in the appropriate manner. A control point that complements the value at any of the nodes in *S* is a solution for the fault provided that it doesn't block propagation of the fault to a primary output.

An example is shown in Fig. 7. Fault-free simulation is performed for a pattern that sensitizes an undetected bridging fault at the output of gate *G6* to a primary output. In order to provoke the wired-and bridging fault so that it causes a faulty value at the output of gate *G6*, the output of gate *G6* needs to be complemented. Backward path tracing from the output of gate *G6* is used to identify sensitized paths. Both inputs of gate *G6* have a sensitized path to the output of gate *G6*. Neither of the inputs of gate *G4* have a sensitized path to the output of gate *G4*. One of the inputs of gate *G3* has a sensitized path to the output of gate *G3*. Inserting a control-1 point at node *a*, *c*, *d*, or *e* would complement the value at the output of gate G6 thereby provoking the fault. However, forward path tracing from node *e* identifies that it has a sensitized path to gate *G9*, so inserting a control-1 point at node *e* would block the effect of the fault from propagating to a primary output. Therefore, only control-1 points at nodes *a*, *c*, and *d* are solutions. The union of the set of control point solutions for a particular fault for each pseudo-random pattern gives the full set of control point solutions for the fault.

A fast approximate procedure for path tracing is given in [Abramovici 84] and an exact method is given in [Menon 91]. These papers describe path tracing from the primary outputs (called *critical path tracing*), however the procedures can be easily generalized for path tracing from a fault site.

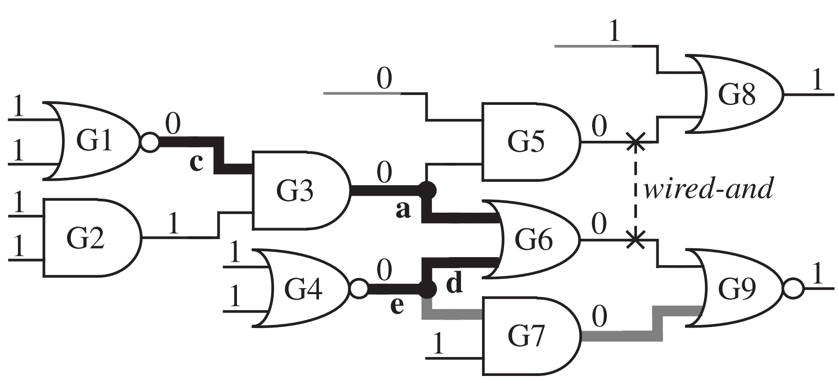


Figure 7. Example: Control-1 Point at Node *a*, Node *c*, or Node *d* is a Solution, but Node *e* is Not a Solution Because it Blocks Propagation to a Primary Output.

4.3 Selecting a Set of Test Points to Insert

Once the set of test point solutions for each undetected single-stuck at fault and bridging fault has been computed, a set covering procedure can be used to select a minimal set of test points that will enable all of the faults to be detected. A matrix is constructed in which each column corresponds to a test point solution. For each undetected fault, a row is added to the matrix in which an 'X' is placed in each column that corresponds to a test point solution for the fault. An example is shown in Fig. 8. The first row corresponds to *fault 1* for which the set of single test point solutions is an observation point at node *w*, a control-1 point at node *u*, and a control-0 point at node *v*.

	O-v	O-w	O-x	C1-u	C0-v	C0-w	C1-y	C1-z
Fault 1		X		X	X			
Fault 2				X	X		X	
Fault 3			X		X			X
Fault 4	X		X		X			
Fault 5				X		X		

Figure 8. Example: Matrix of Test Point Solutions for Each Fault

A set covering procedure [Christofedes 75] is used to select a minimal set of columns that has at least one 'X' in each row. One 'X' in each row ensures that all of the faults will be detected. In the example in Fig. 8, one such solution is the third column (observation point at node *x*) and the fourth column (control-1 point at node *u*). The test points corresponding to the selected columns are inserted into the circuit. Once the test points have been inserted, a procedure is given in [Touba 96] for synthesizing logic to drive the control points.

5. EXPERIMENTAL RESULTS

The procedure described in this paper was used to insert test points in some of the ISCAS 85 [Brglez 85] and ISCAS 89 [Brglez 89] benchmark circuits that contain random-pattern-resistant bridging faults. LFSR's were used to apply 32,000 pseudo-random test patterns to each circuit. It was assumed that the flip-flops in the ISCAS 89 circuits were configured as part of the LFSR during testing so that the circuits are tested like combinational circuits. The number of stages in the LFSR for each circuit was equal to the number of primary inputs plus the number of flip-flops.

Test points were inserted into each circuit so that all single stuck-at faults and all detectable wired-and, wired-or, and dominant driver non-feedback bridging faults were detected by the set of 32,000 pseudo-random test patterns. The results are shown in Table 2. A ".s" posta circuit indicates that it was simplified by removing redundant logic. The number of undetected single stuck-at faults and non-feedback bridging faults before test point insertion and after test point insertion is shown. Two test point insertion procedures were used. The first targets single stuck-at faults only. The second targets both single stuck-at faults and non-feedback bridging faults. The number of control points (*Num Con*) and the number of observation points (*Num Obs*) that were inserted by each procedure is shown.

The results indicate that circuits that are random pattern testable for single stuck-at faults are not necessarily random pattern testable for bridging faults. Current test point insertion procedures which consider only stuck-at faults may leave many bridging faults undetected. By considering both stuck-at faults and bridging faults, the test insertion procedure described in this paper enables a higher quality test with little or no increase in overhead. In many cases, only one additional observation point is sufficient. For circuit *s838*, the procedure selected a different location for the control point such that no additional overhead was required.

Table 2. Results for Test Point Insertion in Benchmark Circuits

Circuit Name	No Test Points		TPI Targeting Stuck-At Only				TPI Targeting Stuck-At & Bridging			
	Undetected Stuck-At	Undetected Bridging	Num Con	Num Obs	Undetected Stuck-At	Undetected Bridging	Num Con	Num Obs	Undetected Stuck-At	Undetected Bridging
s420	21	256	2	0	0	11	2	1	0	0
s838	44	1527	2	0	0	19	2	0	0	0
s1196	3	29	1	0	0	6	1	1	0	0
s1423.s	0	5	0	0	0	5	0	1	0	0
C2670.s	164	2197	2	2	0	312	2	4	0	0
C3540.s	0	211	0	0	0	211	2	3	0	0
C5314.s	0	17	0	0	0	17	0	1	0	0
C7552.s	130	3487	6	8	0	142	7	9	0	0

6. CONCLUSIONS

This paper presented a low-cost technique for improving the quality of pseudo-random pattern testing. A procedure was described for targeting both single stuck-at faults and non-feedback bridging faults during test point insertion. By considering both types of faults during test point insertion, the location of the test points can be chosen in a way that provides higher fault coverage with little or no additional overhead.

While this paper considered only non-feedback bridging faults, the technique that was described can be applied to feedback bridging faults in a straightforward manner. The only difference is the added complexity for simulation.

ACKNOWLEDGMENTS

This work was supported in part by the Ballistic Missile Defense Organization, Innovative Science and Technology (BMDO/IST) Directorate and administered through the Department of the Navy, Office of Naval Research under Grant No. N00014-92-J-1782, by the National Science Foundation under Grant No. MIP-9107760, and by the Advanced Research Projects Agency under prime contract No. DABT63-94-C-0045.

REFERENCES

[Abramovici 84] Abramovici, M., P.R. Menon, and D.T. Miller, "Critical Path Tracing: An Alternative to Fault Simulation," *IEEE Design & Test of Computers*, Vol. 1, pp. 89-93, Feb. 1984.

[Abramovici 85] Abramovici, M., and P.R. Menon, "A Practical Approach to Fault Simulation and Test Generation for Bridging Faults," *IEEE Transactions on Computers* , Vol. C-34, No. 7, pp. 658-663, Jul. 1985.

[Acken 83] Acken, J.M., "Testing for Bridging Faults (Shorts) in CMOS Circuits," *Proc. of the 20th Design Automation Conference*, pp. 717-718, 1983.

[Acken 88] Acken, J.M., *Deriving Accurate Fault Models*, Ph.D. Thesis, CSL-TR-88-365, Computer Systems Laboratory, Stanford University, Oct. 1988.

[Acken 91] Acken, J.M., and S.D. Millman, "Accurate Modeling and Simulation of Bridging Faults," *Proc. of 1991 Custom Integrated Circuits Conference*, pp. 17.4.1-17.4.4, 1991.

[Acken 92] Acken, J.M., and S.D. Millman, "Fault Model Evolution for Diagnosis: Accuracy vs. Precision," *Proc. of 1992 Custom Integrated Circuits Conference*, pp. 13.4.1-13.4.4, 1992.

[Briers 86] Briers, A.J., and K.A.E. Totton, "Random Pattern Testability by Fast Fault Simulation," *Proc. of International Test Conference*, pp. 274-281, 1986.

[Brglez 85] Brglez, F., and H. Fujiwara, "A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translator in Fortan," *Proc. of International Symposium on Circuits and Systems*, pp. 663-698, 1985.

[Brglez 89] Brglez, F., D. Bryan, and K. Kozminski, "Combinational Profiles of Sequential Benchmark Circuits," *Proc. of International Symposium on Circuits and Systems*, pp. 1929-1934, 1989.

[Butler 92] Butler, K.M., and M.R. Mercer, *Assessing Fault Model and Test Quality*, Boston: Kluwer Academic Publishers, 1992.

[Cheng 95] Cheng, K.-T., and C.J. Lin, "Timing-Driven Test Point Insertion for Full-Scan and Paritial-Scan BIST," *Proc. of International Test Conference*, pp. 506-514, 1995.

[Chess 93] Chess B., and T. Larrabee, "Bridge Fault Simulation Strategies for CMOS Integrated Circuits," *Proc. of the 30h Design Automation Conference*, pp. 458-462, 1993.

[Chess 94] Chess B., A. Freitas, F.J. Ferguson, T. Larrabee, "Testing CMOS Logic Gates for Realistic Shorts," *Proc. of International Test Conference*, pp. 395-402, 1994.

[Christofedes 75] Christofedes, N., and K. Korman, "A Computational Survey of Methods for the Set Covering Problem," *Management Science*, Vol. 21, No. 5, pp. 591-599, Jan. 1975.

[Eichelberger 83] Eichelberger, E.B., and E. Lindbloom, “Random-Pattern Coverage Enhancement and Diagnosis for LSSD Logic Self-Test,” *IBM Journal of Research and Development*, Vol. 27, No. 3, pp. 265-272, May 1983.

[Ferguson 88] Ferguson, F.J., and J.P. Shen, “A CMOS Fault-Extractor for Inductive Fault Analysis,” *IEEE Transactions on Computer-Aided Design*, Vol. 7, No. 11, pp. 1181-1194, Nov. 1988.

[Ferguson 91] Ferguson, F.J., and T. Larrabee, “Test Pattern Generation for Realistic Bridge Faults in CMOS ICs,” *Proc. of International Test Conference*, pp. 492-499, 1991.

[Gayle 93] Gayle, R., “The Cost of Quality: Reducing ASIC Defects with IDDQ, At-Speed Testing, and Increased Fault Coverage,” *Proc. of International Test Conference*, pp. 285-292, 1993.

[Greenstein 92] Greenstein, G.S., and J.H. Patel, “E-PROOFS: A CMOS Bridging Fault Simulator,” *Proc. of International Conference on Computer-Aided Design (ICCAD)*, pp. 268-271, 1992.

[Hajj 92] Hajj, I., and T. Lee, “Simulation of Physical Faults in VLSI Circuits,” *Proc. of VLSI Test Symposium*, pp. 202-207, 1992.

[Iyengar 89] Iyengar, V.S., and D. Brand, “Synthesis of Pseudo-Random Pattern Testable Designs,” *Proc. International Test Conference*, pp. 501-508, 1989.

[Jee 93] Jee, A., and F.J. Ferguson, “Carafe: An Inductive Fault Analysis Tool for CMOS VLSI Circuits,” *Proc. of VLSI Test Symposium*, pp. 92-98, 1993.

[Kapur 91] Kapur, R., K.M. Butler, D.E. Ross, and M.R. Mercer, “On Bridging Fault Controllability and Observability and Their Correlations to Detectability,” *Proc. of European Test Conference*, pp. 333-339, 1991.

[Krishnamurthy 86] Krishnamurthy, B., and I. Tollis, “Improved Techniques for Estimating Signal Probabilities,” *Proc. of International Test Conference*, pp. 244-251, 1986.

[Lee 91] Lee, T., and I. Hajj, “A Switch-Level Matrix Approach to Transitor-Level Fault Simulation,” *Proc. of International Conference on Computer-Aided Design (ICCAD)*, pp. 554-557, 1991.

[Levi 81] Levi, M.W., “CMOS is Most Testable,” *Proc. International Test Conference*, pp. 217-220, 1981.

[Maxwell 91] Maxwell, P.C., R.C. Aitken, V. Johansen, and I. Chiang, “The Effect of Different Test Sets on Quality Level Prediction: When is 80% Better Than 90%,” *Proc. of International Test Conference*, pp. 358-364, 1991.

[Maxwell 93] Maxwell, P.C., and R.C. Aitken, “Biased Voting: A Method for Simulating CMOS Bridging Faults in the Presence of Variable Gate Logic Thresholds,” *Proc. of International Test Conference*, pp. 63-72, 1993.

[Mei 74] Mei, K.C.Y., “Bridging and Stuck-At Faults,” *IEEE Transactions on Computers* , Vol. C-23, No. 7, pp. 720-727, Jul. 1974.

[Menon 91] Menon, P., Y. Levendel, and M. Abramovici, "SCRIPT: A Critical Path Tracing Algorithm for Synchronous Sequential Circuits," *IEEE Transactions on Computer-Aided Design*, Vol. 10, No. 6, pp. 738-747, Jun. 1991.

[Millman 88] Millman, S.D., and E.J. McCluskey, “Detecting Bridging Faults with Stuck-At Test Sets,” *Proc. of International Test Conference*, pp. 773-783, 1988.

[Millman 91] Millman, S.D., and J.P. Garvery, Sr., “An Accurate Bridging Fault Test Pattern Generator,” *Proc. of International Test Conference*, pp. 411-418, 1991.

[Pancholy 90] Pancholy, A., J. Rajski, and L. McNaughton, “Empirical Failure Analysis and Validation of Fault Models in CMOS VLSI,” *Proc. of Int. Test Conf.*, pp. 938-947, 1990.

[Perry 92] Perry, R., “IDDQ Testing in CMOS Digital ASIC’s - Putting It All Together,” *Proc. of International Test Conference*, pp. 151-157, 1992.

[Rearick 93] Rearick, J., and J.H. Patel, “Fast and Accurate CMOS Bridging Fault Simulation,” *Proc. of International Test Conference*, pp. 54-62, 1993.

[Savaria 91] Savaria, Y., M. Youssef, B. Kaminska, and M. Koudil, “Automatic Test Point Insertion for Pseudo-Random Testing,” *Proc. of International Symposium on Circuits and Systems*, pp. 1960-1963, 1991.

[Seiss 91] Seiss, B.H., P.M. Trouborst, and M.H. Schulz, “Test Point Insertion for Scan-Based BIST,” *Proc. of European Test Conference*, pp. 253-262, 1991.

[Shen 85] Shen, J.P., W. Maly, and F.J. Ferguson, “Inductive Fault Analysis of MOS Integrated Circuits,” *IEEE Design & Test of Computers*, pp. 13-26, Dec. 1985.

[Storey 90] Storey, T.M., and W. Maly, “CMOS Bridging Fault Detection,” *Proc. of International Test Conference*, pp. 842-851, 1990.

[Storey 91] Storey, T.M., W. Maly, J. Andrews, and M. Miske, “Stuck Fault and Current Testing Comparison Using CMOS Chip Test,” *Proc. of International Test Conf.*, pp. 311-318, 1991.

[Touba 96] Touba, N.A., and E.J. McCluskey, “Test Point Insertion Based on Path Tracing,” *Proc. of VLSI Test Symposium*, 1996.

[Youssef 93] Youssef, M., Y. Savaria, and B. Kaminska, “Methodology for Efficiently Inserting and Condensing Test Points,” *IEE Proceedings-E*, Vol. 140, No. 3, pp. 154-160, May 1993.

[Williams 73] Williams, M.J.Y., and J.B. Angell, “Enhancing Testability of Large-Scale Integrated Circuits via Test Points and Additional Logic,” *IEEE Transactions on Computers* , Vol. C-22, pp. 46-60, Jan. 1973.

[Williams 96] Williams, T.W., R. Kapur, M.R. Mercer, R.H. Dennard, and W. Maly, "IDDQ Testing for High Performance CMOS - The Next Ten Years," *Proc. of European Design & Test Conference*, pp. 578-583, 1996.