DIAGNOSING RESISTIVE BRIDGES USING ADAPTIVE TECHNIQUES

Jayabrata Ghosh-Dastidar and Nur A. Touba

Computer Engineering Research Center
Department of Electrical and Computer Engineering
University of Texas, Austin TX 78712-1084
Email: {dghosh, touba }@cat.ece.utexas.edu

Abstract

A systematic procedure for locating resistive bridges is presented. Critical path tracing is used to identify a set of "suspect" bridges whose presence could explain all of the observed faulty behavior of the circuit for the original test set. The set of suspects is then reduced by adaptively applying additional tests derived from the failing vector pairs in the original test set. Unlike other approaches, the approach presented here is not based on any bridge fault modeling and does not require any fault simulation.

1. Introduction

An increasing prevalent defect in current technologies is resistive bridges. This is due to growing metal density and number of layers of metal. A bridging fault between two lines l_1 and l_2 in a circuit occurs when the two lines are unintentionally shorted during the manufacturing process. When the two lines l_1 and l_2 have opposite logic values the gates driving the two lines will have a logic contention. Depending on various factors including the strengths of the gates driving the two lines, their input values, and the resistance of the bridge, the bridged lines can have a range of values. The logic gates at the fanout of a bridged node may have varying threshold voltages and may interpret the voltages at the bridged nodes differently. This effect is well known as the *Byzantine Generals problem* [Acken 92], [Maxwell 93]. One of the factors that determines the behavior of a bridge that is of growing importance and interest is the resistance of the bridge. Various studies [Renovell 95], [Mandava 99], have shown that the resistance of the bridge has a pronounce effect on the behavior of the interconnect bridging defect. For low resistance values the bridge behaves more as a static logic failure, *i.e.*, one end of the bridge doesn't reach its correct logic value when logic contention occurs. But as the bridge resistance increases, the bridge fault start causing more speed failures rather than logic failures. In the second case, both ends of the bridge reaches their correct logic values when a logic contention occurs but one of them gets delayed. As feature sizes are scaled down, the metal pitch is reduced in tandem to increase density. Reduced metal pitch in turn imposes limitations on the height of metal interconnects which must also decrease to improve manufacturability. Thus the line resistance goes up nearly quadratically. So in future devices, we can expect to see more speed failures caused by resistive bridges in comparison to logic failures.

Several techniques have been proposed for diagnosing bridges that cause logic failures. In [Millman 90], a method to diagnose bridging faults using stuck-at dictionaries was presented. [Chess 95] and [Lavo 96] further improved this technique. These techniques are constrained to using a reduced set of faults extracted from the layout as they enumerate the bridging faults. [Chakravarty 93] proposed a voltage-based algorithm that uses a wired-AND (wired-OR) model and stuck-at fault dictionaries. All these methods are based on using stuck-at fault dictionaries, the creation and storage of which might be quite expensive. Also, as the behavior of the bridging defect start diverging from stuck-at behavior, the accuracy of such techniques can degrade. Any method based on stuck-at fault dictionary will also have to worry about the stated *Byzantine Generals problem.* [Venkataraman 97] proposed a deductive strategy for diagnosing bridging faults, but that method does not consider the possibility that resistive bridges may cause delay failures.

In this paper, we present a new method based on critical path tracing and adaptive generation of additional vectors to improve the diagnostic resolution. The method presented here takes into consideration the fact that a bridge defect can behave as a delay fault. This is more general than the previous approaches as the stuck-at behavior of a bridge defect can be thought as a delay defect of infinite size. Since the method here uses critical path tracing as a starting point, there is no requirement for creating fault dictionaries or explicit simulation of the faults. It is independent of the faulty behavior of the bridge, i.e., there is no need to assume a fault model like wired-OR or wired-AND at the logic level.

The method presented here uses critical path tracing to identify a set of "suspect" bridges, whose presence would explain all of the observed faulty behavior of the circuit for the original test set. The set of suspects is then reduced by adaptively applying additional tests that are derived from the failing vector pairs in the original test set. Two strategies are described for generating additional vectors that help reduce the potential candidates for the bridging fault. These techniques reduce the search space and help guide direct probing which can save a lot of time during failure analysis. Experimental results indicate that the number of suspect lines can be greatly reduced.

0-7803-5809-0/00/$10.00 © 2000 IEEE

In the remainder of the paper, it is assumed that bridges will act as delay faults, but note that all the strategies described here are valid even when the bridge causes static logic errors since that is a special case of a delay fault where the delay value is infinite.

2. Critical Path Tracing

A delay defect requires a two-pattern test for its detection. So in our diagnosis procedure, we consider failing vector pairs (V_1, V_2) instead of just the failing vectors individually. For each failing vector pair (vector pair which gave an erroneous output), a six-valued simulation is performed and critical path tracing done starting from each failing output. The idea of performing critical path tracing using a 6-valued algebra to identify a set of suspects that may explain an observed faulty output was proposed in [Girard 92]. For the original test sequence, each two-pattern test for which the circuit-under-test produced a faulty output is simulated using a 6-valued algebra based on the H6 algebra [Hayes 86]. The symbols used are the following: *S0* for static zero, *S1* for static one, *R1* for a rising transition, *F0* for a falling transition, *X0* for static-0 hazard, and *X1* for a static-1 hazard. The advantage of using this 6-valued algebra is that it does not depend on any gate propagation delay or delay fault size. From each faulty output, critical path tracing is performed to identify the suspects (i.e., critical lines) that may have caused the faulty value. A suspect is a fault that if present could explain all of the observed faulty behavior. For each failing vector pair (V_1, V_2), for which the circuit-under-test (CUT) has produced an erroneous output we construct two sets *SUSPECT_0* and *SUSPECT_1*. *SUSPECT_0* denotes those suspect lines obtained by critical path tracing from a failing output where the correct logic value at that line for vector V_2 is zero. Similarly we can define *SUSPECT_1* as those critical lines for which the correct logic value after vector V_2 is applied is one. We also define two additional sets *LINE_0* and *LINE_1*, where *LINE_0* denotes those lines that are not part of the critical lines and the logic value at that line after application of vector V_2 is zero. *LINE_1* is similarly defined. Given these four sets, the set of all possible bridge faults are: any node in *SUSPECT_0* bridged with any node in *SUSPECT_1*or *LINE_1*and any node in *SUSPECT_1* bridged with any node in *SUSPECT_0* or *LINE_0*.

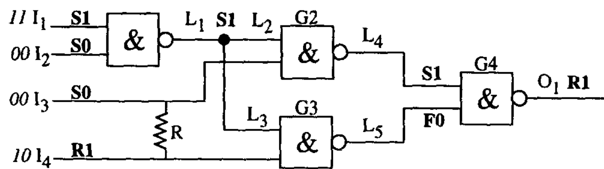

Figure 1. Six valued simulation for the failing vector pair (*1000, 1001*)

Figure 1 shows an example where the two lines I_3 and I_4 have been shorted by a resistive bridge. Let as assume that on application of the vector pair (*1000, 1001*) the logic value I_4 is slow to rise, causing a erroneous output at O_1. We perform 6-valued simulation for this vector pair, and do critical path starting from output O_1. So *SUSPECT_0* = $\{L_5\}$ and *SUSPECT_1* = $\{O_1, I_4\}$. Now if we consider what lines were at values 0 and 1 after application of vector *1001*, we obtain sets *LINE_0* = $\{I_2, I_3\}$ and *LINE_1* = $\{L_1, L_2, L_3, L_4, I_1\}$. So the possible bridges are any element of *SUSPECT_0* shorted to any element of *SUSPECT_1* or *LINE_1*, or any element of *SUSPECT_1* shorted to any element of *SUSPECT_0* or *LINE_0*. Note here that these are the only possible bridges in the circuit consistent with the observed behavior of the circuit when tested. Let us define *BRIDGE_SUSPECTS* as the set of all suspect bridges in the circuit in any step of the diagnosis process. By looking at the suspect sets for each failing test, we can construct the set *BRIDGE_SUSPECTS*.

For each failing vector pair 6-valued simulation is performed and the sets *SUSPECT_0, SUSPECT_1, LINE_0, LINE_1* are created. Any bridge $\{l_1, l_2\}$ in the set *BRIDGE_SUSPECTS* has to satisfy one of the following conditions for every failing vector pair.

1. $l_1 \in$ *SUSPECT_0* and $l_2 \in$ (*SUSPECT_1* $\cup$ *LINE_1*)
2. $l_1 \in$ *SUSPECT_1* and $l_2 \in$ (*SUSPECT_0* $\cup$ *LINE_0*)

After constructing the set *BRIDGE_SUSPECTS*, if its cardinality is small enough, the diagnosis is stopped. Otherwise additional vectors are generated adaptively to further narrow down the set of possible bridges. Two techniques are described for generating the diagnostic vectors in the next two sections. Both techniques take the failing vector pairs as their starting point.

Note that it is very unlikely that two nodes far apart in the layout of the circuit can ever be bridged. So if layout information is available then the set *BRIDGE_SUSPECTS* can be created more realistically by including only those bridges that can occur given the circuit layout and the possible defect sizes. The procedure remains the identically the same just that the domain of bridges considered is reduced by using layout information.

3. Deriving Minimum Input Transition Test

In this strategy the objective is to reduce the number of suspect lines derived by critical path tracing. The idea behind such a strategy is based on the fact that a node pair is eliminated from the set *BRIDGE_SUSPECTS* if none of the nodes in that pair are part of the suspect set derived by critical path tracing of a faulty output. So if we can reduce the size of the suspect sets derived by critical path tracing then we can reduce the cardinality of *BRIDGE_SUSPECTS*. The strategy here is to begin with an original two-pattern test that failed and systematically reduce the number of

transitions as much as possible while still detecting the fault. Given a two-pattern test $(V_{1,orig}, V_{2,orig})$ in the original sequence that failed, let $DIFF_INPUTS(V_{1,orig}, V_{2,orig})$ be the set of inputs whose values differ in $V_{1,orig}$ and $V_{2,orig}$ (i.e., the set of inputs on which there are transitions). If there are n inputs in the set $DIFF_INPUTS(V_{1,orig}, V_{2,orig})$, then the first step is to derive n two-pattern tests by simply removing one of the input transitions in the original two-pattern test. This is done by setting the corresponding input bit value in the V_1 pattern equal to that in the V_2 pattern. One of the resulting two-pattern tests that still produces a faulty output is then chosen arbitrarily and the process repeats recursively until a point is reached where none of the derived two-pattern tests produces a faulty output. The two-pattern test with the fewest number of input transitions, $(V_{1,min_tran}, V_{2,min_tran})$, that still produces a faulty output can then be used for critical path tracing to generate the suspect set. The advantage of using a minimum input transition test during diagnosis is that if that test results in a faulty output value, then the set of suspects derived by critical path tracing will be very small. This is because there is a transition on a smaller number of inputs, so the number of lines in the circuit which are tested by the new test is relatively small which makes diagnosis much easier. By deriving minimum input transition tests that fail from the original tests that failed, we are able to get a better resolution in diagnosing the resistive bridging faults.

4. Deriving Additional Tests

In the second technique for improving diagnostic resolution, we modify the V_2 vector to a new vector $V_{2,new}$, with the condition that the resulting vector pair $(V_1, V_{2,new})$ still causes the CUT to fail. The motivation behind modifying the vector V_2 is that some nodes in the CUT will have complementary values for vector V_2 and $V_{2,new}$. So those elements in the set *BRIDGE_SUSPECTS* that have one node that has complementary values for vector V_2 and $V_{2,new}$ will now fail the conditions 1 or 2 defined in Sec. 2 and will be pruned out. This is because the error is still observed at the output, but now both sides of the bridge have the same value, so it cannot be the location of the defect. Let us assume $\{l_1, l_2\}$ is a bridge fault in the set *BRIDGE_SUSPECTS*, where the logic value at the two lines after application of V_2 are $logic(l_1) = 0$ and $logic(l_2) = 1$. If after application of vector $V_{2,new}$, the logic values at the two nodes are $logic(l_1) = 0$ and $logic(l_2) = 0$ then $\{l_1, l_2\}$ is no longer a candidate of *BRIDGE_SUSPECTS* because it could not have caused the observed faulty behavior.

The initial strategy for creating $V_{2,new}$ is based on finding a minimum set of inputs for the vector V_2 that has to be kept unchanged in $V_{2,new}$ for the CUT to fail for the vector pair $(V_1, V_{2,new})$. If the vectors are n bits wide, we first create n vector pairs (V_1, V_2') where each V_2' is created by complementing one bit position in V_2. The set of n vector pairs is applied to the CUT. Among the vector pairs that fail, we choose one and repeat the same process, till we cannot create a vector pair that fails. This is a greedy heuristic and is the same as the one described for deriving minimum input transition test. We take all the vector pairs for which the inputs were complemented and the CUT failed and perform 6-valued simulation, critical path tracing and pruning for them. Since input positions in each $V_{2,new}$ vector was forced to differ from the V_2 vector, there will be many nodes in the CUT will have complementary values for the two vectors, and hence some node pairs will be eliminated from the set *BRIDGE_SUSPECTS*. One advantage of our diagnosis strategy is that in no step does it add anything to the set *BRIDGE_SUSPECTS*. So in any step, it can only decrease the cardinality of the set *BRIDGE_SUSPECTS*. Additional $(V_1, V_{2,new})$ vector pairs can be generated by choosing different failing vectors in each step of deriving $V_{2,new}$.

If after these steps, the cardinality of the set *BRIDGE_SUSPECTS* is small enough, the diagnosis process is stopped. Otherwise, a targeted approach can be used to generate further two-pattern tests to reduce the set *BRIDGE_SUSPECTS*. A new vector $V_{2,X}$ is created such that $V_{2,X}$ and $V_{2,new}$ are same except all the bit positions in which V_2 and $V_{2,new}$ differ are now made 'X' in $V_{2,X}$. For all the remaining node pairs in *BRIDGE_SUSPECTS* it is determined whether any node in the pair has an X-path through it for $V_{2,X}$. Let us assume $\{l_1, l_2\}$ is such a pair, with l_1 having a X-path through it for $V_{2,X}$. Also assume that the logic value at l_1 for vector V_2 was 0. Then automatic test pattern generation (ATPG) can be performed to justify a '1' at l_1 with $V_{2,X}$ as the initial input assignment. The ATPG is not allowed to alter the initial input assignments, only assign values to the X's. All such nodes l_1 that have X-paths through them for $V_{2,X}$ can be targets of additional vector generation. The additional two-pattern tests that derived in this manner can be used to prove that the targeted bridge is not the cause of the faulty output. If the output response when the new test is applied still is erroneous, then the bridge where both lines now have the same logic value on them cannot be the cause of the faulty output and can be eliminated from the set *BRIDGE_SUSPECTS*.

5. Experimental Results

Experiments using the adaptive techniques described in this paper were performed for some of the ISCAS 85 benchmark circuits [Brglez 85]. For the experiments, two nodes were selected at random and a bridge fault was inserted between them. Table 1 shows sample results for the experiments. The first column shows the circuit, the second column shows the number of test vectors generated

by the tool *Soprano* [Soprano 90]. By just doing critical path tracing, the resulting number of suspect lines are shown. These are lines in the circuit which contain one of more bridges in the set *BRIDGE_SUSPECTS.* For the suspect lines, the next column shows the average number of bridge suspects that are associated with that line. Results are then shown for the set of suspect lines after the adaptive techniques have been applied. Note the significant improvement in diagnostic resolution that occurs by using the adaptive techniques.

Table 1. Experimental Results for Fault Diagnosis of Resistive Bridges causing Delay Fault

Circuit	Test Length	Critical Path Tracing Alone		Adaptive Techniques	
		Number of Suspect Lines	Avg. Num. of Bridge Suspects	Number of Suspect Lines	Avg. Num. of Bridge Suspects
C432	160	78	11	41	5.5
C432	160	85	12.3	52	8.8
C432	160	119	15.8	100	9
C880	198	151	4.4	144	3.5
C880	198	110	4.5	16	4.5
C880	198	251	27.8	205	9.13
C1908	409	168	15.3	17	13
C1908	409	390	20	52	10
C1908	409	464	62.2	31	8.4

6. Conclusion

The diagnostic procedure described in this paper targets resistive bridges. Both bridges that cause static logic faults as well as those that cause delay faults can be handled. A hill climbing strategy is used to continually decrease the number of bridge suspects by deriving additional two-pattern tests from the tests that failed in the original test set to further prune the suspect set. The procedure presented in this paper can be used to reduce the search space for direct probing and better guide it towards finding resistive bridge defects. This results in less time and lower cost for failure analysis.

References

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

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

[Chakravarty 93] Chakravarty S. and Y. Gong, "An Algorithm for Diagnosing Two Line Bridging Faults in CMOS Combinational Circuits", in *Proc. of Design Automation Conf.*, pp. 520-524, June 1993.

[Chess 95] Chess B. D. B. Lavo, F. J. Ferguson, and T. Larrabee, "Diagnosing of Realistic Bridging Faults with Stuck-at Information", in *Proc. of the IEEE/ACM Intl. Conf. On Computer-Aided Design*, pp. 268-271, Nov. 1995.

[Girard 92] Girard, P., C. Landrault, S. Pravossoudovitch, "A Novel Approach to Delay-Fault Diagnosis", *Proc. 29th Design Automation Conference*, pp. 357-360, 1992.

[Hayes 86] Hayes, J.P., "Digital Simulation with Multiple Logic Values", *IEEE Trans. Computer-Aided Design*, vol. 5, no. 2, pp. 274 -283, Apr. 1986.

[Lavo 96] Lavo D. B. T. Larrabee, and B. Chess, "Beyond the Byzantine Generals: Unexpected Behavior and Bridging Fault Diagnosis," in *Proc. of International Test Conference*, pp. 611-619, Oct. 1996.

[Mandava 99] Mandava S., S. Chakravarty, S. Kundu, "On Detecting Bridges Causing Timing Failures", *Proc. Of International Conf. On Computer Design (ICCD)*, 1999.

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

[Millman 90] Millman S. D., E. J. McCluskey, J. M. Acken, "Diagnosing CMOS Bridging Faults with Stuck-At Fault Dictionaries", *Proc. of International Test Conference*, pp. 860-870, 1990.

[Renovell 95] Renovell M., P. Huc and Y. Bertrand, "The Concept of Resistance Interval: A New Parametric Model for Realistic Bridging Faults", pp. 184-189, 1995.

[Venkataraman 97] Venkataraman S., and W. K. Fuchs, "A Deductive Technique for Diagnosis of Bridging Faults", in *Proc. of the IEEE/ACM Intl. Conf. On Computer-Aided Design*, pp. 562- 567, 1997.