Interconnect Open Defect Diagnosis with Minimal Physical Information*

Chen Liu[1] Wei Zou[2] Sudhakar M. Reddy[1] Wu-Tung Cheng[2]
Manish Sharma[2] Huaxing Tang[2]

1. ECE Dept.
University of Iowa
Iowa City, IA 52242, USA
{cheliu,reddy}@engineering.uiowa.edu

2. Mentor Graphics Corporation
8005 SW Boeckman Road
Wilsonville, OR 97070, U.S.A
{wei_zou,wu-tung_cheng,
manish_sharma,
huaxing_tang}@mentor.com

Abstract

We consider the problem of determining the location of open defects in interconnects of deep submicron (DSM) designs. The target defect sites for this work are the vias in interconnects which are known to be defect prone. It is known that in DSM designs below 90 nm technology the circuit parameters may vary widely from nominal or design values and process variations make them less predictable. Thus it becomes necessary to develop methods for locating defect sites without accurate knowledge of circuit parameters. Logic diagnosis which is based on gate level net lists is one such method but the resolution of defect sites obtained by logic diagnosis is considered to be unacceptably low for locating open vias. We investigate a procedure that uses minimal information beyond the net lists and give experimental results to demonstrate the defect resolution obtained using the method. The additional information used by the proposed method is a list of nodes in the neighborhoods of circuit nodes and the circuit layout. Specifically, difficult to determine circuit parameters of manufactured instances of a design such as coupling capacitances between circuit nodes and threshold voltages of gates in the circuit are not needed to use the proposed diagnosis procedure.

Key Words: Fault Diagnosis, Defect Location, Interconnect Opens, Via Opens.

1. Introduction

In deep sub-micron (DSM) designs open is a common defect type. Opens most frequently occur in contacts and vias. Open can be of finite resistance or infinite resistance (complete opens). In this work we consider complete opens in vias in circuit interconnects.

When a complete open occurs some circuit node is disconnected from the gate driving it and the disconnected node is said to be floating. The open node is part of a circuit net which typically contains many sections of interconnect and vias. Given a circuit net with an open, the location of the open can be determined from attenuation and phase shift measurements during physical failure analysis [1]. Fault diagnosis procedures are first used to determine a list of candidate sites for physical failure analysis. Depending on the fault diagnosis procedure used, the candidate sites could be circuit nets or segments or vias. Methods that determine open segments narrow the sites for failure analysis more than the methods that locate the opens to within circuit nets. Similarly methods that resolve the opens to vias in circuit nets provide much shorter interconnect sections for investigation during failure analysis. The work reported in this paper considers locating interconnect opens to locations of vias in circuit nets.

The voltage on a floating node depends on several things including the state of the neighboring nodes and coupling capacitances between the floating node and its neighbors, the capacitances to power supply lines and substrate, initial trapped charge, leakage currents and the internal capacitances of the gates driven by the floating node [2 – 4]. Additionally, depending on their threshold voltages, the voltage on the floating node may be interpreted differently by different gates driven by the open node. This is referred to as Byzantine effect [5]. Several methods have been proposed to diagnose interconnect open defects [6 – 19]. The methods in [6 – 16] use gate level net lists only and do not use layout or cell library information. Such diagnosis procedures are referred to as logic diagnosis procedures in this work. The logic diagnosis methods include diagnosis based on the net fault model [6, 7], symbolic simulations [8, 9], incremental heuristic using X simulations [10, 15] and location based diagnosis [11 – 14,

---

* The work of SMR was supported in part by Semiconductor Research Corporation Grant 2004-TJ-1243.

16]. These logic diagnosis methods typically report a candidate list that contains suspect circuit nets. Circuit nets may have many sections of interconnects that span multiple metal layers and have many vias. Tracing a suspect net during physical failure analysis to find the defect site could be a long and expensive task.

To reduce the time and cost of physical failure analysis, Huang [17] proposed segment fault model. A segment is a unique subset of gates driven by a gate through an interconnect net. Segment fault model requires the layout of the circuit in addition to the gate level net list to determine the gates in a segment. Symbolic simulation is proposed to locate suspect segments of nets that are likely to contain the open via [17]. Thus the method of [17] reports suspect segments instead of suspect nets as done by logic diagnosis procedures. Hence diagnosis based on segments identifies smaller sections of interconnects in circuit nets for physical failure analysis. Sato et al. [18] proposed to identify open vias instead of open segments by using a physical interconnect open model. This method uses the values of capacitances between the floating node and its neighboring nodes and also considers the initial trapped charge on the floating node. However the capacitances between internal nodes of the driven gates are not included in the model. Additionally the threshold voltages of all gates driven by the floating node are considered to be identical. In [19], Zou et al. investigated a procedure which takes into account the capacitances between the floating node and its neighboring nodes, the initial trapped charge, the internal capacitances of the driven gates and differing gate input threshold voltages of the driven gates. The threshold voltages of library cells were determined in a preprocessing step. This method is more accurate since essentially all the circuit parameters that determine the behavior of the floating node are included. Since the methods of [18] and [19] determine open vias the diagnosis resolution provided by them is finer than that provided by [17] using segment fault model. However the methods of [18] and [19] need accurate extraction of capacitances between circuit nodes which may not be feasible in nanometer designs. Also the threshold voltages of different instantiations of the same library cell in a design may vary considerably and hence threshold voltages that are determined a priori may not accurately reflect the actual values in a manufactured design. For nanometer devices which may have large process variations and whose circuit parameters may deviate considerably from nominal values, diagnosis procedures that do not use extracted capacitance values and parameters of library cells may be needed.

In this paper, we investigate a diagnosis method with the goal to determine open vias. It uses knowledge of neighbors of a circuit node and the circuit layout only. The information regarding the neighboring nodes can be obtained through proximity analysis and hence can be regarded as a minimal requirement. Layout information is needed for any method whose goal is determination of the location of open vias. Specifically, in the proposed method the coupling capacitances between circuit nodes and parameters of library cells, such as threshold voltages and internal capacitances are regarded as unknowns. Hence, the method does not require extraction of coupling capacitances and knowledge of inter node capacitances and threshold voltages of library cells in the devices which failed manufacturing test.

The rest of the paper is organized as follows. In Section 2 we briefly review the previous related work and give an overview of the proposed procedure. In Section 3 we describe the proposed procedure. In Section 4 we present experimental results. In Section 5 we have some discussions. Section 6 concludes the paper.

2. Preliminaries

In this section we first give a brief review of previous works related to the proposed method followed by an overview of the proposed diagnosis procedure to identify open vias in interconnects.

2.1 Review of previous works

If a net in the gate level netlist drives multiple gates, the routing of the net in the layout can be divided into several segments that drive different subsets of gates [17].

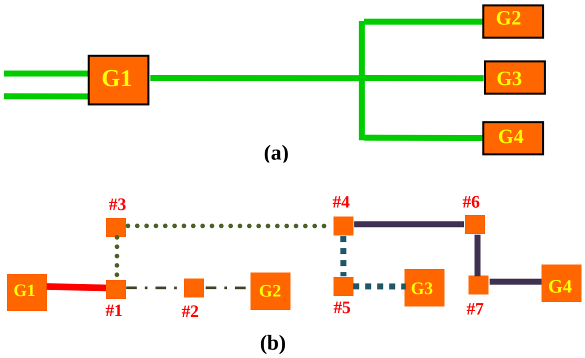

Figure 1: A net's routing in the layout

In Figure 1(a) we show a net driven by gate G1 driving gates G2, G3 and G4. The net can be considered to contain five segments [17] as shown in Figure 1(b). Each segment contains a part of the net that drives different subsets of gates. The five segments S1, S2, S3, S4 and S5 in Figure 1 (b) drive subsets of gates {G2, G3, G4}, {G2}, {G3, G4}, {G3} and {G4}, respectively. In Figure 1(b) the smaller squares represent vias. From the layout we can determine that segments S1, S2, S3, S4 and S5 contain subsets of vias {1}, {2}, {3, 4}, {5} and {6, 7}, respectively. The method by Huang [17] locates open vias up to segments. For example if segment S3 is identified as containing the open via then the actual open via could be via 3 or via 4. The methods of [18] and [19] attempt to obtain better resolution using extracted capacitances coupled to the open net. For example in the previous case these methods may determine which of the vias 3 or 4 is open. This is possible because the capacitances coupled to the open node and the neighbors may be different when via 3 is open compared to when via 4 is open. The neighbors are the nodes that are in the neighborhood of the sections of interconnect downstream of the open. Because of the distance between vias 3 and 4, their neighbors are quite likely to be different. Similarly, in segment S5 it may be possible to locate the open via to 6 or 7. The goal of our work is also to locate the open vias in segments similar to that of [18] and [19] but without requiring the knowledge of circuit parameters whose values may not be determinable precisely.

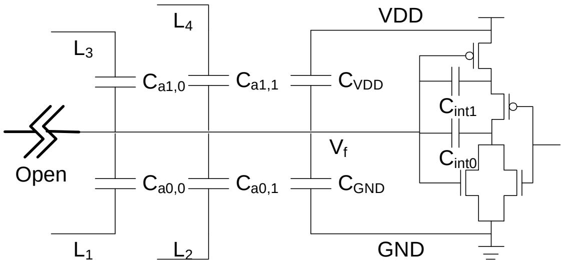

Figure 2: Interconnect open model

The model used in [18, 19] to determine the voltage on a floating node is illustrated in Figure 2, which shows an input node of a 2-input NOR gate open. The voltage V_f on the floating node satisfies the following equations:

$$V_f = \frac{C_1}{C_0 + C_1} V_{dd} + \frac{Q_{trap}}{C_{gnd}} \quad (1)$$

$$C_0 = C_{gnd} + C_{a0} + C_{\text{int}0} \quad (2)$$

$$C_1 = C_{vdd} + C_{a1} + C_{\text{int}1} \quad (3)$$

In the equations above, C_{a0} and C_{a1} are the sums of the capacitances between the floating node and its neighboring nodes which have logic 0 and logic 1 values, respectively. Q_{trap} is the initial trapped charge of the floating node. C_{vdd} and C_{gnd} are the capacitances between the floating node and power supply and ground rail. Because the voltages on the adjacent nodes depend on the test pattern *P* applied, we have $C_{a0}= C_{a0}(P)$ and $C_{a1}=C_{a1}(P)$. That is, these capacitances are pattern dependent. C_{int0} and C_{int1} are the capacitances internal to the driven gate whose actual values depend on voltage V_f [2].

To avoid calculating V_f explicitly, [18] defines a variable *E* and also assumes that C_{a0} and C_{a1} are dominant.

$$E = E(P) = \frac{C_1}{C_0 + C_1} = \frac{C_{a1}(P)}{C_{a0}(P) + C_{a1}(P)} \quad (4)$$

Since Q_{trap} / C_{gnd} is constant, variable *E* is sufficient to capture the change in V_f with varying patterns. Furthermore the test patterns that imply logic value 0 and 1 at the floating node of the circuit are divided into sets Ω0 and Ω1, respectively. Ω0 (Ω1) is the set of test patterns under which the floating node voltage is less than (larger than) the threshold voltage of the driven gate. In [18] the threshold voltages of all driven gates are assumed to be identical. Next two ranges of values, E(Ω0) and E(Ω1), defined below are introduced.

$$E(\Omega 0) = [\min E(P), \max E(P)] \ \forall P \in \Omega 0$$

$$E(\Omega 1) = [\min E(P), \max E(P)] \ \forall P \in \Omega 1$$

For a floating node due to an open defect the following must be true:

$$E(\Omega 0) < E(\Omega 1) \quad (5)$$

The meaning of equation (5) is that the range of values in E(Ω0) must be below the range of values in E(Ω1).

The method in [18] was enhanced in [19] by computing the threshold voltages of all library cells and using them in the diagnosis procedure. Each input of every gate type has a threshold. Additionally, the capacitances internal to the driven gates are considered in the form of trapped charge:

$$Q_{trap} = Q_{wire}(P, V_f) + Q_{gate}(V_f) \quad (6)$$

$$Q_{wire}(P,V_f) = V_f \bullet C_{a0}(P) + (V_f - V_{dd}) \bullet C_{a1}(P) \quad (7)$$

In equations (6) and (7), $Q_{wire}(P, V_f)$ is the sum of the charge stored in the capacitors between the floating node and its neighboring nodes. Q_{gate} is the charge stored in the capacitors inside the gate driven by the suspect via. The remaining variables are as defined earlier. The test patterns are also divided into two sets Ω0 and Ω1 as defined earlier. Let

$$Q(P,V_f) = Q_{wire}(P,V) + Q_{gate}(V).$$

For patterns in Ω0, let V_i be the smallest threshold voltage of the driven gates that has fault effect, then $V_f < V_i$ and we have $Q_{trap} < Q(P_i,V_i)$. For patterns in Ω1, let V_j be the largest threshold voltage of the driven gates that has fault effect, then $V_j < V_f$ and we have $Q(P_j,V_j) < Q_{trap}$. Then for a candidate open via, the following should be true:

$$\text{Max}\{Q(P_j,V_j)\} < \text{Min}\{Q(P_i,V_i)\}, \ \forall P_i \in \Omega 0, \forall P_j \in \Omega 1 \quad (8)$$

The method in [19] can achieve better resolution since it includes different threshold voltages for different library cells and also implicitly includes capacitances internal to driven gates. However as noted earlier, both methods require values of inter node coupling capacitances and use threshold voltage information, both of which may not be accurately known for nanometer designs.

2.2 Overview of the proposed diagnosis procedure

The proposed diagnosis procedure first identifies a set of candidate segments that are open using segment fault model [17] and determines the vias in the suspect segments. Next, the vias in the suspect segments are analyzed to determine the suspect vias. Interconnect open model discussed in the section above is used during this step. However, the actual values of various capacitances and gate threshold voltages are assumed to be unknown. Using variables to represent these unknown quantities, for each via in a suspect segment certain sets of inequalities are set up. The inequalities which have solution identify the vias which are suspected to be open by the proposed method. As in earlier works [18] and [19] we assume that the defect in chip being diagnosed has a single open via.

3. Identifying open defects with only neighborhood node list information

In this section we describe the proposed diagnosis procedure. The procedure uses three steps. In the first step candidate open nets are identified using logic diagnosis. In the second step, the candidate open segments are identified using segment fault model. In the third step suspect open vias are determined. These steps are described next.

3.1 Identifying the open nets using logic diagnosis

In the first step the logic diagnosis procedure of [16] is used to derive a list of candidate nets. We use only SLAT [12] (single location at a time) failing patterns in this step. SLAT patterns are those patterns that can be explained by a single fault site [12]. That is, the circuit outputs produced by the failing chip when the pattern is applied are matched by a single fault injected at a fault site. When all the single fault sites that can explain some failing SLAT pattern(s) are determined a set of candidate circuit nodes are obtained. At this point, typical logic diagnosis procedures [11-14] determine minimal sized subsets of the set of candidate sites that can explain all the failing SLAT patterns. Each such subset corresponds to potential candidate sites for defects. Each candidate subset is then analyzed based on fault models used. Since all the branches of a fanout stem are the same net in the layout, in [16], if one or more branches of a fanout stem are included in the set of candidate fault sites, all branches are replaced by their parent fanout stem followed by determining a minimum set cover to find the subsets of candidates with minimal size which can explain all the failing patterns. Thus, at the end of logic diagnosis suspect nets are determined.

3.2 Identifying the open segments using segment fault model

Next a segment fault model described in [17] is used to find the open segments and only the vias on the open segment will be analyzed by the physical open model in the next step described in Section 3.3. If a segment drives multiple gates, it may cause failures on one or more driven gates. We simulate all the possible combinations for the faults. Each driven input can have faulty or not faulty values. So for a segment including N gates, (2^N – 1) multiple faults are simulated. A segment is added to the list of suspect segments if any one of these faults explains the failing pattern. Otherwise the segment is dropped from the candidate list. Since some gates are included in several segments we simulate faults of increasing multiplicity (i.e. faults with multiple fault sites) as well as use the results of fault simulation used in logic diagnosis of Step 1 discussed in the last section. We also take advantage of the earlier proposed methods to fault simulate multiple faults associated with stems of large fan outs described in [20]. These steps aid in improving the efficiency of the procedure to simulate multiple faults at gates in a segment.

3.3 Identifying open vias by solving inequalities

Diagnosis procedures for interconnect opens use the information regarding a fault at a gate input explaining or not explaining the observed response from a tested device. In determining the expected behavior of the device under test, when a candidate open defect is considered, one needs to know the threshold voltage of the gate inputs driven by the floating node. For primitive gates such as NAND, NOR etc. for each gate input only one threshold voltage is needed, since inputs to such gates can be sensitized only if all other inputs to these gates are at non-controlling value. In non-primitive gates a gate input may be fanned out to more than one pair of NFET and PFET. For such gates one will have to use more than one threshold voltage for a gate input. In general if a non-primitive gate has n inputs then for each gate input one may have to use 2^(n-1) threshold voltage values. Multiple threshold voltages at a non-primitive gate input are also needed for the procedure in [19].

Next we give a sketch of the proposed method by giving the details for the case when a suspect via drives a single gate. In Section 3.4 we discuss the general case of a candidate open via driving multiple gates. In the following the failing patterns are patterns that failed the device under test on the tester and the passing patterns are those that passed the device under test. By a fault explaining a failing pattern we mean that circuit with the fault produced outputs observed on the tester when the pattern was applied.

Recall that prior to identifying suspect vias, suspect segments have been determined in Step 2 discussed in Section 3.2. Corresponding to each suspect segment the set of test patterns can be divided into failing and passing patterns. These patterns are analyzed to set up inequalities corresponding to the gate driven by vias on the suspect segments to determine potentially open vias.

In the model for interconnect opens described in Section 2.1, let $C_{tot} = C_0 + C_1$.

For each gate driven by the subnet that is floating due to an open via the set of test patterns are divided into five classes, Cls1 to Cls5, defined below. Basically each failing and passing pattern is analyzed to see if the logic value of the floating node can be determined by comparing the tester fail log and fault simulation results. The information needed for this classification is known from fault simulations used in Steps 1 and 2 discussed in Sections 3.1 and 3.2.

Cls1 = {P: a failing test pattern under which the driven gate has a fault value of logic 1 that explains this pattern. That is, P detects a stuck-at 1 fault at the input of the driven gate and produces the same circuit output fails as observed on the tester when P was applied to the failing device}

Cls2 = {P: a failing test pattern under which the driven gate has a fault value of logic 0 that explains this pattern. That is, P detects a stuck-at 0 fault at the input of the driven gate and produces the same circuit output fails as observed on the tester when P was applied to the failing device}

Cls3 = {P: a passing test pattern under which a faulty logic value of 1 at the input to the driven gate would be detected but was not detected on the tester or a failing test pattern which can detect a faulty logic value of 1 on the open node but the fault does not explain the fail log for the pattern}. **Note:** In this case, V_f the voltage on the floating node should be less than the threshold of the driven gate which causes the gate input to have fault-free logic value 0.

Cls4 = {P: a passing test pattern under which a faulty logic value of 0 at the input to the driven gate would be detected but was not detected on the tester or a failing test pattern which can detect a faulty logic value of 0 on the open node but the fault does not explain the fail log for the pattern}. **Note:** In this case, V_f the voltage on the floating node should be greater than the threshold of the driven gate that causes the gate input to have fault-free logic value 1.

Cls5 = {P: a test pattern that does not belong to the classes 1 to 4}.

The voltage, V_f on a floating node caused by an open via should simultaneously satisfy the following sets of inequalities (note that each pattern in classes Cls1 through Cls4 gives rise to one inequality):

$$\begin{cases} V_f(P) = \frac{C_1(P)}{C_{tot}} V_{dd} + \frac{Q_{trap}}{C_{gnd}} > V_{th}(P), \forall P \in Cls1 \cup Cls4 \\ V_f(P) = \frac{C_1(P)}{C_{tot}} V_{dd} + \frac{Q_{trap}}{C_{gnd}} < V_{th}(P), \forall P \in Cls2 \cup Cls3 \end{cases} \quad (9)$$

As discussed earlier, the threshold voltage $V_{th}(P)$ is pattern dependant for non-primitive gates. However it is the same for two patterns that set the inputs of the driven gate to the same value. The first set of inequalities are for the case when either a faulty logic value of 1 is detected at the input of the driven gate (i.e. patterns in Cls1) or a faulty logic value of 0 was not detected (i.e. a passing pattern Cls4) because V_f, the voltage on the floating node was more than V_{th}. The second set of inequalities are for the case when either a faulty logic value of 0 is detected at the input of the driven gate or a faulty logic value of 1 was not detected (by a passing pattern) because V_f, the voltage on the floating node, was less than V_{th}. If the set of simultaneous inequalities does not have a solution for some set of values of capacitances, trapped charge and threshold voltage, then the via under consideration is defect free and can be removed from the candidate list.

We convert the inequalities to linear restrictions as described next. Assume that C_{tot} and V_{dd} are known to be constants. Assume C_{int0} and C_{int1} are pattern independent constants. Let $k = (Q_{trap}/C_{gnd})*C_{tot} + (C_{int1}+C_{vdd})*V_{dd}$, and k is a pattern independent variable. Then (9) becomes:

$$\begin{cases} C_{a1}(P)V_{dd} + k - V_{th}(P)C_{tot} > 0, \forall P \in Cls1 \cup Cls4 \\ C_{a1}(P)V_{dd} + k - V_{th}(P)C_{tot} < 0, \forall P \in Cls2 \cup Cls3 \end{cases} \quad (10)$$

Note that the above inequalities are linear. We can use a simplex method based solver [21] to determine a solution if it exists. If the solver reports that inequalities in (10) have no solution the suspect via is removed from the candidate list, otherwise it is retained.

The following example is used to illustrate the proposed method.

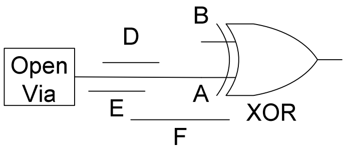

Figure 3: Circuit for example 1

Example 1: Consider the case, illustrated in Figure 3, of an open via driving input A of an exclusive OR gate with two inputs A and B. Let the neighbors of node A be nodes D, E and F. Because the XOR gate's threshold on input A is dependent on the input of B, we have two thresholds for A: V_{thB0} when B is 0 and V_{thB1} when B is 1. The two thresholds are to address the fact that with B = 0 and B = 1 the gate output is effected through different transistors/paths driven by input A. As discussed earlier, in general, the use of multiple thresholds is necessary for non-primitive gates. Also assume that there are three test patterns P1 a failing pattern, P2 a passing pattern and P3 a failing pattern belonging to classes Cls1, Cls3 and Cls2, respectively. The remaining test patterns belong to Cls5. Let C_i represent the capacitance between the floating node and neighbor i, i = D, E or F. Let $V_{dd} = 1$.

Assume that for failing pattern P1, B = 0 and A stuck-at 1 (actually A having an incorrect logic value of 1) explains this pattern. Also for P1 let D = 0, E = F = 1. Then we have,

$$C_E + C_F + k - V_{thB0}C_{tot} > 0 \quad (11a)$$

For passing pattern P2, let B=0 and assume that an incorrect value of 1 on A would have been detected but in actuality no fault was detected. Thus $P2 \in Cls3$. Let in this D = E = 1, F = 0. We have:

$$C_D + C_E + k - V_{thB0}C_{tot} < 0 \tag{11b}$$

For failing pattern P3, let B = 1 and an incorrect value of 0 on A explains this pattern. Thus, $P3 \in Cls2$. Let D = 0, E = F = 1 for P3. We have:

$$C_E + C_F + k - V_{thB1}C_{tot} < 0 \tag{11c}$$

We have at least one solution for the three inequalities above with k= 1, $C_D = 2$, $C_E = 1$, $C_F = 3$, $C_{int0} + C_{gnd} = 1$, $C_{int1} + C_{vdd} = 1$, $V_{thB0} = 0.6$, and $V_{thB1} = 0.7$. $C_{tot} = C_D + C_E + C_F + C_{int0} + C_{gnd} + C_{int1} + C_{vdd} = 8$. This via will be included in the list of suspect vias reported by the procedure. Next an example of removing a candidate via is given.

Example 2: Consider the case of an open via driving input A of an AND gate with inputs A and B. Let the neighbors of node A be nodes D and E. To detect the faults on A, B has to be set to 1, and hence only one threshold voltage for input A need be used. Assume two test patterns, P1 a passing pattern, P2 a failing pattern belonging to Cls4 and Cls2 respectively, and let the remaining test patterns belong to Cls5. Let C_i represent the capacitance between the floating node and neighbor i, i = D or E. For passing pattern P1, assume that an incorrect value of 0 on A would have been detected but not actually detected. Thus $P1 \in Cls4$. For this input let D = 0, E = 1, C_{vdd}=1. We have:

$$C_E + k - V_{th}C_{tot} > 0 \tag{12a}$$

For failing pattern P2, let an incorrect value of 0 on A explains this pattern. Thus, $P2 \in Cls2$. Let D = E = 1 for P2. We have:

$$C_D + C_E + k - V_{th}C_{tot} < 0 \tag{12b}$$

It is easy to see that (12a) and (12b) are in conflict, and hence no solution satisfying the inequalities exists. The suspect via is dropped from the candidate list.

3.4 Open via driving multiple gates

For a via that drives multiple gates, in order to set up the inequalities discussed above, we use the results of multiple fault simulation performed during the segment fault diagnosis step described in Section 3.2. The main difference in the treatment of the general case of a via driving multiple gates and the special case of the via driving a single gate discussed above is the fact that now we need to consider faults on multiple gates which may mask some of the fault effects.

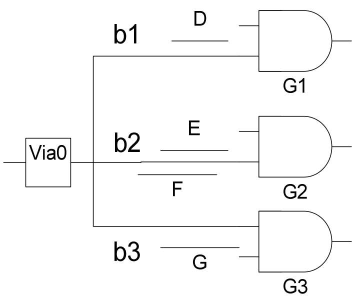

Figure 4: Example of via driving multiple gates

For example consider the circuit of Figure 4 showing a suspect via0 driving gates G1, G2 and G3. Let b1 stand for branch 1 of the segment driving G1, b2 for branch 2 driving G2, and b3 for branch 3 driving G3. Branch b1 has neighbor node D, branch b2 has neighbor nodes E and F and b3 has neighbor G. The open fault at via0 could create a Byzantine effect on the branches. Multiple faults on the branches may mask each other or a multiple fault is detected when a single fault is not detected. Each inequality is for one driven gate and involves all the neighbors of the interconnection section downstream of the via. We use bi/1 (bi/0) to denote branch bi stuck-at 1(0) fault. A multiple fault is denoted by a set of single faults. For example {b1/1, b2/1} is a double fault.

We evaluate each pattern to see if an inequality based on this pattern should be added to the set of inequalities to be used for each gate driven by the open via. If the pattern is a passing pattern, we determine whether a fault at the driven gate input can be detected to add an inequality. If the pattern is a failing pattern, we look for faults that explain this pattern or faults that will cause mismatches. That is, the fault will cause different failing bits when simulated than in the fail log. A brief description of the procedure we use is given next.

A. Passing patterns: If a passing pattern P can detect a fault on the driven input to gate Gi as determined by fault simulation and no multiple fault including the fault on Gi is *undetectable* then we conclude that the input to Gi must be at the fault free value when the passing pattern P is applied. Otherwise, the passing pattern P should have been a failing pattern. In this case we include an inequality for the gate as similar to the case of passing patterns in classes Cls3 and Cls4 discussed in Section 3.3.

B. Failing patterns: We need to consider two scenarios in deriving the inequalities for inputs of the driven gates using failing patterns. One is the set of faults on the inputs of the driven gates that explain the failing pattern and the other is the set of faults that cause mismatches.

B.1. Faults that explain the failing pattern

For a failing pattern P we determine all the multiple faults on the driven inputs of the gates that explain the observed outputs on the tester when P is applied to the chip being

1-4244-1128-9/07/$25.00 © 2007 IEEE

diagnosed. We use the simulation data saved in the step of logic diagnosis in Section 3.2. Next we determine the intersection of the sets corresponding to these faults and the faults in the intersection determine the gates for which we add an inequality similar to those for the patterns in classes Cls1 and Cls2 discussed in Section 3.3.

B.2. Faults that cause mismatches

If a driven gate, say G, is not included in the faults that explains failing pattern P, we can consider P to be a passing pattern for faults on G and add an inequality for G as done in A above if the following conditions hold. Fault on the input to G driven by the open via is detected by P in simulation but does not explain pattern P and every multiple fault containing the fault on G is detected by P but none of the faults can explain the fail log for P. It can be seen that these patterns are similar to the passing patterns in classes Cls3 and Cls4 discussed in Section 3.3.

After all the patterns are processed, we solve for the inequalities to check whether the suspect via is a valid candidate or not. We illustrate the general case discussed above using Example 3 given next.

Example 3: For the circuit illustrated in Figure 4, let the test pattern set be {P1, P2, P3} with P1 and P2 failing patterns and P3 a passing pattern.

Let the fault-free value under pattern P1 be 0. We use fault simulation results for faults: b1/1, b2/1, b3/1, {b1/1, b2/1}, {b1/1, b3/1}, {b2/1, b3/1} and {b1/1, b2/1, b3/1} and check which one of these faults explains pattern P1. Assume that only faults b1/1 and {b1/1, b2/1} explain this pattern. We choose the intersection of the fault combinations that explain the failing pattern which in this case is {b1/1}. Hence, $P1 \in Cls1$ with respect to branch b1. Suppose D = E = 1 and F = G = 0 for this pattern and let k_1 and V_{thG1} denote the k and V_{th} variables for branch b1 in (10). We add the following inequality for b1:

$$C_D + C_E + k_1 - V_{thG1} C_{tot} > 0 \tag{13a}$$

Next for P1 we look for the branch faults that cause a mismatch. We only consider the branch faults that do not belong to the faults explaining P1. In this case only b3/1 does not belong to any multiple fault that explains P1. Suppose b3/1 causes a mismatch and {b2/1, b3/1} is not detected by P1. Since {b2/1, b3/1} is a super set of b3/1 we cannot conclude that b3 is fault-free. For this reason no inequality is added for b3 corresponding to pattern P1.

Let the fault free value of the open node under the failing pattern P2 be 1 and assume that only the fault {b2/0, b3/0} explains P2. Suppose D = F = 1 and E = G = 0 under P2. We add the inequalities given below for b2 and b3, respectively:

$$\begin{cases} C_D + C_F + k_2 - V_{thG2} C_{tot} < 0 \\ C_D + C_F + k_3 - V_{thG3} C_{tot} < 0 \end{cases} \tag{13b}$$

For P2 b1/0 is not contained in the faults which explain P2. Suppose b1/0 is detected and causes a mismatch and {b1/0, b2/0}, {b1/0, b3/0}, and {b1/0, b2/0, b3/0} are all detected and cause mismatches. We conclude that b1 must be fault-free and we add the following inequality to b1:

$$C_D + C_F + k_1 - V_{thG1} C_{tot} > 0 \tag{13c}$$

For passing pattern P3, let the fault free value of the open node be 1 and assume that faults b1/0, b2/0, {b1/0, b2/0}, {b2/0, b3/0} and {b1/0, b2/0, b3/0} can be detected by P3. Since {b1/0, b3/0} is not detected and it is a super set of b1/0 as well as b3/0 no inequality for b1 or b3 is added based on P3. For b2 we add an inequality. If D = E = G = 1 and F = 0 for pattern P3 we add the following inequality to the set for b2:

$$C_D + C_E + C_G + k_2 - V_{thG2} C_{tot} > 0 \tag{13d}$$

We solve for inequalities (13a)-(13d) to check if this suspect via is a valid candidate.

4. Experimental results

In order to validate the proposed diagnosis procedure for opens, we conducted experiments on CMU benchmark circuit [23] and on ISCAS-85 combinational circuits. For each circuit a number of test cases were created with one open via defect injected into each test case. The responses of the defective circuit to tests in a set of single stuck-at fault detection test set were determined. We used SPICE simulation in evaluating the responses of the gates driven by the open net. A set of parameters are defined as follows to evaluate the quality of the diagnosis for the test cases of each circuit.

Exact: the number of test cases for which the set of candidate vias given by the diagnosis procedure has only the via in which an open defect was injected.

Contain: the number of test cases for which the set of candidate vias given by the diagnosis procedure included the via with the injected defect together with some other candidates.

None: the number of test cases for which the set of candidate vias given by the diagnosis procedure did not contain the via with the injected open.

Ave Net: average number of nets in the set of defect candidates reported by the diagnosis procedure.

Ave Via: average number of vias in the set of defect candidates reported by the diagnosis procedure.

The CMU benchmark [23] is a 4-bit ALU circuit fabricated with a 5-metal-layer TSMC 180nm CMOS technology [22]. For this benchmark, the circuit layout and the capacitances between two adjacent wires in the layout together with n-detection test sets for up to n = 5 are provided. We used 5-detection test sets and created fail logs for 300 random open defects using SPICE simulations. For ISCAS-85 benchmark circuits, the layouts and coupling capacitances between adjacent wires are obtained from a Texas A & M University website [24]. The ISCAS-85 benchmark circuits are also fabricated with a 5-metal-layer TSMC 180nm CMOS technology. For each injected defect, fail logs for 1-detect test sets for single line stuck-at faults were created using SPICE to simulate the sub circuit involving the open defect to get the voltage on the floating node. The voltage on the floating node was interpreted into logic values according to the threshold voltage of the driven gates. The gate level logic simulation is used to simulate the remaining circuit to get the failing and passing responses. During the creation of a test case, the initial trapped charge is randomly chosen from a range of -1 volt to +1 volt. For the purposes of comparison, we implemented five different diagnosis methods, LG, LGS, PHY, REF18 and PROP. LG is a gate level net list based logic diagnosis procedure that does not use physical information. LGS is an implementation of the segment fault model based diagnosis method of [17] which uses layout information to determine segments of nets. PHY is the diagnosis procedure in [19]. REF18 is an implementation of the diagnosis method of [18] reported in [19]. PROP is the proposed diagnosis method. The results using the five methods are reported in Table 1. For procedure REF18 two additional parameter values explained later are given. All methods used the same test sets.

The test cases for ISCAS-85 circuits are from the experiments used in [19] which also used the procedures LG and REF18 for comparison purposes and thus the results for LG, PHY and REF18 are produced by the implementation of the procedures in [19].

In Table 1, row 1 gives the circuit names. For each circuit diagnosis results using the five diagnosis procedures are given. From Table 1 it can be seen that the proposed method gives considerably smaller set of candidate defect nets and defect vias than the logic diagnosis method. It also gives considerably fewer candidates than the segment model based diagnosis procedure LGS. Compared to the segment method the additional data used by the proposed method is the neighbors of the nodes. In comparison to REF18 the proposed method does not drop the real defect site as REF18 procedure which typically drops 20% to 30% of the injected defects. The average number of nets and vias reported by the proposed method are some what higher than those reported by REF18. Since REF18 drops several real defect candidates we also computed the average numbers of candidate nets and vias reported by REF18 for the cases when the real defect via is not dropped as Ave Cnet and Ave Cvia, respectively. Both these numbers are mostly larger than Ave Net and Ave Via for REF18 over all test cases. Comparing the values of Ave Cnet and Ave Cvia for REF18 and the Ave Net and Ave Via for PROP we note that the proposed procedure

Table 1: Experiment Results

Methods	Circuits	c432	c499	c880	c1355	c1908	c2670	c3540	c5315	c6288	c7552	CMU
LG	Exact	0	0	0	0	0	0	0	0	0	0	0
	Contain	300	300	300	300	300	300	300	300	300	300	300
	None	0	0	0	0	0	0	0	0	0	0	0
	Ave Net	7.39	31.62	12.63	45.62	22.51	37.07	13.59	20.74	12.66	28.16	5.37
	Ave Via	53.98	214.66	81.57	372.53	140.58	242.37	102.07	184.22	77.84	234.2	39.64
LGS	Exact	0	0	1	0	1	0	0	0	0	0	0
	Contain	300	300	299	300	299	300	300	300	300	300	300
	None	0	0	0	0	0	0	0	0	0	0	0
	Ave Net	6.15	28.89	9.18	27.63	14.1	28.14	10.05	12.05	5.63	18.07	4.26
	Ave Via	22.07	111.08	30.38	92.25	49.22	103.97	38.04	44.63	17.17	68.33	15.45
PHY	Exact	11	8	4	6	16	10	4	15	3	21	22
	Contain	289	292	296	294	284	290	296	285	297	279	278
	None	0	0	0	0	0	0	0	0	0	0	0
	Ave Net	3	2.06	3.18	2.52	3.11	4.68	3.35	2.76	2.16	2.51	2.17
	Ave Via	10.66	7.36	10.98	8.63	10.98	17.44	12.78	9.84	7.02	9.04	7.64
REF18	Exact	0	1	1	0	2	0	2	3	2	1	0
	Contain	293	217	186	248	240	238	234	225	209	234	284
	None	7	82	113	52	58	62	64	72	89	65	16
	Ave Net	5.6	20.27	8.5	24.4	13.27	26.49	9.18	10.79	5	16.45	4.08
	Ave Via	13.28	31.75	15.95	39.88	22.49	41.37	17.52	17.97	8.78	24.78	10.62
	Ave Cnet	5.53	26.58	9.45	27.85	13.65	28.61	9.4	11.21	5.66	17.18	4.12
	Ave Cvia	13.3	41.95	17.45	45.42	23.45	45.87	18.49	19.51	10.33	26.3	10.75
PROP	Exact	0	0	1	0	1	0	1	1	0	0	0
	Contain	300	300	299	300	299	300	299	299	300	300	300
	None	0	0	0	0	0	0	0	0	0	0	0
	Ave Net	5.69	20.41	8.62	24.49	13.33	26.62	9.27	10.9	5.14	16.51	4.09
	Ave Via	14.2	34.05	18.82	41.62	24.45	45.68	20.59	20.97	11.01	27.7	11.5

reports comparable numbers of candidate nets and vias. Procedure PHY returns fewer suspect candidates than the proposed procedure. However procedure PHY requires extracted capacitances as well as threshold voltages of library cells. Reasonably accurate values of these required parameters may not be forthcoming in nanometer designs.

In order to illustrate the potential effect of inaccuracies in extracted capacitance values on the performance of procedures that depend on the extracted capacitances we performed an experiment using the CMU benchmark. The coupling capacitances to a floating nodes used by the tools REF18 and PHY were changed to random values over different ranges from the actual values in the benchmark. For example let C be the value of coupling capacitance in the benchmark and we set the capacitance used by REF18 and PHY to be different and over a range of 0.5C and 2.0C. In this case we randomly pick a different value in the range (0.5C, 2.0C) for the capacitances for different instances of defective chips. In choosing random values for capacitances and threshold voltages we used uniform distribution of values over the specified ranges. For procedure PHY the threshold voltages used were also randomly varied by ±15%. We used the 300 fail logs used in Table 1 and report the results in Table 2. In Table 2 under C we report the results when the capacitance values used by Procedures REF18 and PHY are the same as the ones used in the CMU benchmark. Next three columns give the results when the capacitances used by the procedures REF18 and PHY are randomly set over the range indicated in the column headings. We also computed the average numbers of candidate nets and vias reported when the real defect via is not dropped as Ave Cnet and Ave Cvia, respectively.

Table 2: Inaccurate Neighbor Capacitances

		C	0.5C-2C	0.67C-1.5C	0.75C-1.25C
REF18	Exact	0	0	0	0
	Contain	284	272	277	282
	None	16	28	23	18
	Ave Net	4.08	4.09	4.09	4.10
	Ave Via	10.62	10.59	10.64	10.66
	Ave Cnet	4.12	4.22	4.18	4.14
	Ave Cvia	10.75	11.02	10.92	10.81
PHY	Exact	22	17	22	18
	Contain	278	254	260	261
	None	0	29	18	21
	Ave Net	2.17	2.06	2.12	2.11
	Ave Via	7.64	7.38	7.47	7.45
	Ave Cnet	2.08	2.27	2.24	2.4
	Ave Cvia	7.15	8.15	7.93	8.54
PROP	Exact	0	0	0	0
	Contain	300	300	300	300
	None	0	0	0	0
	Ave Net	4.09	4.09	4.09	4.09
	Ave Via	11.5	11.5	11.5	11.5

From Table 2 it can be noted that procedures REF18 and PHY drop several actual defect sites from the reported candidate lists as noted in the row None. Since the proposed method does not use capacitance and threshold values, the number of None cases remains 0.

5. Discussion

For the experimental results presented in the paper we used fault detection test sets for single line stuck-at faults. We assumed that the test results are available for all the tests in the test sets. In the future we plan to investigate the effectiveness of the proposed method when only a limited set of failing patterns instead of results for all tests are available. We also plan to investigate if using other test sets, such as diagnosis test sets that resolve all resolvable pairs of stuck-at faults or n-detection test sets, will lead to better diagnosis resolution.

In this work we assumed that the coupling capacitances and threshold voltages gates are completely unknown. This requires considering (2N – 1) combinations of multiple stuck-at faults when a candidate via drives N gates. One can reduce the number of faults considered to (N - 1) if it is assumed that the threshold voltages of different driven gates change from the nominal values in a correlated manner. One may also obtain improved diagnosis results if it can be assumed that the coupling capacitances vary from nominal values within some bounds.

In this work we only considered complete opens of vias. Considering resistive opens using the proposed framework will require use of tests for delay faults and modeling the effects of opens on signal propagation through the defect sites. Such a study will be part of our future work.

6. Conclusions

A new interconnect open defect diagnosis method using less physical information is proposed. Specifically the method does not require the values of inter node capacitances and gate threshold voltages which are difficult if not impossible to determine for manufactured instances of nanometer designs. Experiments conducted on benchmark circuits validated the effectiveness of the proposed method.

References:

[1] R. Rodriguez-Montanes and J. Figueras, “Electrical and Topological Characterization of Interconnect Open Defects”, IEEE international Workshop on Current and Defective Based Testing, 2005.

[2] H. Konuk, “Fault Simulation of Interconnect Opens in Digital CMOS Circuits”, in Proc. ICCAD, 1997, pp. 548-554.

[3] S. Rafiq, A. Ivanov, S. Tabatabaei, and M. Renovell, “Testing for Floating Gates Defects in CMOS Circuits”, in Proc ATS 1998, pp.228-236.

[4] D. Arumi, R. Rodriguez-Montanes and J. Figueras, “Defective Behaviours of Resistive Opens in Interconnect Lines”, Proc. of ETS 2005.

[5] D. B. Lavo, T. Larabee, and B. Chess, “Beyond the Byzantine Generals: Unexpected Behavior and Bridging Fault Diagnosis”, in Proc. ITC 1996, pp 611-619.

[6] S. Venkatarman and S. B. Drummonds, “A Technique for Logic Fault Diagnosis of Interconnect Open Defects”, in Proc. VTS 2000, pp. 313-318.

[7] S. Venkatarman and S. B. Drummonds, “Poirot: A Logic Fault Diagnosis Too and its Applications”, in Proc. ITC 2000, pp. 253-262.

[8] S.-Y. Huang, “Speeding Up the Byzantine Fault Diagnosis Using Symbolic Simulations”, in Proc. VTS 2003, pp. 193-198.

[9] X. Wen, T. Miyoshi, S. Kajihara, L-T. Wang, K. K Saluja and K. Kinoshita, “On Per-Test Fault Diagnosis Using the X-Fault Model”, in Proc. ICCAD 2004, pp. 633-640.

[10] J. B. Liu, A. Veneris and H. Takahashi, “Incremental Diagnosis of Multiple Open-Interconnects”, in Proc. ITC 2002, pp. 1085-1092.

[11] J. Waicukauski and E. Lindbloom, “Failure Diagnosis of Structured VLSI”, IEEE Design and Test of Computer, vol. 6, no. 4, 1989, pp.49-60.

[12] T. Bartenstein, D. Heaberlin, L. Huisman and D. Sliwinski, “Diagnosing Combinational Logic Designs Using the Single Location At-a-Time (SLAT) Paradigm”, in Proc. ITC 2001, pp. 287-296.

[13] Z. Wang, K-H. Tsai, M. M. Sadowska, and J. Rajski, “An Efficient and Effective Methodology on the Multiple Fault Diagnosis”, in Proc. ITC 2003, pp. 329-338.

[14] D. B. Lavo, I. Hartanto, and T. Larrabe, “Multiplets, Models, and the Search for Meaning: Improving Per-Test Fault Diagnosis,” in Proc. ITC 2002, pp. 250-259.

[15] X. Wen, H. Tamamoto, K. K. Saluja and K. Kinoshita, “Fault diagnosis for physical defects of unknown behaviors”, in Proc. ATS 2003, pp. 236-241.

[16] W. Zou, W.-T. Cheng and S.M. Reddy, “On Methods to Improve Location Based Logic Diagnosis,” Proc. VLSI Design 2006.

[17] S. Y. Huang, “Diagnosis of Byzantine Open-Segment Faults”, in Proc. ATS 2002, pp. 248-253.

[18] Y. Sato, I. Yamazaki, H. Yamanaka, T. Ikeda and M. Takakura, “A Persistent Diagnosis Technique for Unstable Defects”, in Proc. ITC2002, pp. 242-249.

[19] W. Zou, W.-T. Cheng, S. M. Reddy, “Interconnect Open Defect Diagnosis with Physical Information”, in Proc. ATS 2006.

[20] S. M. Reddy, I. Pomeranz, H. Tang, S. Kajihara and K. Kinoshita, “On Testing of Interconnect Open Defects in Combinational Logic Circuits with Stems of Large Fanout”, Proc. ITC 2002, pp. 83-89.

[21] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*, Second Edition. MIT Press and McGraw-Hill, 2001. Section 29.3: The simplex algorithm, pp.790–804.

[22] The MOSIS Website, http://www.mosis.org

[23] T. Vogels, T. Zanon, E. Desineni, S. Blanton, W. Maly, et al., “Benchmarking Diagnosis Algorithms with a Diverse Set of IC Deformations”, in Proc. ITC 2004, pp 508-517.

[24] The TAMU Website, http://ece.tamu.edu/~xiang/iscas.html