The analysis of spot defect induced faults in MOS circuits

China 1991 International Conference on Circuits and Systems, June 1991, Shenzhen, China

The Analysis of Spot Defect Induced Faults in MOS Circuits

Chennian Di and Jose Pineda de Gyvez

Department of Electrical Engineering
Eindhoven University of Technology
The Netherlands
Email di@es.ele.tue.nl

Abstract

A strategy for modeling spot defect induced faults by their corresponding boolean functions is developed. The presented strategy is based on the principle of local conduction path analysis. This way of modeling is much more general in the sense that all kinds of faults are unified by one concept, the boolean function. In this way testing related applications can be done efficiently and can maintain a high quality.

I Introduction

For the dominating MOS technology, it is known that faults induced by spot defects can no longer be modeled as conventional stuck–at faults. Moreover, the occurrence of a fault depends on the layout and on some defect conditions[1,2,3]. Many new fault models are suggested in order to supply a high quality test set, e.g. transistor stuck open/close faults, bridges, and opens in complex gates. However, such new fault classes are rather arbitrary and heuristic [4]. To be more accurate and realistic, some methods presented in [4,5,6] suggest a dynamical modeling by extracting the faults from the physical layout of a design. The defects, which are conceptualized as extra and missing materials, are then systematically abstracted as node bridges and open line at the circuit level. But the large number of extracted faults and the variety of fault types may make it impractical to derive an efficient test strategy. Most of the test methods developed so far deal with each fault class separately [7,8,9,10,11 ,12,13]. Moreover, most of these methods derive the tests based only on the circuit topology for which a circuit level fault simulation has to be performed in order to validate the tests[10,12]. Definitely, this procedure is quite costly and inefficient. Therefore, it is essential to know the exact logic effects of defect induced faults and to model them at higher levels, e.g. logic level, before the test is generated. This paper concentrates on the logic analysis of defect induced faults for nMOS combinational circuits. Accurate logic effects of each fault are obtained by local circuit analysis. This analysis is achieved by only searching the faulty paths and, at the meantime, by taking into account the circuit parameters. From the analysis, each fault can be represented by its corresponding boolean function. By doing so, all different faults are accurately modeled by one concept, the boolean function. Therefore, testing related applications can be done at a high level. For instance, the faults can be collapsed by boolean manipulations, and the test patterns can be generated in a rather general manner by satisfying the boolean functions. Therefore the potential high quality and efficiency can be maintained.

II Preliminaries

The analyzed nMOS combinational circuit can be viewed as interconnecting nMOS *blocks*. Each block consists of a single depletion transistor as the pull–up part, and serial–parallel connected enhancement transistors as the pull–down part. The node where these two parts are joined together is referred to as *block output node*. Each pull–down path is defined as a *conduction path* of the block, i.e. a series of connected transistors beginning from a block output node and ending in a Vss node.

For the purpose of the analysis, the nodes are classified into three types.

+ *I* (input node):
  all those nodes representing a primary input, *Vdd*, and *Vss* are defined as *I* node.
+ *L* (logic node):
  any node representing a block output node is defined as a *L* node.
+ *nL* (non–logic node):
  any node where just enhancement transistors' drain (source) are connected together is defined as a *nL* node.

As mentioned before, the faults induced by defects are highly dependent on circuit layout styles and on defect conditions. To be general, it is assumed that a defect may occur anywhere in the layout and consequently all kinds of circuit faults can be induced. In this paper, the analysis is restricted only to catastrophic defects which cause either two different nodes to be bridged or a single line to be opened.

Referring to the previous node classification, the following types of bridges can be encountered: *I–L*, *I–nL*, *nL–nL*, *L–L*, and *nL–L*. That is, all possible bridges between different types of nodes. The open faults can happen on a *nL* node, a *L* node and in one of the terminals of a depletion transistor. For obvious reasons, the faults which involve primary inputs (except *Vdd(Vss)–nL*, *Vdd(Vss)–L* bridges) will not be analyzed. Further, only the static behavior is considered without dynamic analysis, the timing feature of each fault is not essential here.

It is known that when a defect is present various possible pull–up to pull–down resistances can be formed in a wrongly connected circuit[10]. If such a wrongly connected path needs to be activated, there may be an intermediate value between *Vdd* and *Vss* produced at output. This value can be interpreted as a logic 1 or 0. If the exact value of an intermediate output voltage can be computed and the logic threshold voltage of each block is known as well, then such a value can easily be interpreted as a logic 1 when it is bigger than the logic threshold voltage of the fanout blocks and, otherwise, as a logic 0. However, in practice both computations are not easy tasks. First of all, it is hard to compute an intermediate output voltage, and in some situations it is almost impossible to know the exact value without running a circuit simulation. For instance, if a path contains a transistor with the gate bridged to the drain, it is difficult to predict the output voltage value. Imagine, both depletion transistor and enhancement transistors having all possible sizes and also the body effects becoming not a negligible factor. Secondly, the logic threshold voltage of each block is not a constant value. For a complex block its value varies in a range determined by the way in which the block is driven [14]. This situation makes the exact fault modeling at logic level very difficult.

Fortunately, several facts may make it still feasible. First of all, most of the bridge faults do create some unnecessary conduction paths, called *faulty paths* here. By using a simplified MOS transistor model, the voltage value at the outputs

which results from these faulty paths can be predicted with sufficient accuracy, and furthermore, the computation can significantly be reduced. The proposed simplified MOS model is taken under the assumption that all the depletion transistors are in the saturated region ($|V_{dep}| < Vdd - V_{ds}$) and that all the enhancement transistors are in the linear region ($V_{gs} - V_{th} > V_{ds}$), where V_{dep} and V_{th} are the threshold voltage of a depletion transistor and an enhancement transistor respectively, and V_{ds}, V_{gs} are the drain–source and gate–source voltages. Furthermore, the body effect is also neglected. Obviously, each depletion transistor is modeled as a current source, and each enhancement transistor as a resistor. The node voltage can easily be computed by solving linear equations for the selected paths.

Secondly, in most designs, though the size of the depletion transistor in each block can be different, the enhancement transistors in each block are usually sized in such a way that the equivalent pull–up to pull–down beta ratio of each path is the same. This beta ratio is also known as the *beta ratio of the design*. Another fact is that during testing usually only one conduction path is assumed to be active in order to propagate a fault. These two facts suggest that, for each block, the *defined logic threshold voltage* can be used for comparison. Moreover, the defined logic threshold voltage is as the one of an inverter with an equivalent pull–up to pull–down beta ratio. In this paper, the notation V_{logic} is used for the defined logic threshold voltage of a design. In addition, for a correctly designed circuit the gain is usually very high near the region V_{logic} in the d–c transfer characteristic of each block. Under this assumption, an intermediate voltage input value can quickly converge to a value near either *Vdd* or *Vss* after it is propagated through several blocks. Eventually, after several blocks this voltage behaves as a logic 1 or 0. All of these facts make the fault modeling at the logic level be practical.

In the next sections, it will be shown how the faulty paths can properly be chosen for each type of fault and how their logic level can be determined by the following criterion:

Under certain input conditions, if the fault free output value is a logic 0(1) and the faulty voltage of the output turns to be a value bigger(smaller) than V_{logic}, then the related fault is 0(1) detectable. The faulty value is expressed by its opposite value, logic 1(0).

In the figures that are used for illustrations, each box represents a serial–parallel connected subcircuit of enhancement transistors. The subfunction of each box (sum of products of each serial–parallel connected enhancement transistor path) is represented by a boolean variable inside each box. The non–logic nodes are labeled by lower case letters (see *a, b* in Fig.1a). The logic nodes are represented by upper case letters which are also referred to as boolean variables representing the boolean function of the corresponding blocks. As an example, consider Fig.1a. The fault–free boolean function of block *F* expressed in a complemented sum of products form is $F = \overline{f_1 \cdot G \cdot f_2 + f_3}$. Each bridge is indicated by a dashed line.

III Logic Manipulation of Bridges

3.1 Logic effects of I–L bridges

For this type of bridge, only the *Vdd(Vss)–L* type is analyzed. Needless to say, their effects can be analyzed without searching any faulty paths. Assume the bridges between *Vdd(Vss)* and *G* in Fig.1a. Let us represent them as <*Vdd, G*> and <*Vss, G*>, respectively. Obviously, <*Vdd, G*> will cause *T* to be on, and <*Vss, G*> will force *T* to be off all the time. The faulty boolean functions are obtained simply by assigning a logic 1(0) to the boolean variable *G* for the case of <*Vdd, G*> (<*Vss, G*>). Consequently, *F* becomes

$$F = \begin{cases} \overline{f_1 \cdot f_2 + f_3}, & \text{if} < Vdd, G > \quad \text{(a1)} \\ \overline{f_3}, & \text{if} < Vss, G > \quad \text{(a2)} \end{cases}$$

3.2 Logic effects of I–nL bridges

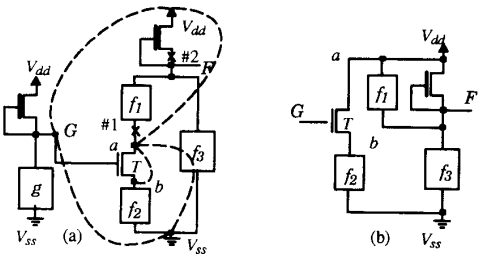

Figure 1 *Vdd(Vss)–L, Vdd(Vss)–nL* and *nL–nL* type bridges

Only the *Vdd(Vss)–nL* type is considered. Assume now two bridges to be <*Vss, a*> and <*Vdd, a*> as in Fig.1a. Consider first the bridge <*Vss, a*>. Since node *a* is directly connected to *Vss* the states of all the transistors between *a* and *Vss* will not affect *F*. So, if $f_1 = 1$, *F* will be driven to a logic 0 level. Therefore the faulty function is obtained as

$$F = \overline{f_1 + f_3} \quad \text{(b1)}$$

Consider now <*Vdd, a*>. The block *F* will be wrongly connected as it is shown in Fig.1b. It can be observed that the states of *T* and all the transistors from f_2 will not influence the functionality of the logic node *F*. If $f_3 = 0$ and $f_1 \cdot G \cdot f_2 = 1$, the logic node *F* will be driven to *Vdd*. Therefore, a logic 0 detectable condition arises. If $f_1 = 1$ and $f_3 = 1$, it is difficult to determine the voltage value at *F* since the body effects cannot be neglected for the transistors in f_1. It is expensive to compute the exact output voltage under the condition that the resistance of path *Vdd–>F–>(via f_3)–>Vss* may have various possible values. To avoid the uncertainty in the derived faulty function and to maintain a manageable computation during the analysis, the circuit is assumed to function correctly. As a result, *F* will be driven to a logic 0. Thus, the derived faulty function is the same as eq.(a2). Though the derived faulty function is incomplete, it is sufficient to model the logic effects by setting the path *Vdd–>F–>a–>b–>Vss* off all the time.

3.3 Logic effects of nL–nL bridges

A *nL–nL* bridge may occur in two different ways. First, two non–logic nodes are located on the same path. This type of bridge is illustrated by the bridge <*a, b*> in Fig.1a. Since there is a direct path from *a* to *b*, the state of *T* does not affect the output *F*. So the faulty function is easily derived as

$$F = \overline{f_1 \cdot f_2 + f_3} \quad \text{(c1)}$$

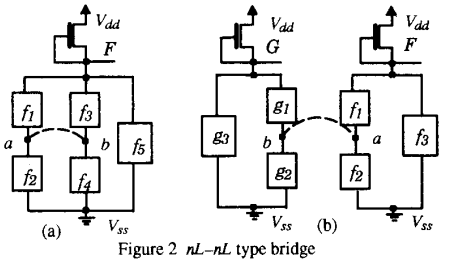

Figure 2 *nL–nL* type bridge

Now assume that two non–logic nodes are located on different paths. This situation is illustrated by the bridge <*a, b*> in Fig.2. Fig.2a depicts the case where the involved paths belong to the same block, and Fig.2b depicts the case where the involved paths belong to different blocks. The faulty paths created in Fig.2a are *Vdd–>F–>b–>a–>Vss* and *Vdd–>F–>a–>b–>Vss*. In Fig.2b, the faulty paths are *Vdd–>F–>a–>b–>Vss* and *Vdd–>G–>b–>a–>Vss*. The only way to expose these two bridges is by setting one faulty path on while turning all the other paths including the rest of the faulty paths off. In our approach, the output voltage is computed for each faulty path to determine its logic level. There may be many possible results. One of them is that, in both cases, all the faulty paths may cause the output voltages $V_F < V_{logic}$ and $V_G < V_{logic}$, which give a logic 1 detectable condition. Then the faulty functions can be expressed as

$$F = \overline{f_1 \cdot f_2 + f_3 \cdot f_4 + f_5 + X} \quad \text{(c2)}$$

for bridge in Fig.2a, where $X = f_1 \cdot f_4 + f_3 \cdot f_2$, and as

$$\begin{cases} F = \overline{f_1 \cdot f_2 + f_3 + X} & \text{(c3)} \\ G = \overline{g_1 \cdot g_2 + g_3 + Y} & \text{(c4)} \end{cases}$$

for bridge in Fig.2b, where $X = f_1 \cdot g_2$ and $Y = g_1 \cdot f_2$.

For the bridge of Fig.2b, another possible result may be that the path *Vdd->F->a->b->Vss* causes the output voltage $V_F > V_{logic}$, and that the path *Vdd->G->b->a->Vss* causes the output voltage $V_G < V_{logic}$. These voltages may result from the possible transistor sizes between different blocks. The resistance of a pull-down faulty path may be so big that it causes the output voltage $V_F > V_{logic}$. For this order, the faulty function will be

$$\begin{cases} F = \overline{f_1 \cdot f_2 + f_3} & \text{(c5)} \\ G = \overline{g_1 \cdot g_2 + g_3 + Y} & \text{(c6)} \end{cases}$$

That is, F remains the same as the fault free situation, and G results in eq.(c4).

For other possible different results, the faulty functions can be derived similarly.

3.4 Logic effects of L-L bridges

The analysis is conducted for two different cases. Firstly, one bridged logic node will not directly fanout to another block where the bridge occurs. This situation is illustrated by the bridge <*F, G*> in Fig.3a. Secondly, one bridged logic node does fanout to another block where the bridge occurs. This type of bridge is depicted in Fig.3b. In both cases, F and G always have the same voltage value, V_{bridge}, under the bridge condition. If F and G are assumed to have the same logic level, the bridge <*F, G*> will not cause a malfunction in both cases. Only when F and G are supposed to have opposite logic levels a logic 1 detectable condition may be established by one of the logic nodes through a faulty path. This can be determined by computing the output voltage for each faulty path. For the bridge in Fig.3a, the faulty paths are *Vdd->G->F->(via f)->Vss* and *Vdd->F->G->(via g)->Vss*. For the same reasons as for the bridge in Fig.2b, one possible result is that all the faulty paths result in an output voltage $V_{bridge} < V_{logic}$. Thus the faulty function is derived as

$$F = G = \overline{f + g} \quad \text{(d1)}$$

For the same reason mentioned when deriving eq.(c5) and eq.(c6), the second possible result can be that the faulty path *Vdd->G->F->(via f)->Vss* causes an output voltage $V_{bridge} > V_{logic}$. But the faulty path *Vdd->F->G->(via g)->Vss* results in an output voltage $V_{bridge} < V_{logic}$. Then the faulty function will be

$$F = G = \overline{g} \quad \text{(d2)}$$

Other possible faulty functions can be derived accordingly.

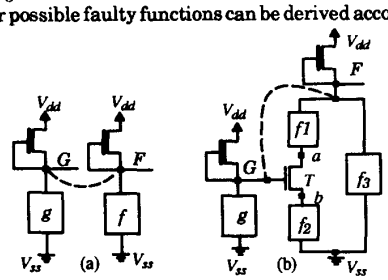

Figure 3 *L-L type bridge*

Now consider the bridge <*G, F*> in Fig.3b. It is hard to predict the output voltage value of F when only the faulty path *Vdd->F->a->b->Vss* is supposed to be on, i.e., $g = f_3 = 0$ and $f_1 = f_2 = 1$. The difficulty comes from the various possible transistor sizes and also because T does not work in the linear region anymore. From results of SPICE simulations, it was seen that the output voltage can slightly be bigger or smaller than V_{logic} when the number of transistors and their sizes are changed in such a path. To cope with such a situation, the strategy used here is that the faulty paths are chosen in such a way that the output voltage can still be computed by using the simplified model. For the bridge in Fig.3b, the faulty paths chosen are *Vdd->F->G->(via g)->Vss* and *Vdd->G->F->(via f3)->Vss*. If path *Vdd->F->a->b->Vss* is supposed to be on, the output logic level is modeled as if it functions correctly. As a result, F is driven to a logic 0. By doing so, the 'real' logic effects of the fault are reflected in the derived faulty function without any uncertainty. At the meantime, the expensive computation is avoided. Surely, the modeled faulty function F is not complete. However, other faulty paths suffice to reflect the logic effects of the fault in the faulty functions.

Similar to the previous case, one possible result is that all the chosen faulty paths cause the output voltage $V_{bridge} < V_{logic}$. Then the faulty function can be expressed as

$$\begin{cases} F = \overline{f_1 \cdot f_2 + f_3 + g} & \text{(d3)} \\ G = \overline{g + f_3} & \text{(d4)} \end{cases}$$

For other possible results, the faulty functions can be derived accordingly.

3.5 Logic effects of nL-L bridges

A *nL-L* bridge is analyzed also for two situations. In the first situation, two faulty nodes are located on the same path. This type of bridge is illustrated in Fig.4. The relatively simple bridge is <*G, d*>, see Fig.4. Even if the transistors between c and d are on, the voltage drop between the gate and the source of T can never be bigger than 0. Therefore, T is off all the time and consequently the path *Vdd->F->a->b->c->d->Vss* is off as well. As one possibility, the faulty path *Vdd->G->b->Vss* may cause the output voltage $V_G < V_{logic}$. There, a logic 1 detectable condition is established if G fanouts to other blocks. So, the faulty functions can be expressed as

$$\begin{cases} F = \overline{f_5} & \text{(e1)} \\ G = \overline{g + f_4} & \text{(e2)} \end{cases}$$

The same strategy which is used for the analysis of the bridge <*G, F*> in Fig.3b is applied to analyze the bridge <*G, a*> of Fig.4. Following this strategy, the output logic level is modeled as a logic 0 when only the path *Vdd->F->a->b->c->d->Vss* is supposed to be on. That is, this path is modeled as if it functions correctly. The faulty paths chosen are *Vdd->F->a->G->(via g)->Vss* and *Vdd->G->a->F->(via f5)->Vss*. Their logic effects can be determined by comparing the computed output voltages V_G and V_F to V_{logic}.

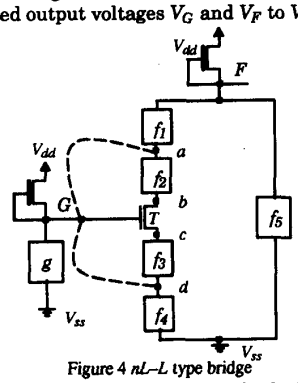

Figure 4 *nL-L type bridge*

Again, one possible result is that all the faulty paths cause both output voltages $V_G < V_{logic}$ and $V_F < V_{logic}$. Thus there will a logic 1 detectable condition formed. The faulty functions are derived as

$$\begin{cases} F = \overline{f_1 \cdot g + f_1 \cdot f_2 \cdot f_3 \cdot f_4 + f_5} & \text{(e3)} \\ G = \overline{g + f_1 \cdot f_5} & \text{(e4)} \end{cases}$$

Other possible faulty functions can be derived similarly.

Now consider the second situation. Two faulty nodes are located on different paths. This type of bridge is illustrated in Fig.5. Fig.5a depicts the case where the two faulty nodes belong to different blocks. Fig.5b depicts the case where two

faulty nodes belong to the same block.

The faulty paths for the bridge <*G, a*> of Fig.5a are *Vdd->G->a->Vss* and *Vdd->F->a->G->Vss*.

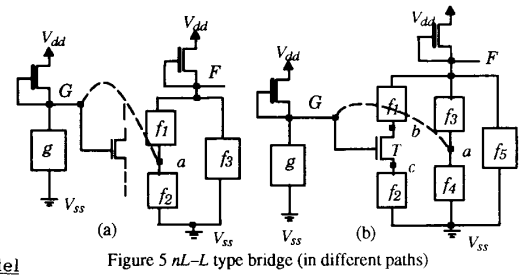

Figure 5 *nL–L* type bridge (in different paths)

t model

The analysis of faulty paths follows the same principle as it is done in the previous cases. One result among all possible results is that all faulty paths cause the output voltages $V_G<V_{logic}$ and $V_F<V_{logic}$. Then a logic 1 detectable condition is established. Consequently, the faulty function is expressed as

$$\begin{cases} F = \overline{f_1 \cdot g + f_1 \cdot f_2 + f_3} & \text{(e5)} \\ G = \overline{g + f_2} & \text{(e6)} \end{cases}$$

Similarly to the previous cases, other possible faulty functions can be derived accordingly.

To analyze the bridge <*G, a*> in Fig.5b, the same strategy which is used for the analysis of the bridge <*G, F*> in Fig.3b is also applied. That is, for $f_3 = 1$, if only path *Vdd->F->b->c->Vss* is supposed to be on, both outputs *G* and *F* are considered to be working correctly. Thus, *F* has a logic 0 value and *G* has a logic 1 value. The faulty paths chosen are *Vdd->G->a->Vss* and *Vdd->F->a->G->Vss*. One possible result will be that all the faulty paths cause both output voltages $V_G<V_{logic}$ and $V_F<V_{logic}$. In this case, if the faulty path *Vdd->G->a->Vss* is activated, that is $f_1 \cdot f_2 = 1$ and $g = f_3 = f_5 = 0$, then *F* will be driven to *Vdd*. This is because *G* is driven to a logic 0 through the faulty path and consequently the path *Vdd->F->b->c->Vss* is off. Therefore, a logic 0 detectable condition is established. The faulty functions are then derived as

$$\begin{cases} F = \overline{f_1 \cdot f_2 \cdot \bar{g} \cdot \bar{f_4} + f_3 \cdot g + f_3 \cdot f_4 + f_5} & \text{(e7)} \\ G = \overline{g + f_4} & \text{(e8)} \end{cases}$$

It can be noticed that the derived faulty functions for bridges in both Fig.5a and Fig.5b are not complete since the logic effects of some faulty paths are not included. However, as it is stated during the analysis of the bridge in Fig.3b, that the faulty paths chosen are sufficient to model the logic behavior of each fault.

All the analysis for these bridges have been verified extensively by SPICE circuit simulation and the results match the logic modeling results presented here. Therefore, the logic representations developed are quite accurate.

IV Logic Manipulation of Opens

For the fatal opens, their logic effects are analyzed under three different types.

4.1 Logic effects of an open on a nL node

This type of open may occur on any *nL* node. Obviously, it will cause the related path to be off all the time. Assume the open #1 on node *a* of Fig.1a. Since the path *Vdd->F->a->b->Vss* is off all the time, the resulting faulty function is the same as eq.(a2).

4.2 Logic effects of an open on a L node

If an open occurs on a logic node, the transistor gates to which this logic node fans in are floating. From the analysis in [10], it is known that if the open is permanent, then the stored charge at the floating gates will eventually leak away through the leakage path at the gate terminal to the substrate. The voltage of the floating gates will stay at the level of the substrate bias. Therefore, this type of open can be modeled as a logic node stuck at 0.

4.3 Logic effects of an open terminal in a depletion transistor

The effects of this type of open are the same as the ones of the open in section 4.2 because they cause the related output node to be floating. From the analysis, this type of open can also be modeled as the corresponding logic node stuck at 0. Therefore, the faulty function can be derived as it is done for a *Vss–L* bridge. For instance, if the open #2 of Fig.1b occurs, the faulty function can simply be expressed as $F = 0$.

V Conclusions

In this paper the logic analysis for defect induced faults in nMOS circuits is performed for a broader range of fault types. It can be observed that different faults can potentially have different effects. These effects not only depend on the circuit interconnections but also on the circuit parameters, e.g. the different transistor sizes. Therefore, the local path analysis can lead to a more accurate modeling. Since it is hard to map these faults to the corresponding stuck at faults which are the fault models used in practice, the faulty boolean functions do play a good alternative role for dynamic fault modeling. Though the analysis is done just for nMOS combinational circuits, it is very easy to extend the results to CMOS combinational circuits following the same concepts.

VI References

[1] Galiay, J., Y. Crouzet, and M. Vergniault, "Physical Versus Logical Fault Models MOS LSI Circuits: Impact on Their Testability," *IEEE Trans. on computers*, vol. c–29, no. 6, pp. 527–531, June, 1980.

[2] Wadsack, R.L., "Fault Modeling and Logic Simulation of CMOS and MOS Integrated Circuits," *The Bell System Technical Journal*, vol. 57, no. 5, pp. 1449–1474, May–June, 1978.

[3] Maly, W., "Realistic Fault Modeling for VLSI Testing," *Proc. of 24th Design Automatic Conference*, pp. 173–180, 1987.

[4] Ferguson, F.J. and J.P. Shen, "A CMOS Fault Extractor for Inductive Fault Analysis," *IEEE Trans. on Computer-Aided Design*, vol. CAD–7, no. 11, pp. 1181–1194, November 1988.

[5] Maly, W., F.J. Ferguson, and J.P. Shen, "Systematic Characterization of Physical Defects for Fault Analysis of MOS IC Cells," *Proc. of 15th Int. Test Conf., IEEE*, pp. 390–399, 1984.

[6] Gyvez, J.P. de and C. Di, "IC Defect– Sensitivity for Footprint Type Spot Defects," *IEEE Trans. on Computer-Aided Design*, Accpedted for publication, 1990.

[7] Lamoureux, P. and V.K. Agarwal, "Non–Stuck–at Fault Detection in NMOS Circuits by Region Analysis," *Proc. of Int. Test Conference*, pp. 129–137, 1983.

[8] Chiang, K–W and Z.G. Vranesic, "Test Generation for MOS Complex Gate Network," *Proc. of FTCS– 12*, pp. 149–157, 1982.

[9] Jain, S.K. and V.D. Agrawal, "Test Generation for MOS Circuits Using D–Algorithm," *Proc. of 20th Design Automation Conference*, pp. 64–70, 1983.

[10] Banerjee, P. and J.A. Abraham, "Characterization and Testing of Physical failures in MOS Logic Circuits," *IEEE Design & Test*, pp. 76–86, August, 1984.

[11] Reddy, M.K., S.K. Reddy, and P. Agrawal, "Transistor Level Test Generation for CMOS Circuit," *Proc. of 22nd Design Automation Conference*, pp. 825–828, 1985.

[12] Shih, H. and J.A. Abraham, "Transistor– Level Test Generation for Physical Failures in CMOS Circuits," *Proc. of 23rd Design Automation Conference*, pp. 243– 249, 1986.

[13] Reddy, S.M., V.D. Agrawal, and S.K. Jain, "A Gate Level Model for CMOS Combinational Logic Circuits with Application to Fault Detection ," *Proc. of 21th Design Automation Conference*, pp. 504–509, 1984.

[14] Mukherjee, A., "Introduction to nMOS and CMOS VLSI Systems Design," *Prentice-Hall International Editions*, 1986.