Test Set Enrichment using a Probabilistic Fault Model and the Theory of Output Deviations

Zhanglei Wang†, Krishnendu Chakrabarty†*, and Michael Goessel‡
†Dept. Electrical & Computer Engineering
Duke University, Durham, NC 27708, USA
E-mail: {zw8,krish}@ee.duke.edu
‡Institute of Computer Science
University of Potsdam, Germany
E-mail: mggoessel@cs.uni-potsdam.de

Abstract—We present a probabilistic fault model that allows any number of gates in an integrated circuit to fail probabilistically. Tests for this fault model, determined using the theory of output deviations, can be used to supplement tests for classical fault models, thereby increasing test quality and reducing the probability of test escape. Output deviations can also be used for test selection, whereby the most effective test patterns can be selected from large test sets during time-constrained and high-volume production testing. Experimental results are presented to evaluate the effectiveness of patterns with high output deviations for the single stuck-at and bridging fault models.

I. INTRODUCTION

Due to shrinking process technologies and the use of new materials such as copper and high-K dielectrics for manufacturing, defects for integrated circuits (ICs) are becoming increasingly complex [1], [2]. It is not always possible to accurately model these defects or predict their occurrences. A single spot defect is likely to affect several logic gates and multiple defects on a die are likely. The locations of these defects are unpredictable. The failure probability of a gate due to a defect depends also on factors such as die temperature and power supply variations. As a result, it is no longer adequate to rely only on deterministic fault models. Stochastic techniques have also been proposed to compute reliably using logic gates that fail probabilistically [13].

II. MODELING OF GATE RELIABILITY

In this work, we consider a probabilistic fault model that allows any number of gates in the IC to fail probabilistically. Tests for this fault model, determined using the theory of output deviations, can be used to supplement tests for classical fault models, thereby increasing test quality and reducing the probability of test escape. By targeting multiple fault sites in a probabilistic manner, such a model is useful for addressing phenomenon or mechanisms that are not fully understood. Output deviations can also be used for test selection, whereby the most effective test patterns can be selected from large test sets during time-constrained and high-volume production testing [7].

*The work of K. Chakrabarty was supported in part by a research fellowship from the Alexander von Humboldt Foundation, Germany.
schematics.

The reliability of a single-output gate encompasses all the different input combinations of the gate, and for a given input combination, it provides the probability that the gate output is correct for the corresponding input combination. The probability that a gate output is correct can be different for the various input combinations. Formally, we define the reliability of a gate with \( m \) inputs and one output by a \( 2^m \)-dimensional vector.

**Definition 1:** The reliability \( R_i \) of a gate \( G_i \) with \( m \) inputs and a single output is a vector with \( 2^m \) components, defined as:

\[
R_i = (r_i^{(00...00)} r_i^{(00...01)} r_i^{(00...10)} \ldots r_i^{(11...11)}),
\]

where each component of \( R_i \) denotes the probability that the gate output is correct for the corresponding input combination.

For example, \( r_i^{(00)} \) is the probability that the output of a 2-input gate \( G_i \) is correct under input 00. If \( m = 2 \) for a logic gate, we have \( R_i = (r_i^{(00)} r_i^{(01)} r_i^{(10)} r_i^{(11)}) \).

The above gate-level reliability vectors can be generated in a number of ways, e.g., using layout information, inductive fault analysis [20], and failure data analysis. It can also be estimated using simple transistor-level failure probabilities. Our objective here is not to develop new techniques for determining reliability vectors, but rather to use these as inputs for the computation of output deviation. In practice, deviation-based test patterns can be generated using multiple sets of reliability vector estimates and the probabilistic fault model.

Consider the 2-input NAND and NOR gates shown in Fig. 1. Suppose each transistor can be stuck-open due to a defect, i.e., it cannot be switched on, with probability \( \alpha \). Similarly, suppose each transistor can be stuck-on due to a defect, i.e., it cannot be switched off, with probability \( \beta \). Next let us consider input combination \( x_1 x_2 = 00 \). If only stuck-on faults are considered, the NAND gate produces the correct output for this combination with probability \( 1 - \alpha^2 \) because the gate produces an incorrect output only if both transistors are stuck-open. (The absence of a second pull-up path affects the pull-up time for this input combination but this issue is ignored since worst-case pull-up times consider only one pull-up path.) Likewise, if we only consider stuck-on faults, the probability that the NAND gate produces the correct output for input 00 is \( 1 - 2\beta + \beta^2 \).

Table I presents the reliabilities of the NAND and NOR gates expressed in terms of transistor stuck-on and stuck-open probabilities.

![Fig. 1. 2-input NAND and NOR gates in static ratioless CMOS technology.](image)

<table>
<thead>
<tr>
<th>Combination</th>
<th>( R_{1 \text{NAND}}(x_1, x_2, \alpha) )</th>
<th>( R_{2 \text{NAND}}(x_1, x_2, \beta) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>( 1 - \alpha^2 )</td>
<td>( 1 - \beta^2 )</td>
</tr>
<tr>
<td>01</td>
<td>( 1 - \alpha )</td>
<td>( 1 - \beta )</td>
</tr>
<tr>
<td>10</td>
<td>( 1 - \alpha^2 )</td>
<td>( 1 - \beta^2 )</td>
</tr>
<tr>
<td>11</td>
<td>( 2\alpha - \alpha^2 )</td>
<td>( 2\beta - \beta^2 )</td>
</tr>
</tbody>
</table>

![Table I](image)

Let \( \bar{R}(\bar{x}_1, \bar{x}_2, \alpha, \beta) \) be the gate reliability for input combination \( (\bar{x}_1, \bar{x}_2) \), when any number of stuck-open and stuck-on faults can simultaneously occur. It can be easily shown that \( \bar{R}(\bar{x}_1, \bar{x}_2, \alpha, \beta) = R_1(\bar{x}_1, \bar{x}_2, \alpha) \cdot R_2(\bar{x}_1, \bar{x}_2, \beta) \). The result follows from the fact that for any input combination, the set of transistors that affects the output under stuck-on conditions is disjoint from the set of transistors that affects the output under stuck-open conditions.

In this paper, we limit ourselves to basic gates with one or two inputs. Extension of the theory of output deviations to gates with larger fanout is straightforward. Here, gates with larger fanout are first expanded to a network of basic gates.

### III. Computation of Signal Probabilities

Next we associate signal probabilities \( p_{i,0} \) and \( p_{i,1} \) with each line \( i \) in the circuit, where \( p_{i,0} \) and \( p_{i,1} \) are the probabilities for line \( i \) to be at logic 0 and 1, respectively. Obviously we have \( p_{i,0} + p_{i,1} = 1 \). The calculation of the signal probabilities is along the same lines as introduced in [21], and used later in [22]. To reduce the amount of computation as in [21], [22], signal correlations due to reconvergent fanout are not considered here.

Let \( i \) be the output of a two-input gate \( G \). Let \( j \) and \( k \) denote the input lines for this gate. If \( G \) is a NAND gate, we have:

\[
p_{i,0} = p_{j,0} p_{k,0} r_i^{(11)} + p_{j,0} p_{k,0} (1 - r_i^{(00)}) \]
\[
+ p_{j,0} p_{k,1} (1 - r_i^{(01)}) + p_{j,1} p_{k,0} (1 - r_i^{(10)})
\]
\[
p_{i,1} = p_{j,0} p_{k,0} r_i^{(00)} + p_{j,1} p_{k,1} r_i^{(01)}
\]
UTPUT DEVIATIONS AND TESTING

For any logic gate (or primary output) \( g \) in a circuit, let its fault-free output value for any given input pattern \( t_j \), be \( d \), \( d \in \{0, 1\} \). The output deviation \( \Delta_{g,j} \) of \( g \) for input pattern \( t_j \) is defined as \( p_{g,\overline{t}_j} \), where \( \overline{t} \) is the complement of \( t \). Intuitively, the deviation for an input pattern is a measure of the likelihood that the gate output is incorrect for that input pattern.

Next we formally define the probabilistic fault model for a combinational circuit \( C \), and its application in testing.

**Definition 2**: A combinational circuit \( C \) is defined as \( C = \{ G, P, T, Z, R \} \), where \( G = \{ G_1, G_2, \ldots, G_N \} \) is the set of logic gates in \( C \), \( P \) and \( T \) are the sets of primary inputs and outputs respectively, and \( R = \{ R_1, R_2, \ldots, R_N \} \) is the set of reliabilities of the gates in \( G \).

**Definition 3**: A probabilistic fault model \( F \) for circuit \( C \) is defined as follows. Each gate \( G_i \) can fail independently of other gates and its fault behavior is defined by \( R_i \).

Under this fault model, the output probability values of the circuit in response to an input pattern is no longer deterministic. Rather, it is given by the signal probabilities at primary outputs. Note that the circuit behavior is assumed to be deterministic after manufacturing; the probabilistic fault model is only used during test development to generate tests for defects that occur in a random fashion, and which cannot be easily modeled.

Consider the simple circuit shown in Fig. 2. According to \( F \), this circuit can fail in a number of ways, each of which is termed a fault event. Table II lists the various fault events \( E_1, E_2, \ldots, E_7 \) and the event \( E_0 \), corresponding to the fault-free case. It also lists the probability associated with each fault event, and the corresponding circuit output, for input pattern \( abcd = 0000 \). Only the events \( E_1, E_3, E_5, \) and \( E_6 \) are detected by the given input pattern. Let \( E \) be the event that the pattern 0000 detects a fault in the circuit. It can be easily seen that

\[
P(E) = P(E_1 \cup E_3 \cup E_5 \cup E_6)
\]

Figure 2 shows a simple circuits consisting of three gates \( G_1, G_2, \) and \( G_3 \) with reliability vectors (0.9 0.9 0.9 0.8), (0.8 0.9 0.9 0.9), and (0.8 0.9 0.9 0.9), respectively. For the three different deterministic input vectors 0000, 010, and 1111, the signal probabilities are determined and presented in Table II. The fault-free values at the output \( z \) are also listed in the first column of Table II.

In the next section, we use the concept of signal probabilities to develop the notion of output deviations. We then describe how to determine deviation-based test patterns that can be used to enrich test sets.

### Table II

**Signal Probabilities for Different Input Combinations**

\[
\begin{align*}
+ & p_{j,1}p_{k,0}r_{i(10)} + p_{j,1}p_{k,1}(1 - r_{i(11)}) \\
\end{align*}
\]

The above definition of the signal probabilities can be easily extended to the case of more than two inputs. It can also be easily verified that \( p_{i,0} + p_{i,1} = p_{j,0}p_{k,0} + p_{j,0}p_{k,1} + p_{j,1}p_{k,0} + p_{j,1}p_{k,1} = 1 \).

Let \( G \) be a gate with two inputs \( j \) and \( k \), controlling value \( c \), and inversion value \( v \). For example, a NAND gate has controlling value \( c = 0 \) and inversion value \( v = 1 \). Let \( T \) be the complement of the controlling value \( c \). The signal probabilities for the output \( i \) of such a gate can be easily expressed as follows:

\[
\begin{align*}
 p_{i,c \in \overline{T}} &= p_{j,c}p_{k,c}r_{i(ce)} + p_{j,c}p_{k,\overline{c}}r_{i(c\overline{e})} + p_{j,\overline{c}}p_{k,c}(1 - r_{i(\overline{c}e)}) + p_{j,\overline{c}}p_{k,\overline{c}}(1 - r_{i(\overline{c}\overline{e})}) \\
 p_{i,\overline{T}} &= p_{j,c}p_{k,c}(1 - r_{i(ce)}) + p_{j,c}p_{k,\overline{c}}(1 - r_{i(\overline{c}e)}) + p_{j,\overline{c}}p_{k,c}(1 - r_{i(\overline{c}e)}) + p_{j,\overline{c}}p_{k,\overline{c}}(1 - r_{i(\overline{c}\overline{e})}) \\
\end{align*}
\]

Next consider a gate without a controlling value, e.g., XOR and XNOR. For the XOR gate, the signal probabilities can be expressed as follows (the formulas for the XNOR gate are similar):

\[
\begin{align*}
 p_{i,0} &= p_{j,0}p_{k,0}r_{i(00)} + p_{j,0}p_{k,1}(1 - r_{i(01)}) + p_{j,1}p_{k,0}(1 - r_{i(10)}) + p_{j,1}p_{k,1}r_{i(11)} \\
 p_{i,1} &= p_{j,0}p_{k,0}(1 - r_{i(00)}) + p_{j,0}p_{k,1}r_{i(01)} + p_{j,1}p_{k,0}r_{i(10)} + p_{j,1}p_{k,1}(1 - r_{i(11)}) \\
\end{align*}
\]

The above example shows that the probability that an input pattern detects a fault can be calculated by enumerating the fault events. However, signal probability and deviation
calculations are sufficient to determine the fault detection probability. This observation is formalized by the following theorem, the proof of which is omitted due to lack of space.

**Theorem 1:** Let \( E \) be the event that input pattern \( t_j \) detects a fault in a circuit \( C \) at output \( z_i \) under fault model \( F \). Then \( P[\bar{E}] = \Delta z_{i,j} \).

The probability that \( t_j \) will produce an observable error at \( z_i \) for fault model \( F \) is directly proportional to \( \Delta z_{i,j} \). Therefore, a goal of testing is to apply those vectors to \( C \) that produce large deviations at the output. These tests can be used to supplement tests for standard fault models. From now on we only consider output deviations at outputs and use the terms “output deviation” and “deviation” interchangeably. In the next section, we experimentally study the fault detection capabilities of sets of patterns with different deviation values.

### V. Experimental Results

We use two arbitrarily chosen sets of reliability vectors for our experiments. These vectors are defined separately for each gate type. For example, for a 2-input NAND gate, we use 1) “low reliability” vector: \( R^{NAND2} = (0.8(00), 0.8(01), 0.8(10), 0.7(11)) \), and 2) “high reliability” vector: \( R^{NAND2} = (0.95(00), 0.95(01), 0.95(10), 0.85(11)) \). The steps in these experiments are as follows:

1. **Step 1:** Generate a set of target faults \( F_{\text{target}} \).
2. **Step 2:** Generate a set of random vectors \( T_{\text{rand}} \).
3. **Step 3:** Compute deviations for the vectors in \( T_{\text{rand}} \).
4. **Step 4:** Perform fault simulation on \( F_{\text{target}} \) using \( T_{\text{rand}} \). Add detected faults to the set \( F_{\text{det}} \).
5. **Step 5:** Use a predetermined threshold \( \gamma \) on the output deviation value to divide \( T_{\text{rand}} \) into two sets \( T_{\text{high}} \) and \( T_{\text{low}} \). Vectors that lead to a deviation greater than the threshold \( \gamma \) on at least one primary output are placed in \( T_{\text{high}} \). The other vectors are placed in \( T_{\text{low}} \).
6. **Step 6:** Determine the fault coverages for \( F_{\text{det}} \) using \( T_{\text{high}} \) and \( T_{\text{low}} \). These fault coverages are denoted by \( FC_{\text{high}} \) and \( FC_{\text{low}} \), respectively. To allow fair comparison involving fault coverages, the larger test set is randomly truncated such that the two test sets \( T_{\text{high}} \) and \( T_{\text{low}} \) have the same number of vectors.

In the following set of experiments (numbered as 1, 2, 3, 4), we use single stuck-line (SSL) faults, single non-feedback bridging-faults (s-NFBFs), and double non-feedback bridging-faults (double-NFBFs), as the types of faults in \( F_{\text{target}} \). Each double-NFBF contains 2 s-NFBFs. Moreover, we ensure that \( F_{\text{target}} \) only contains faults that are not detected by an SSL test set generated by Atalanta [24] with random-fill turned on. For simplicity, in this section \( F_{\text{target}}(i) \) refers to \( F_{\text{target}} \) used in Experiment \( i \). \( T_{\text{rand}} \) contains 9,704 vectors in Experiments 1, 2, 3, and 48,454 vectors in Experiment 3.

Experimental results show that \( FC_{\text{high}} \) is significantly higher than \( FC_{\text{low}} \) for most ISCAS’85 and ISCAS’89 benchmark circuits. Due to limited space, we only include representative results for selected benchmarks.

#### Experiment 1: SSL fault coverage

\( F_{\text{target}}(1) \) contains all SSL faults in the CUT. Fig. 3 shows results for s13207 and s38584 with the “high reliability” vector for the gates. The x-axis corresponds to a range of values for the threshold \( \gamma \). The y-axis indicates the fault coverage obtained using \( T_{\text{high}} \) and \( T_{\text{low}} \).

#### Experiment 2: s-NFBF fault coverage

To generate \( F_{\text{target}}(2) \), we first enumerate all s-NFBFs in the CUT for smaller benchmarks and a sampled set of s-NFBFs for the larger benchmarks. We consider wired-AND and wired-OR shorts in this experiment. Next faults detected by the SSL test set are removed from further consideration. In this way, we are able to demonstrate that high-deviation patterns can enrich test sets for fault detection. Fig. 4 shows the results for s13207 and s38584.

#### Experiment 3: double-NFBF fault coverage

To generate \( F_{\text{target}}(3) \), \( F_{\text{target}}(2) \) is used as a repository, from which 10,000 double-NFBFs are randomly generated. Again, to demonstrate test set enrichment, double-NFBFs detected by the SSL test set are removed. Fig. 5 shows the fault coverage results for c7552 and s38584.
calculate deviations for can therefore be viewed as a special case with (upstream as well as downstream) from obtained for s38584.

Experiment 4: Restricted Neighborhood Mode

A special case of the probabilistic fault model can be viewed as a restricted neighborhood mode. Here we assume a single defect site and impose the condition that gates in the neighborhood of this site are probabilistically faulty. A neighborhood is a set of gates in the vicinity of the gate that corresponds to the defect site. The size of a neighborhood is referred to as the radius \( r \) of the restricted neighborhood mode. Logic gates outside this neighborhood have reliability vectors will all their components equal to 1. This mode is useful in practice when prior information about potential defect locations in a chip is available. It is also useful during “targeted” test generation and test selection.

Information about neighborhoods can be extracted from layout data. In this paper, we only consider \( r = 0 \) and \( r = 1 \); in particular, we assume that the neighborhood consists of the defect site \( G \) and gates that are at most one-level away (upstream as well as downstream) from \( G \). (Experiments 1-3 can therefore be viewed as a special case with \( r = \infty \).) We calculate deviations for \( r = 0, 1, \infty \). Fig. 6 shows the results obtained for s38584.

We can see from Experiments 1-4, and from the results for other benchmark circuits that could not be included due to lack of space, that for most circuits, the fault coverage is higher for a set of vectors with high output deviation. Moreover, the choice of the threshold \( \gamma \) appears to be important. The best threshold value seems to be circuit-dependent.

The program to compute output deviations was implemented using C++. On a Pentium 4 PC with 2.6 Ghz processor and 1G memory running Linux, it takes approximately 80 seconds to compute the deviations for 10,000 vectors for full scan version of the largest benchmark circuit (s38584). Event-driven techniques can be used to decrease computation time; for consecutive vectors, only the gates whose outputs are changed need to be evaluated.

VI. CONCLUSIONS

We have described a probabilistic fault model that allows any number of gates in an integrated circuit to fail probabilistically. We have shown that tests for this fault model, determined using the theory of output deviations, can be used to supplement tests for classical fault models. This can potentially increase test quality and reduce test escapes. We have shown that output deviations can be used as a simple criterion to rank test patterns in terms of their effectiveness. Experimental results for benchmark circuits show that test sets with higher deviation lead to higher fault coverage for single
stuck-at and bridging faults.

The results presented in this paper open a number of interesting research directions. Experiments involving realistic defects such as resistive shorts and opens, as well as delay faults will shed more light on the effectiveness of the probabilistic fault model and the use of output deviations as a pattern grading metric. A deviation-based test generation tool will be useful to generate vectors with high output deviations. The propagation of deviations in the presence of unknown logic values (Xs) also needs to be investigated. Finally, the theory of deviations for sequential circuits is of interest, especially for designs that do not rely on full scan.

REFERENCES


Fig. 6. Fault coverage obtained using $T_{\text{high}}$ and $T_{\text{low}}$ for the s-NFBF faults, with the restricted neighborhood mode and “high reliability” vector.