Diagnosis-Assisted Adaptive Test

Xiaochun Yu, Member, IEEE, and Ronald DeShawn Blanton, Fellow, IEEE

Abstract—This paper describes a method for improving the test quality of digital circuits on a per-design basis by: 1) monitoring the defect behaviors that occur through volume diagnosis; and 2) changing the test patterns to match the identified behaviors. To characterize the behavior of a defect (i.e., the conditions when a defect is activated), physically-aware diagnosis is employed to extract the set of signal lines relevant to defect activation. Then, based on the set of signal lines derived, the defect is attributed to one of several behavior categories. Our defect level model uses the behavior-attribute results of the current failing population to guide test-set customization to minimize defect level for a given constraint on test costs, or alternatively, ensure that defect level does not exceed some predetermined threshold. Circuit-level simulation involving various types of defects shows that defect level can be reduced by 30% using this method. Simulation experiment on actual chips also demonstrates quality improvement.

Index Terms—Adaptive test, defect behavior, diagnosis, test quality.

I. INTRODUCTION

THE DIAGNOSIS of test data for improving yield is a topic of great interest [1]–[5]. For example, the work in [2] uses diagnosis results to track failure rates for various design features. These empirically observed failure rates are compared with expected failure rates to identify anomalies in the design and/or manufacturing process. The thinking is that the identified anomalies can be remedied in order to improve yields.

Diagnosis methods such as the ones described in [6] and [7] derive defect behaviors (i.e., defect activation conditions) in addition to the potential defect locations. The defect characteristics derived from a statistically significant population suggest the susceptibility of a design to a specific manufacturing process. Customizing a test set to those failure characteristics is likely to increase test quality, which is commonly measured using defect level (DL). DL is the ratio of defective chips to all the chips that fully pass all the tests, and it is typically expressed in units of defective parts per million shipped (DPPM). Since the failure characteristics may change over time, the custom test set should also change accordingly. Custom test is a type of adaptive test. Different from a static test method (e.g., a test set that is generated using a fault model) that does not depend on changing parameters, an adaptive test method can alter test conditions, test flow, test content, and test limits based on statistical analysis of various forms of manufacturing data [8]. Adaptive test has been successfully employed in various ways. For example, test cost can be reduced by reordering test patterns based on fail count [9]–[12]. Adaptive limit setting for quiescent supply current (IDDQ) and minVDD outlier screening has also been shown to improve test quality [9], [11]–[15].

This paper describes diagnosis-assisted adaptive test (DAT), an approach that improves test quality or test cost for static defects by identifying and applying tests that optimally detect the defect characteristics of integrated circuits (ICs) being currently manufactured. In its current incarnation, DAT targets static defects which are different from a sequence-dependent defect (e.g., a transistor stack-open defect) whose activation can require multiple clock cycles. The activation and error propagation of a static defect, on the other hand, occurs within a single clock cycle. Moreover, for a static defect at a site $f$, defect activation is assumed to be controlled by the neighborhood of $f$, which is the set of signals that are in close physical or logical proximity to $f$ [6].

DAT uses diagnosis-extracted models of chip failures along with a model for estimating DPPM. Both are incorporated in a quality-monitoring methodology that ensures a desired level of quality by changing some proportion of the production test patterns to match the current defect characteristics derived from diagnosis. DAT is dynamic in nature and thus differs from the typical approach that assumes sufficient quality levels are maintained using the tests developed during the time of design or “first silicon.” DAT presupposes that failure characteristics can change over time but with a time constant that is sufficiently slow, thereby allowing the test set to be altered so as to maximize coverage of the failure characteristics actually occurring. If the failure characteristics are unchanging, it will then be possible to tune the test set once to match the fixed failure characteristics. We therefore believe it is possible to minimize DPPM for a given constraint on test costs, or alternatively, ensure that DPPM does not exceed some predetermined threshold. Fig. 1 illustrates a possible scenario where DPPM changes over time. At $t_1$, DPPM prediction begins based on a sufficient amount of diagnosis data obtained during the period between $t_0$ and $t_1$. When the predicted DPPM approaches a specific DPPM limit at $t_2$, the tests are altered to lower DPPM at $t_3$. When DPPM is sufficiently low (e.g., $t_3$), test quality can be traded off to reduce test application time, which leads to the slight increase in DPPM shown at $t_4$. Notice, however, at $t_5$, an excursion is assumed to have
II. Defect Behavior Attribution

MD’X and FBI derive, for each failing chip, the defect sites and each site’s relevant neighbors. Based on the relevant neighbors identified, each defect can be placed into one of several mutually exclusive behavior categories. Since the behaviors of some defect types overlap, there is no one-one mapping from defect behavior to defect type in many cases. Thus, the objective of defect behavior attribution is not to derive the defect type directly from the relevant neighbors identified. But instead, behavior-attribution results in a behavior distribution, that is, the relative occurrence rates of various behavior categories. The behavior distribution of a failing population of chips captures the failure characteristics of that population. The details of behavior attribution are explained in the following subsections. Specifically, in Section II-A, the behavior-attribute rules are described and the relationship between each behavior category and defect type is analyzed. In Section II-B, various experiments are conducted to validate the relationships described in Section II-A.

A. Categorizing Defect Behavior

Table I lists the behavior-attribute-rules that are based on different sets of relevant neighbors. (The notation used in Table I is described in Table II.) Specifically, Table I lists eight different behavior categories and their association to the four defect types considered, namely, open, bridge, cell defect, and “unknown.” Though eight behavior categories and four defect types are considered in this paper, behavior attribution can be extended to other, and more refined defect types and behaviors.

**Category 1:** FBI identifies the relevant neighbors using the characteristics of the common defect types under consideration. If the relevant neighborhood cannot be identified, then the behavior of the defect is deemed “unknown” to signify the observed behavior does not completely align with the common defect types under consideration.

![Figure 1: Illustration of how DPPM can change over time for a given design in a given manufacturing process.](image1)

![Figure 2: Overview of DAT.](image2)

**TABLE I**

<table>
<thead>
<tr>
<th>ID</th>
<th>Relevant Neighborhood</th>
<th>Defect Behavior</th>
<th>Defect Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Fail in driving RNf</td>
<td>Unknown</td>
<td>Unknown</td>
</tr>
<tr>
<td>2</td>
<td>RNf ≠ Ø and</td>
<td>Stack-at</td>
<td>All types possible</td>
</tr>
<tr>
<td>3</td>
<td>∀(RNf ∩ (f) )/Pf, y ≠ Ø</td>
<td>Cell multibehavior</td>
<td>possible</td>
</tr>
<tr>
<td>4</td>
<td>∀(RNf ∩ (f) )/Pf, y ≠ Ø and</td>
<td>f’s value depends</td>
<td>Bridge</td>
</tr>
<tr>
<td>5</td>
<td>RNf/DPNf/DPNf/DPNf/DPNf</td>
<td>f’s value depends</td>
<td>Bridge</td>
</tr>
<tr>
<td>6</td>
<td>RNf/DPNf/DPNf/DPNf/DPNf</td>
<td>f’s value depends</td>
<td>Bridge</td>
</tr>
<tr>
<td>7</td>
<td>RNf/DPNf/DPNf/DPNf/DPNf</td>
<td>f’s value depends</td>
<td>Bridge</td>
</tr>
<tr>
<td>8</td>
<td>RNf/DPNf/DPNf/DPNf/DPNf</td>
<td>f’s value depends</td>
<td>Open</td>
</tr>
</tbody>
</table>
Category 2: If a defective site \( f \) does not have any relevant neighbors, then \( f \) manifests faulty behavior regardless of its neighborhood. In other words, \( f \) behaves as a signal line being permanently stuck-at a fixed logic value (i.e., a stuck-at fault), under the applied test set. Any type of defect can behave as a stuck-at fault, especially when the applied tests do not expose all the possible behaviors of a defect.

Category 3: If: 1) the union of a defective site \( f \) and its relevant neighborhood is a subset of the union of the inputs and the output of a cell; and 2) there is at least one relevant neighbor, then the behavior of the defect mimics a misbehaving cell (i.e., the truth table of the cell’s function has changed). This is likely due to a cell defect, but there are other possibilities. For example, a cell input affected by an open can manifest as a change in cell functionality. Also, for a given test set, a bridge whose activation depends on the driving strength of a cell can also mimic cell misbehavior.

Category 4: If: 1) the relevant neighborhood of a defective site \( f \) includes another defective site \( g \); 2) \( f \) and \( g \) have different effects on the failing circuit; and 3) the behavior does not agree with Categories 1, 2, or 3, then the value of \( f \) is dependent on \( g \). A bridge defect affecting two signal lines can produce this behavior. Although an open on a stem line can also cause several (or all) of its downstream branches to be defective, error manifestation on the defective branches does not depend on each other. Neither can a cell defect cause the aforementioned behavior. Thus, a defect that causes this observed behavior can be classified as a bridge.

Category 5: If: 1) the relevant neighborhood of a defective site \( f \) includes inputs to the cell driving \( f \) or inputs to the cells driving the physical neighbors of \( f \); 2) at least one of the inputs included in the relevant neighborhood is not a physical neighbor of \( f \) or a side input to any of the cells driven by \( f \); and 3) the behavior does not agree with any of the Categories 1–4, then the logical value resulting in a defective site \( f \) is a function of the drive strength of its driving cell and/or its physical neighbors. A bridge defect can cause such behavior, but an open or a cell defect cannot. Thus, a defect that causes this behavior can be classified as a bridge.

Category 6 and 7: If the relevant neighborhood of a defective site \( f \) includes only the physical neighbors of \( f \), and the relevant neighborhood does not conform to any of the Categories 1–5, then the value of \( f \) depends only on its physical neighbors. For this situation, \( f \) can be affected by an open or bridge, but not a cell defect. If \( f \) is affected by an open and the activation of the defect is only dependent on its physical neighbors, then the coupling between \( f \) and its relevant neighbors should be large enough to determine the value of \( f \). Thus, when \( f \) is affected by an open, it is more likely that the combined effect of multiple physical neighbors will determine the value of \( f \) rather than a single neighbor \[18\]. Also for an open, it is less likely that the value of \( f \) is always controlled by a specific physical neighbor. On the other hand, if \( f \) is affected by a bridge, the physical neighbors that control \( f \) are likely the lines bridged with \( f \). A bridge involving two lines is much more likely than a bridge involving three or more lines. Therefore, an open defect in this behavior category is more likely to have more than one relevant neighbor. On the other hand, a bridge in this behavior category is more likely to have a single relevant neighbor. The behavior is further partitioned by examining: 1) the value of the defective site as a function of a single physical neighbor (Category 6); and 2) the value of the defective site as a function of two or more physical neighbors (Category 7). This is reasonable since the likelihood of a defect of Category 6 being an open over a bridge is likely different from that of Category 7 even though the two categories correspond to the same defect types.

Category 8: If: 1) the value of a defective site \( f \) is dependent on the side inputs to the cells driven by \( f \); 2) at least one side input included in the relevant neighborhood is a) not an input to any of the cells driving the physical neighbors of \( f \), b) not an input to the cell driving \( f \), and c) not a physical neighbor of \( f \); and 3) the behavior does not agree with any categories from 1 to 7, then the value of \( f \) depends on the side inputs of the cells driven by \( f \). An open affecting \( f \) can cause this type of behavior, but a bridge or a cell defect cannot. Thus, a defect exhibiting this behavior is classified as an open.

### Validation Experiment

To validate the effectiveness of the behavior-attribution rules of Table 1, a simulation experiment is performed using several benchmark circuits [19], [20]. To better mimic reality, a pool of defective circuits is generated by randomly injecting cell defects, gross interconnect opens, and two-line bridge defects, [21]. The cell defects injected include internal line opens, bridges between internal lines, bridges between inputs of a cell, feedback bridges between the inputs and output of a cell, transistor stuck-opens, transistor stuck-close defects, and "nasty polysilicon" defects [22]. For each layout with an injected defect, the corresponding circuit-level [23] netlist is extracted and circuit-level simulation is performed using a test set that achieves 100% stuck-at fault efficiency (i.e., a 1-detect test set), and the circuit responses are digitized and collected. A clock cycle that is much slower than the rated clock is used so as to mimic scan test. Table III shows the number of defective circuits created for each defect type and benchmark.

1A nasty polysilicon defect (NPSD) refers to a spot deformation of extra polysilicon that causes an unwanted connection between the gate and the source (drain) of a transistor that also prevents the metal terminal connection of the source (drain) of a transistor. Since an NPSD causes a short and an open at the same time, it is inherently “nasty.”
According to our analysis in Section II-A, all cell defects should be placed in either the stuck-at or the cell-misbehavior category. The behavior of a bridge can be placed in any of the categories except 1 or 8. For an open, its behavior can be in any category except 1, 4, and 5. To evaluate our analysis of the relationship between different categories of defect behaviors and defect types, each defective circuit is diagnosed using MD²X and the relevant neighbors of each defective site are identified using FBI. Then, each defect is categorized using Table I. The percentages of defects exhibiting stuck-at behavior are 59%, 14%, and 97% for cell, bridge, and open defects, respectively. The behavior distributions of defects that do not behave as stuck-at faults are shown in Fig. 3. It shows that among defects that do not behave as stuck-at faults, 84% of the cell defects, 99% of the bridges, and 99% of the opens have behaviors that agree with our analysis. In other words, the behaviors of a vast majority of defects of each type are attributed to the behavior categories corresponding to that defect type (Table I). There is however a small percentage of disagreement, which is due to misidentification of the relevant neighborhood [7]. The behavior attribution error for cell defects is much higher than bridge and open defects, since the complexity exhibited by cell misbehavior can be mimicked by some other defect behaviors.

Defect behaviors are likely to vary across different processes and different designs, which means Fig. 4 varies for different technologies. Thus, for a design manufactured using a different process, behavior attribution needs to be applied to derive the behavior distribution for that design-process combination.

### III. DPPM

In this section, a model for estimating DPPM using the behavior-attribution results is described. The derivation of the model for estimating DPPM is given in Section III-A, and experiment results that validate the model are described in Section III-B.

#### A. DL Model

Many models for predicting DL have been proposed. For example, a model that predicts DL using multimodel fault coverage is presented in [24]. Their DL model is based on the assumption that defects of different types are equally likely to occur, which of course may not be true in reality. In [25], a...
DL model that is based on the number of times each site is observed is described. The particular activation conditions of a given defect type are, however, not taken into account. It is quite likely that the values on the relevant neighbors of a defect that affects a site \(i\) are the same for many patterns where \(i\) is observed. In such cases, using a test set with a repeated number of observations does not necessarily lead to lower defect level especially for the static defects considered here. In addition, the model of [25] assumes that a defective chip passes the testing process and therefore is shipped if the erroneous effect of at least one defective site is not observed. However, in the context of multiple defects, a defective chip passes test only if no error from any of the defective sites is observed. So, the assumption in [25] can lead to an overestimation of defect level.

To address the shortcomings just outlined, a DL model using an estimate of the defect-type distribution has been proposed in [26]. As shown in [7], the defect-type distribution for a design fabricated in a given manufacturing process can be estimated using the behavior distribution of each defect type (e.g., Fig. 3), which is derived from the behavior-attrition results of a sufficient number of PFAed2 defects that affect the design. The method of [7] can effectively estimate the defect-type distribution if the behavior distribution of each defect type: 1) remains static or changes slowly; and 2) varies from other defect types. PFA is an expensive and time-consuming process however. Moreover, there may be no or not enough PFA results to generate an accurate behavior distribution per defect type at the beginning of a product’s life time. Thus, it is desirable if DL can be estimated without PFA results. In the work of [26]-[29], a defective chip examined using diagnosis is classified into one of several defect types directly from the defect behavior extracted without the use of PFA. However, due to the ambiguity in mapping defect behavior to defect type, which is well illustrated by Table I and Fig. 3, such classification methods result in defect-type distributions that are very different from the actual as demonstrated in [7]. The DL model presented here does not depend on an accurate estimation of defect-type distribution. Instead, it uses the occurrence rate of each behavior category, which is derived through diagnosis and the behavior attribution method described in Section II-A. In the following, we define the DL model in terms of several probabilities.

For each site \(i\), the following probabilities are defined.

\[
P_{(\text{defective})} = \text{Probability that site } i \text{ is affected by a defect}.
\]

\[
P_{(\text{detected})} = \text{Probability of detecting the defect that affects site } i.
\]

\[
P_{(\text{behavior } j)} = \text{Probability of a defect with behavior } j \text{ affecting } i, \text{ where behavior } j \text{ is one of the categories in Table I.}
\]

\[
P_{(j \text{ detected})} = \text{Probability of detecting a defect of behavior } j \text{ that affects site } i.
\]

\[
P_{(j \text{ activated state } k)} = \text{Probability of activating a defect of behavior } j \text{ that affects a site } i \text{ when the corresponding relevant neighborhood has state } k.
\]

To make the problem more tractable, independence is assumed to exist among the defect activation and corresponding error-propagation conditions for defects affecting different locations. In other words, it is assumed that the detection of a defect at one site does not influence the detection of a defect at another site when there are multiple defects in the circuit. A chip \(C\) is shipped if: 1) \(C\) is defect-free; or 2) none of the defects affecting \(C\) is detected by the testing process. In other words, a chip \(C\) is shipped if for every site \(i\), there is no defect that affects \(i\), or if there is one, it is not detected.

Thus, the probability of \(C\) being shipped, \(P_{\text{ship}}\), is computed as follows:

\[
P_{\text{ship}} = \prod_{i \in \text{all sites}} (1 - P_{(\text{defective})} \times P_{(\text{detected})})
\]

\[
= \prod_{i \in \text{all sites}} \left(1 - \sum_{j \in \text{BEHVC}} P_{(\text{behavior } j)} \times P_{(j \text{ detected})}\right)
\]

where \(\text{BEHVC}\) is the set of all behavior categories. Derivations for both \(P_{(\text{defective})}\) and \(P_{(\text{behavior } j)}\) are described next.

A defect of behavior \(j\) affecting a site \(i\) is detected if there is a test pattern \(t\) that activates the defect and sensitizes \(i\) (despite the possible existence of other defects) to some observable point. Therefore:

\[
P_{(j \text{ detected})} = 1 - P_{(j \text{ is not activated for any test sensitizing } i)}
\]

\[
= 1 - \prod_{k \in \text{RNS}_i} (1 - P_{(j \text{ activated state } k)})
\]

where \(\text{RNS}_i\) consists of all the states for the relevant neighborhood for a defect of behavior \(j\) at site \(i\) for the test patterns where site \(i\) is sensitized. \(\text{RNS}_i\) can easily be obtained by analyzing the applied test set.

Assume the probability of a defect of behavior \(j\) affecting a site \(i\) is the same for all sites. Thus, the fraction of a failure population affected by a defect of behavior \(j\), which can be derived using the behavior attribution method (Section II-A), equals \(\frac{\text{Yield}}{\sum_{j \in \text{BEHVC}} P_{(\text{behavior } j)}}\). Moreover, the theoretical yield, which is the percentage of the manufactured chips that are defect-free, can be expressed as

\[
\text{Yield} = \prod_{i \in \text{all sites}} (1 - \sum_{j \in \text{BEHVC}} P_{(\text{behavior } j)}).
\]

Thus

\[
\sum_{j \in \text{BEHVC}} P_{(\text{behavior } j)} = 1 - \sqrt{\text{Yield}}
\]

Therefore, \(P_{(\text{behavior } j)}\) can be computed for all behavior categories at every site \(i\) for a certain value of yield.

Using the equations for \(P_{(\text{detected})}\) and \(P_{(\text{behavior } j)}\), \(P_{\text{ship}}\) can be derived for a given test set using (1). Since defect level is defined to be the ratio of the defective chips in the chips that pass all the tests, defect level can therefore be expressed as

\[
DL = \frac{P_{\text{ship}} - \text{Yield}}{P_{\text{ship}}}
\]
TABLE IV
NUMBER OF FAILING LSI TEST CHIPS AND THEIR ASSOCIATED TEST SETS

<table>
<thead>
<tr>
<th>Tester Data Set</th>
<th>No. of Tests</th>
<th>No. of Failing Chips</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>234</td>
<td>773</td>
</tr>
<tr>
<td>2</td>
<td>239</td>
<td>911</td>
</tr>
<tr>
<td>3</td>
<td>218</td>
<td>773</td>
</tr>
</tbody>
</table>

B. Validation

An evaluation experiment is conducted using the LSI test chips whose behavior-attribution results are shown in Fig. 4. The test chip design consists of 384 ALUs. Although each ALU is identical in logic and layout, the ALUs were partitioned into several groups, each with its own test set. The test-set size and number of failing ALUs detected for each group are given in Table IV. The first 197 test patterns are the same across all three test sets. The 197 test patterns are used in this experiment.

Given a yield value, \( P_{\text{FAIL}} \) and DL can be computed according to (1) and (5) using the behavior-attribution results shown in Fig. 4 (Section II-B). For simplicity, it is assumed for a defect of behavior \( j \) affecting a site \( i \) that for any state \( k \in RNS_j \), \( P(k) \) (activated; inactive) \( k = 1/2 \). In other words, each state of the relevant neighborhood for a defect of behavior \( j \) at a site \( i \) is assumed to be equally likely to activate the defect at 50%. However, this probability can be made more precise by using a defect density distribution and the layout information. The 197 tests that are common to each ALU are simulated to derive \( RNS_j \). \( P_i(j, \text{activated}) \), which is the set of relevant neighborhood states for each behavior category, site pair (\( i, j \)). Next, \( P_j \) (detected) is computed for every site \( i \) and every behavior category \( j \) using (2). The probability of a defect of a specific behavior category affecting a site is derived according to (4) using the yield value and the behavior-attribution results from Fig. 4 (Section II-B). Then, \( P_{\text{FAIL}} \) is computed using \( P_i(j, \text{detected}) \) and \( P_i(j, \text{behavior}) \), and DL is derived using (5).

Since the actual escape is unknown, the DL model cannot be directly validated. However, the number of chips failed for each test pattern is known from the fail data. On the other hand, the expected number of chips failed by a set of test patterns, \( E_{\text{FAIL}} \), equals \( (1 - P_{\text{FAIL}}) \times \text{Total} \), where Total is the total number of chips tested. Thus, it is possible to validate the model for \( P_{\text{FAIL}} \). Since the model for \( P_{\text{FAIL}} \) is a function of Yield, \( E_{\text{FAIL}} \) is a function of Yield and \( \text{Total} \), both of which are not given by the fail data. Thus, to validate the model for \( P_{\text{FAIL}} \), a four-fold cross validation [30] is performed. In other words, the set of the chips failed by these 197 test patterns, \( G_i \), is randomly partitioned into four groups of equal size. For each group of failing chips, \( G_i \), the maximum likelihood estimate [30] for Yield and Total, \( \text{Yield}, \text{Total} \), are derived from the chips in \( G_i \). Then, \( E_{\text{FAIL}}(\text{Yield}, \text{Total}) \) is compared with the observed number of chips failed in \( G_i \). Fig. 5 shows for each group in the four-fold cross validation, the estimated and the observed cumulative number of failing chips versus test pattern index, as well as the 95% confidence intervals of the estimations. Specifically, Fig. 5(a) compares the estimated and the observed cumulative number of failing chips in group 1. Similarly, Fig. 5(b)–(d) makes similar comparisons for groups 2, 3, and 4, respectively. As shown in Fig. 5, the estimated cumulative fallout count tracks the observed very well for each group, and the estimated values are well contained in the corresponding 95% confidence intervals. The average differences between the estimated and the observed values for the four groups are 1.73, 2.98, 1.54, and 1.89, respectively. This result means that the estimation of the \( P_{\text{FAIL}} \) model tracks the observed number of failing chips very well.

IV. CUSTOM TEST FOR QUALITY

This section explores customizing the actual test patterns (but not increasing the number of tests) of the applied test set to match the failure characteristics for reducing DPPM. The use of a test-selection method [31], [32] for reducing DPPM is described in Section IV-A, the improvement in DL is predicted in Section IV-B, and the resulting test-quality improvement is validated in Section IV-C.

A. Test Selection for Reducing DPPM

According to the defect level model described in Section III-A, DL can be reduced if for every site \( i \), the defect possibly affecting \( i \) is activated using additional relevant neighborhood states for test patterns that sensitize \( i \). In other words, DL can be reduced by applying a new test that sensitizes the site \( i \) and activates some defect affecting \( i \) using an uncovered relevant neighborhood state (i.e., a relevant neighborhood state which has not been applied by any test where \( i \) has been sensitized). This observation agrees with the result of an experiment for a physically aware N-detect (PAN-detect) test set [33], in which each stuck-at fault is detected using at least \( N \) unique neighborhood states. In that experiment, the PAN-detect test set, as well as stuck-at, IDDQ, logic built-in self-test, and transition-fault tests, are applied to 99718 IBM ASICs manufactured in 0.13 \( \mu \)m technology. The testing-experiment result shows that the PAN-detect test set uniquely fails three chips that passed the other test sets. This result implies that detecting a fault using more unique neighborhood states increases the defect detection probability.

However, since the probability of a defect of a particular behavior category affecting a given site is different, which is demonstrated in Section II-B, a test that activates a certain number of uncovered relevant neighborhood states for one behavior category can be more preferable in reducing DPPM than a test that activates the same number of uncovered relevant neighborhood states for another behavior category. The implications of this observation are explored next. According to (5), defect level decreases as \( P_{\text{FAIL}} \) decreases for a given yield. Recall that \( P_{\text{FAIL}} \) can be computed using (1). Taking the log of both sides of the equation yields the expression

\[
\log(P_{\text{FAIL}}) = \sum_{\text{relevant}} \log(1 - \sum_{\text{cover}} P_i(\text{behavior}) \times P_j(\text{detected})).
\]
Therefore, the reduction in $\log(P_{SHIP})$ that results from activating a defect of behavior $j$ at a sensitized site $i$ for test $t_k$ using an uncovered relevant neighborhood state $s_{ijk}$, $\delta_{ijk}$, can be computed using the following equation:

$$\delta_{ijk} = \text{abs} \left( \log \frac{1 - \sum_{j \in \text{BEHVC}} P_i(\text{behavior } j) \times P_i(\text{j detected}')} {1 - \sum_{j \in \text{BEHVC}} P_i(\text{behavior } j) \times P_i(\text{j detected})} \right)$$  \hspace{1cm} (7)

where

$$P_i(\text{j detected}) = 1 - \prod_{k \in \text{RNS}_j} (1 - P_i(\text{j activated}|\text{state } k))$$  \hspace{1cm} (8)

and

$$P_i(\text{j detected}') = 1 - \prod_{k \in \text{RNS}_j} (1 - P_i(\text{j activated}|\text{state } k)).$$  \hspace{1cm} (9)

Again, $\text{RNS}_j$ is the set of all relevant neighborhood states for a behavior category $j$ in the already-selected test patterns when $i$ is sensitized.

The total decrease in $\log(P_{SHIP})$ due to applying a new test $t_k$ is the summation of the decrease in $\log(P_{SHIP})$ due to the activation of every uncovered relevant neighborhood state (if any) of each defect behavior at every sensitized site $i$ in $t_k$. The test that leads to the largest decrease in $\log(P_{SHIP})$ is the most preferable. Thus, tests are selected, one at a time, from a large test set in a way that greedily decreases $\log(P_{SHIP})$. The pseudocode for test selection is given in Procedure 1.

### B. Improved Test Quality

In this experiment, Procedure 1 is used to select a test set for the ALU that is of the same size as the original test set corresponding to the first test set shown in Table IV. The union of a 20-detect test set generated by the Cadence Tool Encounter Test [34] and the original test set is used as the test pool $T_{pool}$ for selection. The decrease in $\log(P_{SHIP})$ due

---

**Procedure 1** Test selection for reducing defect level

1. Initialize the selected test set, $T_{sel}$, to be empty.
2. Generate an $N$-detect test set, $T_{N-det}$.
3. $T_{pool} = T_{N-det} \cup$ original test set.
4. while the size of $T_{sel}$ is less than the original test-set size do
5. for each test $t_k \in T_{pool}$ do
6. Weight($t_k$) = 0
7. if $t_k \not\in T_{sel}$ then
8. for each defect behavior $j$ and each sensitized site $i$ in $t_k$ do
9. if an uncovered relevant neighborhood state $s_{ijk}$ is established then
10. Weight($t_k$) = Weight($t_k$) + decrease of $\log(P_{SHIP})$ due to $s_{ijk}$
11. end if
12. end for
13. end if
14. end for
15. Add the test $t_k$ with the largest Weight($t_k$) to $T_{sel}$
16. end while
to activating a defect of behavior category $j$ at a sensitized site $i$ using an uncovered relevant neighborhood state $s_{ik}$ is derived using (7). To compare the quality of the new selected test set with traditional $N$-detect test sets, several $N$-detect test sets are also generated using Encounter Test [34]. Fig. 6 shows the DPPM comparison between the selected test set, the original test set, and the traditional $N$-detect test sets assuming a yield value of 85%. Due to the randomness in the generation of the $N$-detect test sets [i.e., each $N$-detect test set is generated independently which means that the tests in the $(n-1)$-detect set are not necessarily a subset of the subsequent $n$-detect test set], the DPPM for the conventional $N$-detect test set does not necessarily decrease monotonically with an increase in $N$. The selected test set achieves a lower DPPM with a smaller test-set size than traditional $N$-detect test sets that are not neighborhood aware [32], nor focused on the characteristics of the failure population. Fig. 6 also demonstrates that the selected test set improves test quality (i.e., reduces DPPM) without increasing the number of tests. Although the data shown in Fig. 6 is based on a yield value of 85%, the DPPM comparison results hold for other values as well.

C. Simulation Validation

The experiment in Section IV-B has shown that the test set selected using Procedure 1 is expected to achieve a lower DPPM value. To show that the number of test escapes can be reduced by applying Procedure 1, a simulation experiment using the full-scan version of the ISCAS89 benchmark circuit s713 [20] is performed. A pool of defective circuits is generated and the corresponding circuit responses are determined using the method described in Section II-B for a 100% stuck-at fault detection test set (i.e., a 1-detect test set). The resulting defect population consists of 419 front-end cell defects, 1829 gross interconnect opens, 806 missing vias, and 512 two-line bridges. During simulation, these defects are detected and those that escape detection are determined. To mimic application in a production environment, the following steps are performed. For each defect type, 25% of the defective circuits affected by that type are randomly selected. The selected defective circuits, referred to as $D_{\text{future}}$, constitute the current failing population. The remaining 75%, which are referred to as $D_{\text{current}}$, are assumed to be the defective chips that are fabricated in the future. The test set applied to $D_{\text{current}}$ is a 1-detect test set. The test pool ($P_{\text{pool}}$) used for test selection is the union of a 50-detect test set generated by Encounter Test [34] and the original 1-detect test set. MD²X, FBI, and behavior attribution are performed on circuits in $D_{\text{current}}$ that fail the 1-detect test set to derive the occurrence rate of each behavior category. Finally, for the defective circuits in $D_{\text{current}}$, circuit-level simulation is used again to identify the test escapes, that is, the defective circuits in $D_{\text{current}}$ that are not detected by $T_{\text{dat}}$. The test-set sizes and the number of defective circuits in $D_{\text{current}}$ that escape each test set are given in Table V.

The simulation results reveal that the resulting selected test set $T_{\text{dat}}$ reduces the number of test escapes in $D_{\text{current}}$, by 30% without increasing the number of tests. Moreover, the DPPM of the selected test set is only 0.09% higher than $T_{\text{dat}}$, even though the size of $T_{\text{dat}}$ is more than three times the size of $T_{\text{dat}}$.

V. APPLICATION SCENARIOS

In this section, challenges related to the application of a DAT-based test set in an actual production environment are discussed. The impact of process variations on the application of DAT is discussed in Section V-A. In Section V-B, the number of tested ICs required to determine that a DAT test set improves test quality with high confidence is derived. Finally, various scenarios of DAT are described in Section V-C.

A. Impact of Process Variations

The discussion thus far in this paper has been on defects that are composed of extra or missing material within the manufactured ICs. Common defects include missing vias, interconnect opens, unintended connections between signal lines, and others. However, due to uncertainty in the manufacturing process, all manufactured ICs are also affected by process variations, which are deviations from the intended structural or electrical parameters of concern [35]. For example, transistor gate length and width can vary significantly for small feature sizes due to the variations in the exposure system [36]. Due
exhibiting behavior $j$ and exhibits behavior $j$ given that the IC has process parameter values $p$, and $P(p)$ is the probability that a failing IC has process parameter values $p$. Therefore, a good estimation for $P(j | p)$ requires the behavior-attribution results from a sufficient amount of failing ICs that can approximate both the actual distribution of relevant process parameters and the behavior distribution for a given set of process parameter values.

In the diagnosis-assisted test, the assumption is that the behavior distribution of failing ICs is monitored over time. A new DAT test set is generated if the following two criteria are satisfied. First, the current behavior distribution derived with high confidence is different from the one used to generate the currently employed DAT test set. Second, for any behavior category considered, the nonoverlapping interval between the confidence intervals for the current and the original behavior distribution is more than the error resulting from MD$^2$X and FBL, which can be deduced using simulation experiments. In general, it is expected that a small number of failing ICs will lead to large confidence intervals which means, of course, that a larger change in behavior distribution has to be observed to trigger the generation of a new DAT test set.

### B. Sample Size for Validating Quality Improvement

To confirm that a DAT test set improves product quality for actual manufactured ICs, the current and the DAT test sets should be applied to a large population of manufactured ICs. In the following, the number of ICs required to draw a high-confidence conclusion concerning DAT effectiveness based on defect level (DL) is derived.

Suppose test evaluation is conducted on a population of $m$ ICs, some of which are defective while others are defect-free. Let $DL_{Crnt}$ be the defect level resulting from the application of the current test set on the $m$ ICs. In other words

$$DL_{Crnt} = \frac{P_{adjCrnt} - Yield}{\hat{Yield} - DL_{Crnt}}$$

where $P_{adjCrnt}$ is the percentage of ICs in the population of $m$ ICs that pass all the tests in the current test set and $Yield$ is the percentage of good ICs in the population of $m$ ICs tested. Thus

$$\hat{P}_{adjCrnt} = \frac{Yield}{1 - DL_{Crnt}}$$

Similarly, $P_{adjDAT}$, the percentage of ICs in the set of $m$ ICs under test that pass the DAT test set, can be calculated using the following equation:

$$P_{adjDAT} = \frac{Yield}{1 - DL_{DAT}}$$

where $DL_{DAT}$ is the defect level resulting from the application of the DAT test set to the same $m$ ICs. Let $q$ be the relative reduction in defect level resulting from the DAT test set over the current test set for the $m$ ICs. In other words

$$DL_{DAT} = (1 - q) \times DL_{Crnt}.$$
The random variable $X_i$ is defined, where $X_i = 1$ if an IC $i$ passes the current test set but is failed by the DAT test set, $X_i = -1$ if an IC $i$ is failed by the current test set but passes the DAT test set, and $X_i = 0$ otherwise. Thus

$$\frac{1}{m} \sum_{i=1}^{m} X_i = \hat{P}_{\text{shipCrnt}} - \hat{P}_{\text{shipDAT}}.$$  \tag{16}$$

According to Hoeffding’s inequality [37], for $\epsilon > 0$

$$P\left(\frac{1}{m} \sum_{i=1}^{m} X_i - (\hat{P}_{\text{shipCrnt}} - \hat{P}_{\text{shipDAT}}) \geq \epsilon\right) \leq e^{-2\epsilon^2 m}.$$  \tag{17}$$

where $P_{\text{shipDAT}}$ ($P_{\text{shipDAT}}$) is the probability of an IC passing all the tests in the current (DAT) test set. Setting $\epsilon$ to $\frac{1}{2} \sum_{i=1}^{m} X_i$ leads to the following inequality:

$$P(\hat{P}_{\text{shipCrnt}} - \hat{P}_{\text{shipDAT}} \leq 0) \leq e^{-\frac{1}{2} \sum_{i=1}^{m} X_i^2}.$$  \tag{18}$$

By analyzing (15), (16), and (18), the minimum $m$ required to conclude $P_{\text{shipDAT}} > P_{\text{shipCrnt}}$ (i.e., the DAT test set results in a reduction in defect level) with a confidence level of $(1 - \alpha)$ is derived to be

$$m = 2 \times \left(\frac{(1-(1-q) \times DL_{\text{Crnt}}) \times (1-\hat{DL}_{\text{Crnt}})}{\text{Yield} \times q \times DL_{\text{Crnt}}} \right)^2 \times \ln \alpha.$$  \tag{19}$$

Fig. 7 shows the sample size $m$ (i.e., the number of ICs) required to conclude that a DAT test set is of higher quality for (a) confidence level of 95% and Yield = 85%, (b) confidence level of 90% and Yield = 85%, (c) confidence level of 99% and Yield = 85%, and (d) confidence level of 95% and Yield = 95%.

C. Application Scenarios

There are two application scenarios of DAT based on behavior distribution changes. If failure characteristics change slowly over time, DAT can be applied over time to improve test
quality. Since the tester response of the failing chips and their lot numbers are normally stored and diagnosed, the failure characteristics can be tracked over time by analyzing the diagnosis results of failing chips using behavior attribution (Section II). To determine whether the change is slow enough, (19) and Fig. 7 can be used. In other words, if 1) the sample size is large enough so that the change in failure characteristics can be determined with high confidence; and 2) the time needed to customize the test set is smaller than the time for manufacturing/collating the failing chips, then distribution change is considered sufficiently slow. Alternatively, if the failure characteristics remain static, then the test set can be one-time tuned to match the unchanging failure characteristics using DAT. Under either scenario, DAT can be used to minimize DPPM for a given constraint on test costs, or alternatively, ensure that DPPM does not exceed some pre-determined threshold.

VI. CONCLUSION

A DAT methodology that ensures a desired level of quality for chip-logic circuits on a per-design basis by changing test content to match diagnosed-deferred failure characteristics was described. DAT began by partitioning the failure population of a manufactured design into several behavior categories using diagnosis results. Then, based on the occurrence rate of each behavior category, DPPM was estimated using a new and novel data-driven defect-level model. Also, using the resulting behavior distribution, the test set was changed using a test-selection methodology to reduce DPPM.

The efficacy of DAT was demonstrated using detailed circuit-level simulations. Specifically, defects of various types were injected into the layout of a benchmark circuit, extracted to create a circuit-level netlist that accurately reflects the presence of the defect, and then simulated to produce virtual test responses. Applying DAT to 25% of the virtual fail population showed that escapes from the remaining 75% could be reduced by 30%. Although not verifiable, application to actual chips also demonstrated quality improvement.

In addition, a formulation was derived for computing the number of tested ICs required to confidently conclude that a manufactured design into several behavior categories using diagnosis results. Then, based on the occurrence rate of each behavior category, DPPM was estimated using a new and novel data-driven defect-level model. Also, using the resulting behavior distribution, the test set was changed using a test-selection methodology to reduce DPPM.

The efficacy of DAT was demonstrated using detailed circuit-level simulations. Specifically, defects of various types were injected into the layout of a benchmark circuit, extracted to create a circuit-level netlist that accurately reflects the presence of the defect, and then simulated to produce virtual test responses. Applying DAT to 25% of the virtual fail population showed that escapes from the remaining 75% could be reduced by 30%. Although not verifiable, application to actual chips also demonstrated quality improvement.

In addition, a formulation was derived for computing the number of tested ICs required to confidently conclude that a manufactured design into several behavior categories using diagnosis results. Then, based on the occurrence rate of each behavior category, DPPM was estimated using a new and novel data-driven defect-level model. Also, using the resulting behavior distribution, the test set was changed using a test-selection methodology to reduce DPPM.

The efficacy of DAT was demonstrated using detailed circuit-level simulations. Specifically, defects of various types were injected into the layout of a benchmark circuit, extracted to create a circuit-level netlist that accurately reflects the presence of the defect, and then simulated to produce virtual test responses. Applying DAT to 25% of the virtual fail population showed that escapes from the remaining 75% could be reduced by 30%. Although not verifiable, application to actual chips also demonstrated quality improvement.

In addition, a formulation was derived for computing the number of tested ICs required to confidently conclude that a manufactured design into several behavior categories using diagnosis results. Then, based on the occurrence rate of each behavior category, DPPM was estimated using a new and novel data-driven defect-level model. Also, using the resulting behavior distribution, the test set was changed using a test-selection methodology to reduce DPPM.

The efficacy of DAT was demonstrated using detailed circuit-level simulations. Specifically, defects of various types were injected into the layout of a benchmark circuit, extracted to create a circuit-level netlist that accurately reflects the presence of the defect, and then simulated to produce virtual test responses. Applying DAT to 25% of the virtual fail population showed that escapes from the remaining 75% could be reduced by 30%. Although not verifiable, application to actual chips also demonstrated quality improvement.


Xiaochun Yu (S’08–M’12) received the B.S. degree in microelectronics from Fudan University, Shanghai, China, in 2005, and the M.S. and Ph.D. degrees in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 2008 and 2011, respectively. Since June 2011, she has been with Intel Corporation, Hillsboro, OR, where she is engaged in diagnosis and debugging solutions. Her current research interests include multiple defect diagnosis, layout-aware diagnosis, defect-type distribution estimation, and adaptive testing.

Ronald DeShawn Blanton (S’93–M’95–SM’03–F’09) received the B.S. degree in engineering from Calvin College, Grand Rapids, MI, in 1987, the M.S. degree in electrical engineering from the University of Arizona, Tucson, in 1989, and the Ph.D. degree in computer science and engineering from the University of Michigan, Ann Arbor, in 1995.

He is currently a Professor with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, where he serves as the Director of the Center for Silicon System Implementation, an organization consisting of 22 faculty members and over 80 graduate students focusing on the design and manufacture of silicon-based systems. His current research interests include the test and diagnosis of integrated, heterogeneous systems, and design-, manufacturing-, and test information extraction from tester measurement data.