On the Decline of Testing Efficiency as Fault Coverage Approaches 100%*

Li-C. Wang and M. Ray Mercer
University of Texas at Austin

Sophia W. Kao
LSI Logic, Inc.

Thomas W. Williams
Microelectronics Division, IBM

Abstract

Testing is an indispensable process to weed out the defective parts coming out of the manufacturing process. Traditionally, test generation target on a specific fault model, usually the single stuck-at fault model, to produce tests that are expected to identify defects such as unintended shorts and opens. With this approach, the test quality relies on fortuitous detection of the non-target defects. As the quality demands and circuit sizes increase, the feasibility of test generation on a single fault model becomes questionable. In the paper, we present empirical data from experiments on ISCAS benchmark circuits to demonstrate that using traditional methods the probability of detecting non-target defects drops rapidly as the fault coverage approaches 100%. By assuming surrogates, we explain the mechanism which produces this effect and describe a new test pattern generation approach with better testing efficiency.

1 Introduction

Testing is performed to weed out the defective parts coming out of the manufacturing process. Traditionally, test generation targets on a specific fault model to produce tests that are expected to identify defects such as unintended shorts and opens. There can be an enormous number of possible defects in a circuit. To do a good job, most of the defects should be detected by the test set. Since the test generation and the test application are limited by available resources like memory and time, generating tests for all defects is infeasible. Instead, a relatively small set of abstract defects, namely faults, are constructed, and these faults are targeted to generate the tests. Usually, the test generation stops when all target faults are detected by the tests produced. With this approach, the test quality relies on fortuitous detection of the non-target defects [BUTL90] [BUTL91a] [BUTL91b].

As the quality demands and circuit sizes increase, the effectiveness of test generation based upon a single fault model becomes questionable. For instance, [KAPU92] showed that for the most commonly used model, the single stuck-at fault, the range in defect which part levels can spread over several orders of magnitude. [PARK94] did extensive studies on this issue and demonstrated that at high fault coverage values, it was hard to predict an equally high quality.

The weakness of the single stuck-at fault model for obtaining good testing quality reveals that one model may bias the selection of tests and reduce the probability of detecting some non-target defects. One possibility to remedy this bias is to use more fault models to generate more tests. For instance, [MAX92A] presents a method using a collection of functional, $DDQ$, and scan tests and reported better results than merely the single stuck-at fault tests. Since each fault model represents a different perspective toward the total defect space, it is likely that an undetected defect left by one test set can be captured by another.

In this paper, we conduct extensive experiments to explore the relationship between the test space for faults and the test space for non-target defects by assuming surrogates in place of the defects. Our data indicate that as the fault coverage approaches 100%, the probability of fortuitous detection of non-target defects drops rapidly. As a result, the effectiveness of tests drop as the fault coverage increases and the defective part level is worse than expected. A new

*This work was supported in part by the Semiconductor Research Corporation under contract SRC 93-DP-142, and in part by the Innovative Science and Technology Office of the Strategic Defense Initiative Organization, administered through the Office of Naval Research under contract # N00014-92-J-1723.
and more effective test generation method is then presented.

2 Background

2.1 Fault Model

Faults (or logical faults) represent the effect of physical defects on the behavior of the modeled circuit. In modeling circuits, we differentiate between the logic function and timing, and hence we also distinguish between faults that affect the logic function and delay faults that affect the speed. Physical defects are difficult to predict and manage. On the other hand, faults are well-defined and their detection rules are clear, although that may not be simple. For testing purposes, usually faults are used.

The most commonly used model is the single stuck-at fault. Other popular fault models include bridging faults, transition faults (gate delay faults), path delay faults, functional faults, etc [ABR90]. In this paper, we consider three fault models — stuck-at faults, bridging faults and transition faults. For bridging, we assume non-feedback types of faults since those causing feedback are easier to detect [MEI74] [MILL88]. We consider transition faults instead of path delay faults because the later in actual commercial circuits is intractable due to its exponential grow in term of the circuit size.

2.2 Test Generation

![Figure 1: An example for test generation](image)

Usually, tests are generated based upon faults. To illustrate the process, suppose that IN2 in the Figure 1 has a stuck-at fault. A test for this fault must satisfies two conditions — (1) to produce a difference at the fault site between its correct and incorrect behaviors and then (2) to allow this difference to be observed at some of the primary outputs of the circuit. The former condition is called excitation and the later observation. In our study, the Ordered Binary Decision Diagram (OBDD) [BRYA86] was used to compute the whole test space for each target fault [BUTT90]. Note that for different types of defects, excitation conditions may differ, but observation conditions depend only upon the location of the defect and not the defect type.

2.3 Faults, Defects, and Surrogates

![Figure 2: An example of distribution of test spaces for target faults and surrogates](image)

A defect is a flaw in a circuit. A fault model is a hypothesis of how defects affect the circuit behavior. Given a fault model, a set of faults is derived, called target faults. Then, tests are generated on these faults. Defects are categorized into those which are mapped directly onto modeled faults and the others that are not. We call the former target defects, and the later non-target defects. Since detecting a particular non-target defect is not ensured, the accuracy of the estimation for defect coverage depends on fortuitous detection of the non-target defects. Note that there can be an enormous number of defects in a circuit. While lacking a clear uniform model to capture all defects, to study the fortuitous detection, in practice we assume surrogates. Surrogates are different modeled defects from those target ones for test generation; they represent a different set of possible defects than those selected as faults. In our study, we used two sets of surrogates, non-feedback AND bridging defects and transition (gate delay defects).

We next consider the set of all tests which detect a fault and their relationship with the set of all tests for a defect. There are three possible relationships be-
between tests for target faults and those for non-target defects. Figure 2 illustrates the three cases. \( F_i \) represents the set of all possible tests for fault \( f_i \) and \( S_i \) represents the set of all possible tests for surrogate \( s_i \). For instance, S3 dominates F2, F1 intersects S1, and S2 is disjoint from both F1 and F2. s3 is ensured to be detected by a test for F2, S1 can be fortuitously detected by a test for F1, and S2 cannot be detected by a test for any target fault. The interaction among the target fault test spaces and surrogate test spaces affects the accuracy of the estimation for non-target defect coverage using target fault coverage.

2.4 Defective Part Level Prediction

After a set of tests is generated, the quality of this test set should be evaluated. If the quality is not satisfied, more tests should be added into the set. The testing quality is measured by its defect coverage (DC) and then with the manufacturing yield \( Y \), is transformed into the defective part level (DL). The defect coverage is the probability that the set detects any defects in the circuit. The yield is the probability that a manufactured circuit is defect-free. The transformation into defective part level is called defective part level prediction. There are many models for such a prediction and perhaps the most popular one is the Williams-Brown model \( DL = 1 - Y^{1-DC} \) [WILL81]. Other popular models include the one presented in [AGR82]. The defective part level usually is measured in term of “parts per million” (PPM) which is the number of possible defective parts in a million. To obtain DL, we need to know DC and Y. Y can be obtained from empirical data. Without know the distribution of defects, usually DC is estimated by the fault coverage (FC) which is the probability that the test set detects a fault in the fault set. Since \( FC \neq DC \), the resulting DL can be inaccurate. The inaccuracy depends on how well a fault model captures realistic defects. With such a testing setup, the quality is limited by 100% fault coverage. Therefore, if a fault model is not good enough to capture most defects, this approach may lead to an unsatisfactory actual defective part level.

3 Problems in the Traditional Testing Method

Using traditional testing methods, a fault model is assumed. Currently, the most commonly used one is the single stuck-at fault model. A set of faults is derived from the fault model. Then, each test generated ensures the detection of a particular fault. Normally, a fault coverage is set as the testing goal, and a set of tests are constructed to guarantee that goal. For instance, for a model with 100 faults and a testing goal of 99% fault coverage, a set with 99 tests, each for a different fault, ensures the goal. Note that fault coverage is not the only factor to affect testing quality. The quality of tests depends on the fortuitous detection of non-target defects. Therefore, if the test set selected results in many non-target defects undetected, the final quality can be disappointing even for 100% fault coverage. In this section, we first describe an example to illustrate this concept, and then we present experiment results on a real circuit.

3.1 The Test Spaces of Faults and Defects, an Example

![Figure 3: A simple circuit example](image)

Figure 4: The truth tables after each step of test application

Our first study concerns the interaction between the test space for modeled faults and the test space for non-target defects. We first use an example to illustrate how the experiment will be performed. Figure 3 depicts a simple circuit. Assume that the single stuck-at fault model is used. In this example, we consider a reduced check point fault set (see [ABR90]) which consists of six stuck-at faults (SAF), A0,A1,B1,C0,C1,D1.
where A0 represents “line A stuck at 0” and so on. We also consider wired-OR type bridging surrogates (BS) between line pairs AB, CD, and BC. There are 4 inputs to the circuit and hence \(2^4 = 16\) possible tests in total. The test set for each fault can be constructed as described in the previous section. For instance, only tests 1100, 1101, 1110 can detect fault A0 and only tests 0101, 1010 can detect surrogate BC. The test distribution for all stuck-at faults is illustrated in Figure 4-(a). The dark entries (including the black one) in the figure represent those tests which will detect at least one stuck-at fault. Therefore, these tests are the only ones which can be selected when test pattern generation begins. In summary, before test pattern generation begins, there are 14 possible tests, any one of which could be selected as the first test (depending upon the first fault selected). Similarly, before any test is selected, there are six potential tests for the three surrogates as shown in Figure 4-(b). Note that the set of test for all faults at this point dominates the set of tests for all surrogates. The numbers are listed in the first column of Table 1.

Table 1: Illustration of each test application step

<table>
<thead>
<tr>
<th>Test applied (ABCD=)</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faults detected</td>
<td>none</td>
<td>0101</td>
<td>1000</td>
<td>1100</td>
<td>0110</td>
<td>1011</td>
</tr>
<tr>
<td># of possible SAF tests, after</td>
<td>14</td>
<td>11</td>
<td>9</td>
<td>6</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td># of possible BS tests, after</td>
<td>6</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td># of tests in common, after</td>
<td>6</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td># of SAFs left undetected</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Fault coverage, after</td>
<td>(\frac{0}{3})</td>
<td>(\frac{0}{3})</td>
<td>(\frac{0}{3})</td>
<td>(\frac{1}{3})</td>
<td>(\frac{1}{3})</td>
<td>(\frac{0}{3})</td>
</tr>
<tr>
<td># of BSs left undetected</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Surrogate coverage, after</td>
<td>(\frac{0}{3})</td>
<td>(\frac{1}{3})</td>
<td>(\frac{1}{3})</td>
<td>(\frac{2}{3})</td>
<td>(\frac{2}{3})</td>
<td>(\frac{2}{3})</td>
</tr>
</tbody>
</table>

Figure 6: The intersection of test space between stuck-at faults and bridging surrogates, the example

Figure 5: (Cont.) The truth tables after each step of test application
Table 1. The surrogate coverage is computed similarly. Figure 6 describes the relationship between the number of tests in the test space and the fault coverage, i.e., at a particular fault coverage, how many potential tests remain for stuck-at faults, how many remain for bridging faults, and how many of them are in common. From this figure, we can easily see if a test applied fortuitously detects surrogate(s) and how likely this is to occur. For example, because the size of the intersected set is zero for the last two tests, no fortuitous detection is possible. Next, we will present similar results for two real circuits.

3.2 The Test Spaces of Faults and Defects, Real Circuits

Figure 7: The intersection of test space between stuck-at faults and bridging surrogates, C432

Figure 8: The intersection of test space between stuck-at faults and bridging surrogates, only for fault coverage > 90%, C432

Here, we conduct the study for circuits C432 and C499. 650 wired-AND bridging defects are randomly generated as surrogates in both cases. Using OBDD,

the test space is computed for each fault and surrogate left undetected after each test is applied. The union of the test sets for all remaining faults, the union of the test sets for all remaining surrogates, and the intersection of these two supersets are also computed. At each step of test application, an undetected fault is randomly picked, and the test for that fault is randomly selected from its test set. Note that this test may detect other faults and all detected faults are removed before the next step. The removal of non-target faults detected by a test is called "fault-dropping," and is common in current testing practice.

Due to resource limitations, we only computed the curves for one test set. Figure 7 shows the result, which is similar to Figure 6. Figures 8 and 9 show the last portion of the curves for fault coverage greater than 90% and 95%, respectively. Several things can be observed from these figures.

- Both the fault set and the surrogate set have large test spaces.
- The bridging surrogate set always has a larger test space in the case for C432.
- The set of all possible tests for all stuck-at faults drops several orders of magnitude as fault coverage increases. Therefore, test generation for later faults may be harder as fault coverage becomes higher.
- As fault coverage increases, only part of the tests from the stuck-at fault test space can detect the bridging surrogate. Therefore, the conditional probability that a test for a stuck-at fault fortuitously detects a bridging surrogate decreases dramatically as the fault coverage approaches 100%.

Figure 9: The intersection of test space between stuck-at faults and bridging surrogates, only for fault coverage > 95%, C499
At the end, even though the stuck-at fault test space is zero (100% fault coverage), the bridging test space is still large because several bridging surrogates are undetected.

3.3 What Happens as the Fault Coverage Approaches 100%, an Example

![Defective part level curves, the simple example](image)

Figure 10: Defective part level curves, the simple example

Our next study concerns the relationship between defective part level and fault coverage. Following the simple example in section 3.1, the relationship between fault coverage and defective part level is shown in Figure 10. Note that unlike previous figures where numbers of tests are drawn against fault coverage, for defective part level curves, the x-axis is “1 – fault coverage”. To compute the defective part level (DL) at a particular defect coverage DC, we use the Williams-Brown model $DL = 1 - Y^{1-DC}$ where $Y = 0.5$ is assumed here. For the curve obtained by using fault coverage in place of defect coverage in the Williams-Brown model, we mark it as “WB-Model” and refer to this graph as the “predicted defective part level.” However, since fault coverage is only an estimate of defect coverage in our example, the real defective part level curve is different. To compute the defect coverage after each step, we assume that the 3 surrogates are those that may possibly go wrong. Hence, we assume an “artificial defect universe” that contains only 3 defects. With this assumption, the defective coverage is the surrogate coverage. Again, using the Williams-Brown model we plot a curve and refer to it as the “real defective part level.” The fault coverages and surrogate coverages are shown in Table 1. The real and predicted curves are compared in Figure 10. From this figure we observe that even though the fault coverage is 100%, the defect coverage is not and hence results in a higher defective part level than expected. Most importantly, after applying the second test, the defective part level remains unchanged. If we think the effectiveness of a test is the amount of defective part level it reduces with respect to the previous test, the horizontal segment in the “real defective part level” curve indicates that the effectiveness of test 3, 4 and 5 is zero. In other words, they detect no undetected defect and hence their application is totally futile.

3.4 What Happens as the Fault Coverage Approaches 100%, a Real Circuit

![Results on bridging surrogates, C432 (Ave is the mean, Std is the standard deviation)](image)

Figure 11: Results on bridging surrogates, C432 (Ave is the mean, Std is the standard deviation)

![Comparison between fault dropping and not](image)

Figure 12: Comparison between fault dropping and not

Now we move to study the relationship between defective part level and fault coverage for a real circuit, C432. As before, 650 bridging surrogates are randomly picked. To get more insight and to avoid bias toward a particular testing software implementation, thousands of test sets are randomly constructed.
Therefore, at each particular fault coverage, an average defective part level and its standard deviation can be computed. Then, instead of showing one curve as in Figure 10, three curves are presented in Figure 11 — one for average and two for average plus/minus twice the standard deviation. Note that fault-dropping is not used here, i.e. the number of tests is equal to the number of targeted faults.

Figure 12 presents the average curve for fault dropping with a comparison to that for using no fault dropping. It is clear that although both cases guarantee 100% fault coverage, at the end of test application the resulting quality using fault dropping is substantially worse. Intuitively, with fault dropping, fewer tests are applied and hence more non-target defects are left undetected. However, fault dropping is commonly used in testing since applying more tests requires a higher test application cost. There is a trade-off between test quality and cost using the traditional testing method, and this trade-off is not clear since a test applied later can be much less effective than an earlier test — in terms of the fortuitous detection of non-target defects.

In addition to the bridging surrogates, we also studied the case for transition surrogates. Results are shown in Figure 13. The numbers of transition surrogates for C432 are 320 which is twice the numbers of gates since each gate output can be slow-to-rise and slow-to-fall.

Previously, [PARK94] studied the limitations of the single stuck-at fault model. Their conclusions about the single stuck-at fault model include 1) test quality varies as fault coverage approaches 100%, 2) fault dropping is questionable, and 3) using fault coverage in place of defect coverage with the William-Brown model is also questionable. Beside confirming their conclusions, we found that transition surrogates consistently result in a higher uncertainty with respect to test quality and test prediction than bridging surrogates. This is expectable since usually the transition test space is larger (which require two tests for a transition defect).

From the three figures shown, we can conclude the following:

- Fault dropping results in a much worse testing quality as the fault coverage approaches 100%.
- For bridging surrogates, stuck-at fault tests lose their effectiveness dramatically as the fault coverage approaches 100%.
- For transition surrogates, the quality of test sets varies significantly for high fault coverages.

4 Potential Solutions

The intuitive way to overcome the difficulty of using a single fault model for high quality testing is to include additional fault model(s). This approach has been addressed by researchers in recent years [[MAX91] [MAX92B] [MAX92A] [PER92]]. Another more fundamental approach may be to use no fault model at all. However, since aiming at the whole defect testing space is usually impractical due to its high complexity, how to develop a good testing method without using any fault model is unclear. This section explains the differences between these two approaches.

4.1 Using a Collection of Different Fault Models

![Figure 14: Illustration of test vs. defect-detection spaces using fault model approaches](image-url)
We first consider the method of using a collection of different fault models. Remember that faults result in a different test space from actual defects. From the illustration in Figure 2 before, we know that some defects may have test spaces totally different from the test spaces of all faults. Therefore, these defects will not be detected by any test set generated using that particular fault model. Even though the test space of a defect is partially covered by the test spaces of some faults, from the simple example illustrated in section 3, we saw that this defect can still be missed at the end of test application (consider that defect CD is missed in that simple example). In order to cover more undetected defects after applying a set of tests, more tests are required. The simplest way is to generate more tests using the same fault model. However, this method can be very ineffective since a particular fault model biases the selection of tests toward some portion of the test space, and it is very likely that tests generated from that fault model never reach the test spaces of some other defects. Figure 14(a) illustrates this concept. With a single fault model, Figure 14(a) indicates the situation that some defects can never be detected. From this figure, we clearly see the bias of test selection imposed by the fault model; a test can only be selected from the dark rectangle regions.

Figure 14(b) shows the case with an additional fault model. Since the second fault model represents a different perspective of the total test space from the original one, any uncovered defect before may be covered now. Note that the bias of test selection is reduced in the case of using two fault models but is not totally removed. As a result, some defects are still not covered.

4.2 Using no Specific Fault Model

The method of using a collection of fault models has been successful in some cases. However, there are two fundamental problems for that approach. First, the bias in test selection can be reduced via this way, but can never be eliminated. Without knowing the distribution of real defects some of them are very likely to be missed. How many different fault models should be included is always a question. Second, the defect distribution highly depends on the technology used in the manufacturing process. Therefore, an effective set of fault models constructed today may become obsolete as the manufacturing technology changes.

"How about using no specific fault model?" Regardless of how easy it may sound, without fault models, testing can be very difficult since the potential test space is huge and how to select tests to detect most of the defects is unknown. Uniformly random testing is an example of using no fault model. In this method, test generation has almost no cost but it requires an extremely long test length to achieve a desired quality. Except for very small circuits, the method is impractical. Many improvements on uniformly random testing are possible and can be found in [ABR90]. Here we mention the method to illustrate the idea of unbiased testing. As shown in Figure 14, a fault model biases the selection of tests toward some predetermined test spaces; and if a defect test space falls beyond those regions, it will never be detected by any tests generated from that fault model. By using no fault model, this bias can be removed. As an example, Figure 15(a) demonstrates the concept of an unbiased method — uniformly random testing. Comparing this figure with Figure 14, we can see that now all defect testing spaces are potentially covered, i.e. every test has a chance to be selected. However, in this extreme case, the potential test space is the total space. One important thing to keep in mind when comparing these figures is that a larger test space implies less effort on test generation but much higher cost on test application. For instance, in Figure 15(a), there is almost no cost associated with test generation since any test can be selected, but in order to cover most of the defects, more tests are required than the cases in Figure 14. The advantage here is that there is no restriction on test selection and hence eventually all defects can be covered. The question is: "when (after how many tests) this will happen?"

15(b) represents an improved case of unbiased testing, where the potential test spaces remain uniformly distributed over the total space but now only a small portion of the total space is targeted. Therefore, the advantage just mentioned can be kept, with
a great reduction of test application cost. 15-(b) illustrates the basic concepts of unbiased testing. A formal definition of unbiased testing can be found in [WMW95].

We conclude this section with two remarks about the evaluation of test quality. First, although in this case fault models are not used for test generation, faults can be selected to evaluate the quality of unbiased tests. We note that the bias introduced by fault models is much less as they are used for evaluation than for test generation since an unbiased test set with 100% fault coverage can have many tests beyond the test spaces of all faults. Second, perhaps an even better method for test quality evaluation is to use a “fault-independent” approach such as the Inductive Fault Analysis described in [FER88]. With an analysis of the circuit layout, a potential defect set can be constructed, and this set can provide an “unbiased” test quality evaluation. Currently, we use faults for the purpose of test pattern sequence quality evaluation and later we plan to develop a realistic defect simulator.

5 Tests that Guarantee Defect Observation

In reality, it is not easy to prove that a particular algorithm for test generation achieves purely unbiased testing. Therefore, we will develop only “pseudo-unbiased” testing methods. In this section, we present one such scheme — Guaranteed Observation tests. Remember that tests from different fault models only differ in their excitation criteria. For a fixed site, regardless of what defects may occur, their observation criteria are the same. To remove the bias of test selection from one fault model with respect to the others, we will eliminate the excitation criteria and only ensure the observation requirements for each logical site. If a defect occurrence is restricted inside each logical site, then this method introduces no bias. However, since it is usually a good assumption but not totally true, the method is pseudo-unbiased and provides a good approximation for unbiased testing.

We obtained the pseudo-unbiased tests by computing the whole boolean difference space for each site using the OBDD tool we built. As usual, thousands of test sets are selected randomly. However, for each test set, we ensure that if possible, every site should be observed at least once. In order to have a fair comparison, the cardinality of the test set here is equal to the number of tests generated using the single stuck-at fault model. Again, the average results and their variances for defective part level curves were plotted. The average results are presented in Figures 16 and 17, with those average curves from stuck-at fault tests and curve from William-Brown model for comparison. One question which remains is “What target fault set should we select to compute the fault coverage?” In the figures, for C432, the fault coverage is obtained using the single stuck-at fault model.

Figures 16 and 17 confirm our previous conjecture that an unbiased test set is superior in terms of reducing the defective part level. Hence, an unbiased method is more effective. Perhaps the most surprising result is the case when using transition surrogates for C432, where the defective part level is better than that predicted by William-Brown model.
6 Conclusion

In this paper, we presented results on the decline of testing efficiency as the fault coverage approaches 100%. From the test space experiments, we observe that the probability of fortuitous detection drops as more tests are applied. We also observe from the defective part level study that the effectiveness of tests applied later is less than those applied earlier. We pointed out that the problem may be associated with the biased selection of tests using the traditional testing method. The concept of unbiased testing was illustrated and one such method was presented. These results for “pseudo unbiased testing” show superior performance to more traditional testing methods.

References


