

Analyzing the Effectiveness of Multiple-Detect Test Sets

R. D. (Shawn) Blanton, Kumar N. Dwarakanath and Anirudh B. Shah

Center for Silicon System Implementation
Dept. of Electrical and Computer Engineering
Carnegie Mellon University, Pittsburgh PA 15213
E-mail: {blanton, nari, abshah}@ece.cmu.edu

Abstract

Multiple-detect test sets have been shown to be effective in lowering defect level. Other researchers have noted that observing the effects of a defect can be controlled by sensitizing affected sites to circuit outputs but defect excitation is inherently probabilistic given a defect's inherent, unknown nature. As a result, test sets that sensitize every signal line multiple times with varying circuit state has a greater probability of detecting a defect. In past work, the entire circuit is considered when varying circuit state from one vector to another for a given signal line. However, it may be possible to improve defect excitation by exploiting the localized nature of many defect types. Specifically, by varying circuit state in the physical region or neighborhood surrounding a line affected by a defect, the defect excitation and therefore detection can be improved. In this paper, we present a method for extracting a physical region surrounding a signal line but more importantly, techniques for analyzing the excitation characteristics of the region. Analysis of 4-detect test sets reveals that 30% to 60% of signal line regions do not achieve at least four unique states, indicating opportunity to further reduce defect level.

1 Introduction

A multiple-detect (or N -detect) test set requires that every single stuck-line (SSL) fault be detected multiple times [1, 2]. The original work in this area [1] revealed that defect coverage was significantly improved in a test chip when N -detect test sets were employed. Subsequent work [3,4] provided both a theoretical and statistical evidence supporting the effectiveness of multiple-detect test sets.

Defect detection is both a deterministic and probabilistic process [3]. Specifically, observation of the effects of defects (*i.e.* errors) requires path sensitization from the affected signal lines to circuit outputs. Sig-

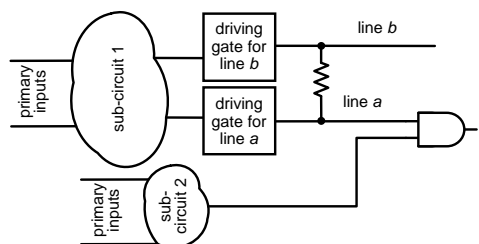


Figure 1: Circuit diagram illustrating the potential ineffectiveness of N -detect test patterns.

nal line sensitization can be easily accomplished using well-known test generation algorithms [5], and therefore is a deterministic task. However, the excitation or activation of a defect is inherently probabilistic since the exact nature of a defect is virtually unknown. Thus, observing a signal line l_i multiple times with varying excitation conditions increases the likelihood of detecting a defect that affects l_i . It therefore follows that test sets that detect every SSL fault multiple times with varying circuit state has an improved defect level over standard 1-detect test sets.

Test patterns for a given SSL fault affecting a signal line l_i in an N -detect test set requires that the excitation conditions be different for every one of the N test patterns. There is however no formal or systematic method utilized for altering the excitation conditions. The only requirement is that the test patterns be unique, that is, for any two test patterns t_i and t_j , it must be true that the logic value for at least one circuit input is different. This approach to N -detect test generation is satisfactory if a defect can physically span a significant portion of the circuit under test. However, if defects are assumed to be localized, then it is more appropriate to restrict the variation of the excitation conditions to the neighborhood or region surrounding the targeted signal line l_i .

For example, consider a bridge defect between two lines a and b as shown in Figure 1. Assume that the

bridge manifests as a dominant bridge fault where b , the aggressor, always imposes its logic value on the victim line a . A test that detects this fault requires that signal line a be sensitized to some output (e.g. Z) while at the same time, a and b are driven to opposite logic values by their respective driving gates. The deterministic portion of N -detect test generation requires that line a be tested for both the stuck-0 and stuck-1 fault, each with at least N different test patterns. The excitation requirements, on the other hand, only require that the N patterns be unique from the perspective of the logic values assigned to the primary inputs. Ideally, for this particular fault, it would be desirable to find test patterns that alter the inputs of sub-circuit one since this increases the likelihood of generating a test pattern that excites the dominant bridge fault between lines a and b . The worst-case situation occurs however when all N test patterns stem from changes made to the inputs of sub-circuit two. For this latter situation, the probability of excitation of the bridge fault is not increased although the criteria for N -detect has been satisfied.

Assuming a defect is localized, it is logical to assume that its excitation conditions are localized as well to the physical neighborhood or region surrounding the signal line(s) affected by the defect. Our aim is to increase the likelihood of defect detection by confining the excitation requirements of N -detect test patterns to the physical, three-dimensional region surrounding the targeted signal line l_i . In particular, by explicitly modeling various types of excitation conditions using fault tuples [6, 7], we can (1) grade existing test sets, including N -detect test sets, to measure their ability to vary excitation conditions within the regions of targeted signal lines, and (2) generate test patterns that satisfy random or deterministic excitation conditions within regions for the purposes of improving defect excitation and therefore, detection.

In this paper, we show that N -detect test patterns generated for a signal line l_i with no restrictions on the excitation conditions inadequately varies the values of signal lines within the physical region $R_p(l_i)$ containing l_i . This observation establishes the need for a more effective means of guiding the N -detect test generation process. Previous research work [8–10] has primarily focussed on evaluating N -detect tests based on a specific defect surrogate.

The rest of this paper is organized as follows. Section 2 describes how physical regions are extracted from a layout a description of the circuit under test. Various options for utilizing the regions for defect excitation are described in section 3. In section 4, we investigate how well N -detect tests exercise physical regions for a 4-bit ALU and several ITC99 benchmark circuits [11]. Fi-

nally, in section 5, we draw conclusions and outline areas of future work.

2 Region Extraction

In this section, we describe our methodology for extracting the physical neighborhood or region of a signal line from its layout description.

There has been a significant body of work that has examined the physical layout of a circuit to identify the most likely defect sites and types [12–16]. Our purpose is not to duplicate that work but leverage it for the objectives described in this paper. LANNE, a LAYout based Net Neighborhood Extractor, was developed to derive physical, three-dimensional regions. The tool utilizes a publicly-available design exchange format (DEF) parser [17]. LANNE accepts a layout description of a circuit C in DEF and a user-provided region radius r . As output, LANNE produces a physical region R_p of radius r for each logical line l_i in the gate-level description of C .

A region $R_p(l_i) = \{l_1, l_2, l_3, \dots, l_n\}$ is a set of logical signal lines, where each $l_j \in R_p(l_i)$ has a corresponding net n_j such that the distance d_{ij} between n_i and n_j is at most r , that is, $d_{ij} \leq r$. A net n_i is the physical correspondent of the logic-level signal line l_i . Specifically, a net $n_i = \{s_{i1}, s_{i2}, \dots, s_{im}\}$ is a set of m segments that consists primarily of metal interconnects and vias. We obtain the region $R_p(l_i)$ for a line l_i by extracting the region for each segment $s_{ij} \in n_i$.

In general, the distance between any two segments is determined by computing the distance between the polygons representing the segments. For example, consider the layout shown in Figure 2(a). The polygons show the physical layout of two parallel nets. The line AB shows the line of symmetry for each rectangle. The distance d_{ij} between the nets n_i and n_j is determined by computing the distance between the lines AB, i.e., the lines of symmetry of the polygon for net n_i and the polygon for net n_j . There are special cases when the distance computation is somewhat complicated. For example, consider the layout of two nets shown in Figure 2(b). The distance d_{ij} between nets n_i and n_j is the distance between the edges of the polygons. For the layout shown in Figure 2(c), distance d_{ij} between nets n_i and n_j is the distance between the edge of the polygon for n_i and the line of symmetry for the polygon for n_j . Figure 2(d) illustrates the distance between nets in different layers. The polygons show the layout of the two nets n_i and n_j in metal layers i and $i + 1$. The distance d_{ij} between n_i and n_j is the distance between the lines of symmetries of the two polygons.

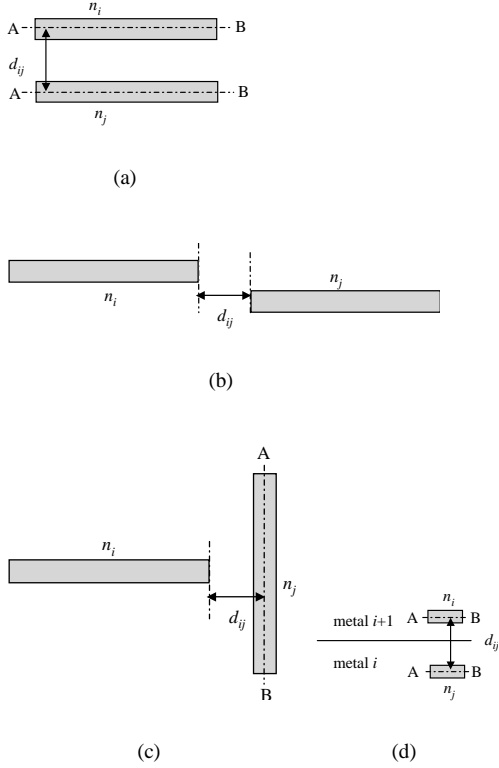


Figure 2: Different cases for computing net distances: (a) distance between lines of symmetry of nets in the same layer, (b) distance between edges of nets in same layer, (c) distance between edge and line of symmetry of nets in same layer and (d) distance between lines of symmetry of nets in different layers.

Figure 3 lists the pseudo-code for extracting the region of a net n_i . For each segment $s_{ij} \in n_i$, all other net segments contained in the three-dimensional cylinder with radius r that surrounds net s_{ij} is identified by computing the distance between segment s_{ij} and every other net segment s_{mn} in the same layer and in adjacent layers. Each logic line l_m with segment s_{mn} with distance $d_{im} \leq r$ is added to $R_p(l_i)$. This process continues for all the segments of the target net n_i .

Consider the layout shown in Figure 4. The target net is n_1 and the target segment is s_{11} . The gray box around the segment s_{11} shows the complete region of radius r around the segment. The segments within r distance of segment s_{11} is the set $\{s_{21}, s_{52}, s_{71}, s_{81}\}$, where s_{21} is a segment of net n_2 , s_{52} is a segment of n_5 , s_{71} is a segment of n_7 and s_{81} is a segment of n_8 . Thus, the region extracted for line l_1 is $R_p(l_1) = \{l_2, l_5, l_7, l_8\}$.

Region analysis was performed using benchmarks

```

net_region( $n_i, L$ )
 $n_i$ : Target net
 $L$ : Layout of circuit under test
begin
1:  $S(n_i)$  = segments of  $n_i$ 
2: foreach segment  $s_{ij}$   $j = 1$  to  $|S(n_i)|$  do
3:  $y$  = layer of  $s_{ij}$  in  $L$ 
4: foreach net  $n_k$   $k = 1$  to number of nets in  $y$  do
5:  $S(n_k)$  = segments of nets  $n_k$  in layer  $y$ 
6: foreach segment  $s_{km} \neq s_{ij}$ ,  $m = 1$  to  $|S(n_k)|$  do
7:  $d$  = distance( $s_{km}, s_{ij}$ )
8: if  $d < r$  then //  $r$  is user-defined region radius
9:   add net( $s_{km}$ ) to  $R_p(l_i)$ 
10: end if
11: end for
12: end for
13: end for
end

```

Figure 3: Pseudo-code for extracting the surrounding region of a net.

b10, b11 and b12 from the ITC99 benchmark suite [11] and the well-known 4-bit ALU [18]. ST Microelectronics 0.18 μm technology was used to automatically generate the layouts using Cadence Silicon Ensemble. Figure 5 shows the minimum, average and maximum sizes (*i.e.* the number of signal lines) of the regions extracted using LANNE. The region radii considered ranged from 0.32 μm to 5.12 μm . As the region radius increases, the average region size for each benchmark does not significantly increase. For example, for benchmark b12 the maximum region size for radius 5.12 μm is 92 but the average region size is significantly lower at 12. Smaller regions are preferable since it implies that defect detection can be accomplished using fewer tests.

3 Region Analysis

In this section, we describe how regions are used to guide the generation of multiple-detect test patterns. First, we describe the alternatives available for selecting excitation condition within the regions. We then discuss the available techniques for actually generating test patterns that satisfy the various excitation conditions.

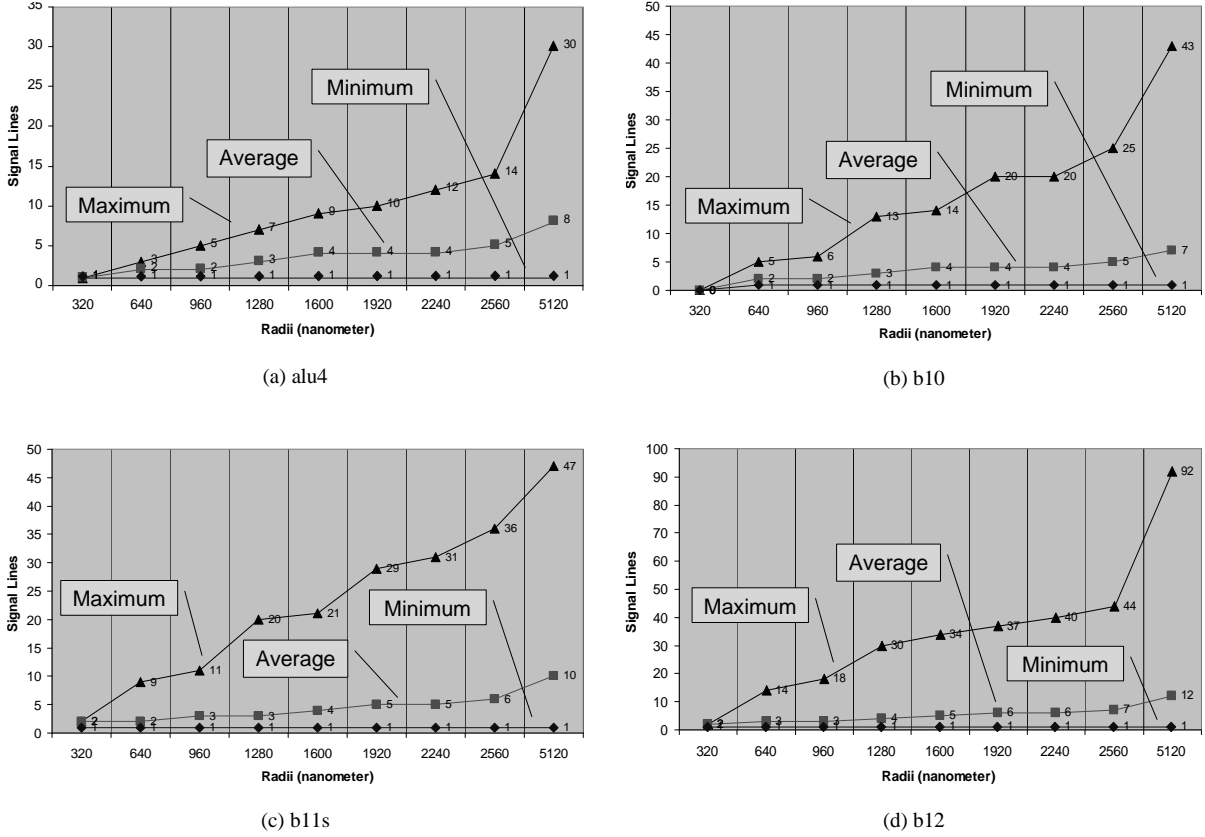


Figure 5: Maximum, minimum and average region sizes for various values of r for (a) alu4, (b) b10, (c) b11s and (d) b12.

3.1 Region Excitation

Given a signal line l_i and its corresponding region $R_p(l_i)$, a question worth considering is how $R_p(l_i)$ should be utilized in order to excite a defect that affects line l_i . There are a number of possibilities.

Obviously, since the nature of the defect is unknown, it is reasonable to employ random excitation conditions where lines within a region are randomly assigned logic values. Note however that the random excitation does not need to be constrained to one clock cycle. For example, it may be desirable to apply sequences of values to the regional lines that include an arbitrary number of rising and falling transitions.

If a region size is small ($|R_p| < 3$ or 4), then exhaustive excitation of the region may be viable. In this case, all $2^{|R_p|}$ combinations of logic values would be applied to the signal lines of R_p . Again, the excitation of the region can be generalized to include sequences. For example, all possible sequences of length two would result in a super-exhaustive exercise of the region involv-

ing $2^{|R_p|} \times (2^{|R_p|} - 1)$ transitions.

Excitation of the region can also be tailored to specific defect types. As mentioned earlier, there has been a significant amount of work dedicated to the extraction of likely defect sites and types [13–16]. Our approach to region extraction is less sophisticated but can be used for similar objectives. For example, we can formulate excitation conditions for a region based on various types of bridge faults (zero-dominating, one-dominating, *etc.*) that can occur between one or more regional lines and the targeted line l_i . For other defect types that include, for example, resistive bridges and opens, the regional lines can be used to alter the defect's environment. For instance, imposing transitions in the region of an interconnect open on l_i may slow down gates driven by l_i or cause them to switch altogether from their correct value [19].

We also recognize that the excitation conditions can vary from region to region. In other words, it not necessary that the excitation condition for a region $R_p(l_i)$ be similar to a region $R_p(l_j)$. A region's size and ex-

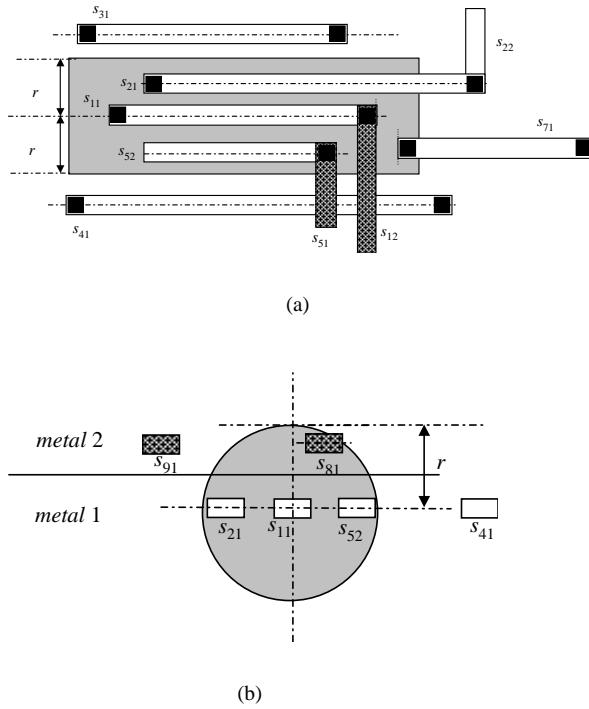


Figure 4: Region extracted LANE for net l_1 (a) intra-layer and (b) inter-layer neighbors.

citation conditions, in general, should be a function of the circuit structure and fabrication technology. For example, it may be more appropriate to have excitation conditions biased towards bridge defects for a dense, hand-drawn layout of a datapath circuit. However, for synthesized logic, random excitation conditions may be sufficient. For this latter case, the size of the region should also be increased to ensure that lines that can influence excitation of the defect affecting line l_i are identified. As already mentioned, the fabrication technology can also play a role. For instance, in technologies where copper is used to construct the nets, it is well-known that open defects are more likely [20]. Thus, the excitation conditions can be biased towards their activation requirements. In addition, the experience gained from a mature fabrication technology through failure analysis can be used to select region size and excitation conditions as well.

3.2 Fault Tuples

The variety of excitation conditions requires a general technique for analysis. However, traditional test analysis algorithms for fault simulation and test generation are limited to just a few fault types and there-

fore a few excitation conditions. Arbitrary signal line conditions can be represented however using fault tuples [6, 7, 21] and IBM pattern faults [22]. Fault tuples are more general in that they can represent conditions across multiple clock cycles and are not limited to a single erroneous line l_i .

A fault tuple is a three tuple $f = (l_i, g/e, T)$, where l_i is any signal line in the circuit, g and e are from the set $\{0, 1, X, U, \text{etc.}\}$, and T is an inter-clock cycle constraint range that describes when l_i must equal g . Fault tuples can be ANDed together to form products, and products can be ORed together to form macrofaults. A macrofault can be used to represent multiple and alternative activation and observation requirements for the logic misbehavior of one or more defects. The complete, formal specification of fault tuples can be found in [23].

Similar to IBM pattern faults, macrofaults can be logically ANDed or ORed together. A product of macrofaults ($M_1 \cdot M_2 \cdot \dots \cdot M_N$) requires a single test or test sequence that detects each macrofault. The difference between a product of macrofaults and a product of fault tuples is that the former represents N separate faulty machines, while the latter is a single faulty machine. In other words, a product of fault tuples represents the set of activation and observation conditions for one or more defects while a product of macrofaults describes the required properties of a test or test sequence. Similarly, a sum of macrofaults ($M_1 + M_2 + \dots + M_N$) requires a test or test sequence that detects at least one M_i .

Macrofaults can be easily used to represent the various excitation conditions for a region. For example, Tables 1 and 2 illustrates exhaustive and 2-line bridge excitation conditions, respectively, for a signal line l_i and its region $R(l_i) = \{l_1, l_2\}$.

4 Simulation Results

For each SSL fault located at signal line l_i , an N -detect test should detect the fault at least N different times with N unique value combinations in the physical region $R_p(l_i)$. As mentioned earlier, such an N -detect test set is meaningful if one assumes that any defect that affects l_i is confined to $R_p(l_i)$. The *effectiveness* of an N -detect test set for a line l_i stuck-at v ($v = 0, 1$) is defined to be the number of combinations or states established in $R_p(l_i)$ when the fault is detected. One way to measure effectiveness is by simulating an N -detect test for an SSL fault l_i stuck-at v against the set of fault tuple based macrofaults that describe all possible excitation conditions for the physical region $R_p(l_i)$.

Excitation description	Macrofault description
$l_1 l_2 = 00$ and l_i stuck-at 1	$\{(l_1, 0/X, i)\}\{(l_2, 0/X, j)\}\{(l_i, X/1, k)\}$
$l_1 l_2 = 01$ and l_i stuck-at 1	$\{(l_1, 0/X, i)\}\{(l_2, 1/X, j)\}\{(l_i, X/1, k)\}$
$l_1 l_2 = 10$ and l_i stuck-at 1	$\{(l_1, 1/X, i)\}\{(l_2, 0/X, j)\}\{(l_i, X/1, k)\}$
$l_1 l_2 = 11$ and l_i stuck-at 1	$\{(l_1, 1/X, i)\}\{(l_2, 1/X, j)\}\{(l_i, X/1, k)\}$
$l_1 l_2 = 00$ and l_i stuck-at 0	$\{(l_1, 0/X, i)\}\{(l_2, 0/X, j)\}\{(l_i, X/0, k)\}$
$l_1 l_2 = 01$ and l_i stuck-at 0	$\{(l_1, 0/X, i)\}\{(l_2, 1/X, j)\}\{(l_i, X/0, k)\}$
$l_1 l_2 = 10$ and l_i stuck-at 0	$\{(l_1, 1/X, i)\}\{(l_2, 0/X, j)\}\{(l_i, X/0, k)\}$
$l_1 l_2 = 11$ and l_i stuck-at 0	$\{(l_1, 1/X, i)\}\{(l_2, 1/X, j)\}\{(l_i, X/0, k)\}$

Table 1: Macrofault representation of exhaustive excitation conditions for $R_p(l_i) = \{l_1, l_2\}$.

Excitation description	Macrofault description
l_1 dominates l_i	$\{(l_1, 0/X, i)(l_i, X/0, i) + (l_1, X/1, i)(l_i, 0/X, i)\}$
l_2 dominates l_i	$\{(l_2, 0/X, i)(l_i, X/0, i) + (l_2, X/1, i)(l_i, 0/X, i)\}$
AND-type(l_1, l_i)	$\{(l_1, 0/X, i)(l_i, X/0, i) + (l_1, X/0, i)(l_i, 0/X, i)\}$
AND-type(l_2, l_i)	$\{(l_2, 0/X, i)(l_i, X/0, i) + (l_2, X/0, i)(l_i, 0/X, i)\}$
OR-type(l_1, l_i)	$\{(l_1, 1/X, i)(l_i, X/1, i) + (l_1, X/1, i)(l_i, 1/X, i)\}$
OR-type(l_2, l_i)	$\{(l_2, 1/X, i)(l_i, X/1, i) + (l_2, X/1, i)(l_i, 1/X, i)\}$

Table 2: Macrofault representation of various bridge excitation conditions for $R_p(l_i) = \{l_1, l_2\}$.

We performed the effectiveness measurement on the four benchmark circuits presented in Section 2, using various values of N (number of multiple detects) and r (region radius). In order to demonstrate our methodology, we present here results for $N = 4$ and region radius of $r = 1.28\mu\text{m}$. All the simulations were performed on a SUN Fire 280R (1.0 GHz) processor with 4.0GB of physical memory running Sun Solaris 8.

The effectiveness of the multiple-detect tests can be better understood if the actual number of unique states possible in each physical region can be obtained. For example, in the layout of b10, the SSL faults $f_1 = \text{U206 stuck-at 0}$ and $f_2 = \text{RTR stuck-at 0}$ have region sizes of six and nine signal lines, respectively, for $r = 1.28\mu\text{m}$. Theoretically, $2^6 = 64$ and $2^9 = 512$ different unique states are possible in the physical regions of f_1 and f_2 , respectively. But due to the logic structure of b10, 72 unique states are possible for f_2 , while only 2 states are possible in f_1 . Thus, an N -detect test set can, at best, obtain only an effectiveness measure of two for f_1 for the physical region considered. We refer to the actual number of unique states of a physical region as the *physical upper bound* of a region. It would have been desirable to compute the physical upper bound for each SSL. However, computing the upper bound for faults in larger circuits is quite intractable. SSL faults that require significant computational effort for deriving their physical upper bound are therefore, not analyzed completely.

Table 3 provides statistics concerning the SSL faults of each benchmark considered for our experiments. Column “No. of SSL Faults” shows the to-

tal number of SSL faults for each benchmark. Column “No. of Faults Dropped” shows the number of SSL faults dropped from complete analysis. Specifically, column “No. of 1-Red” shows the number of SSL faults that are redundant, column “No. of 4-Red” shows the number of SSL faults that do not have 4 distinct tests, and column “Intractable Upper Bounds” shows the number of faults removed because the corresponding upper bounds could not be computed in a reasonable amount of time. Finally, column “No. of Faults Considered” shows the total number of faults considered in our analysis¹.

Table 4 shows the distribution of the physical upper bounds of the SSL faults considered for the experiment. Column “Physical Upper Bounds” lists the number of SSL faults that have physical upper bounds of 1, 2, 3 and ≥ 4 states. Since a 4-detect test set is used for the simulation experiments, only the above-mentioned categories are needed to analyze the effectiveness of the multiple-detect test sets. For example, of the 835 SSL faults that have at least four unique tests for b10, only 498 have four or more unique states possible within their corresponding physical regions. Due to the circuit structure, the remaining 337 faults cannot achieve four unique states even though 4-detect tests exist for all of them.

In this paper, the effectiveness of two different types of multiple-detect test sets are examined. For the

¹The number of faults dropped due to intractable upper bounds is large in the case of benchmarks b11s and b12. Therefore, these benchmarks are not considered in the analysis based on physical upper bounds.

Benchmark	No. of SSL faults	No. of faults dropped			No. of faults considered
		No. of 1-red	No. of 4-red	Intractable upper bounds	
alu4	436	0	1	0	435
b10	856	0	21	0	835
b11s	2318	63	1	—	2254
b12	4772	3	0	—	4769

Table 3: Characteristics of the SSL faults considered for physical upper-bound analysis.

Benchmark	Physical Upper Bound			
	1	2	3	≥ 4
alu4	97	99	53	186
b10	125	133	79	498

Table 4: Categorization of the SSL faults based on the upper bound on the number of states possible in their physical regions.

first type, called “exact N -detect test set,” exactly N tests are generated for each SSL fault, that is, each SSL fault has N tests, no more, no less. Since it is likely an SSL fault is detectable by more than N tests, five different exact- N test sets are created and used to examine effectiveness of exact- N test sets. In the second multiple-detect test set, called “traditional N -detect test set,” an N -detect test set is generated in the traditional fashion as described in Figure 6. Similar to exact N -detect, five different traditional N -detect test sets are created and used to measure effectiveness of this category of N -detect test sets.

traditional $_N(N)$

N : Minimum no. of tests required for each SSL fault

begin

```

1: foreach fault  $s_i$   $i = 1$  to Total do
2:   if detect_count( $s_i$ ) <  $N$  then
3:     required = detect_count( $s_i$ ) -  $N$ 
4:      $T_i$  = generate required tests for  $s_i$ 
5:     fault_simulate( $T_i$ , all faults)
6:     update detect counts of faults
7:     Add  $T_i$  to test set
8:   end if
9: end for

```

end

Figure 6: Pseudocode for generating a traditional N -detect test set.

The details of the test sets that achieve the maximum number of unique states across all the SSL faults for the five different test sets are presented here. Specifically, the faults for which the physical upper bounds could be determined are partitioned into several “buckets,” each representing an upper bound ranging from 1 to ≥ 4 . Figure 7 shows the partitioning of SSL faults

for the two types of N -detect test sets for alu4 and b10. The x -axis is the size of the physical upper bound (*i.e.* the number of possible states in the physical region) and the y -axis shows the number of SSL faults that have the indicated bound size. The chart bars denoted “Physical Upper Bounds” provides the number of SSL faults with the specific physical upper bounds (presented earlier in Table 4). For example, 498 faults in b10 have an upper bound of four or more. The bars titled “Exact 4-Detect” and “Traditional 4-Detect” show the effectiveness of SSL faults achieved by the exact and traditional 4-detect test sets, respectively. For example, for benchmark b10 the exact 4-detect test set had an effectiveness of four for only 356 of the 498 SSL faults that *can* have an effectiveness of four, while the traditional 4-detect test set did appreciably better with 480. In general, the exact-4 test set is quite ineffective. For instance, 186 of the 835 SSL faults in b10 failed to reach their upper bound using the exact-4 test set. For the traditional 4-detect test set, the number of faults that fail to achieve an effectiveness equal to their upper bound reduces to 22, but still falls short of achieving maximum effectiveness.

The following observations can be made about the data presented in Tables 3 and 4 and Figure 7.

1. First, both types of N -detect test sets fail to reach maximum effectiveness for b10 and alu4. For the best traditional N -detect test set, 2.9% ($13/435 \times 100$) and 2.6% ($22/835 \times 100$) of the SSL faults with known upper bounds fail to reach their maximum effectiveness for alu4 and b10, respectively. For these faults, the 4-detect tests are unable to maximally excite their corresponding physical regions. Thus, the resources used to generate and apply a 4-detect test set are wasted and should be better utilized to exercise the physical regions surrounding these faults.
2. Second, resources are also misdirected for SSL faults that have upper bounds smaller than N . Table 3 shows that alu4 and b10 have 435 and 835 SSL faults that have at least four unique tests, respectively. However, Table 4 indicates that only

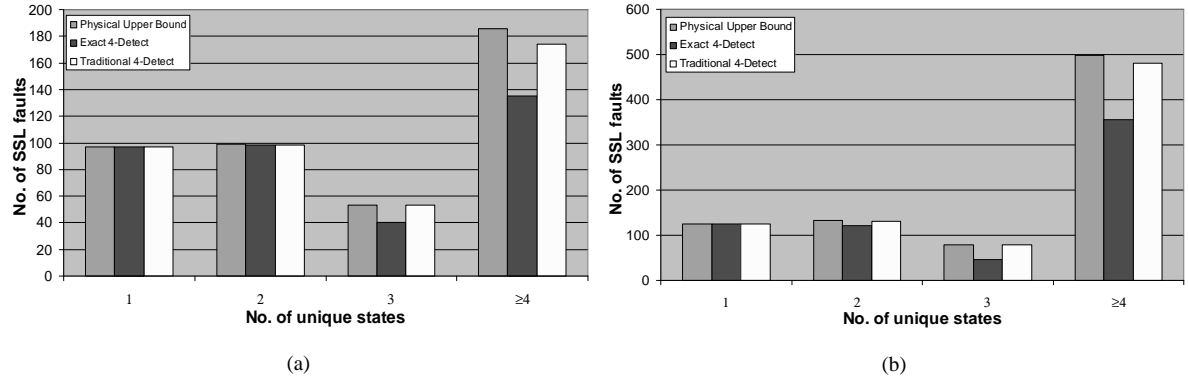


Figure 7: Effectiveness measures obtained by the best exact and traditional 4-detect test sets for (a) alu4 and (b) b10 in context of the achievable effectiveness.

43% ($186/435 \times 100$) and 60% ($498/835 \times 100$), respectively, of these faults can have an effectiveness of four. Thus, the remaining faults are “region-redundant,” that is, generating additional tests that exceeds their upper bound is unnecessary since all possible unique states in the region should have been excited by previous tests.

- Observations one and two lead to the conclusion that N -detect test generation can and should be improved using the physical regions or some other signal line characteristics as a guide. However, there is additional cost associated with identifying the most pertinent signal line characteristics and using them in an ATPG environment. Moreover, the improvement in part quality (*e.g.* escape rate), if any, should be assessed more formally in moving from traditional N -detect sets to those guided by signal line characteristics such as the physical neighborhood. Hence, additional research is needed to address these issues.

Figure 8 reports effectiveness for all benchmark SSL faults that have at least four tests. (Thus, faults for which the physical upper bound is not known are also included here.) The results in Figure 8 stem from the most effective exact and traditional 4-detect test sets from five different runs. Each bar corresponding to a benchmark circuit is partitioned into four sections based on the effectiveness achieved by an exact test set (Figure 8(a)²) and a traditional 4-detect test set (Figure 8(b)). Consider benchmark b12. It has 4769 SSL faults, each of which has at least four unique tests (see Table 3). For the exact-4 test set, 832 faults have an ef-

²The reader should note that for b10, the discrepancy in the number of SSL faults with upper bound 4 is because of certain inadequacies of the tools involved.

fectiveness of one, 1190 have two, 1078 have three, and 1669 have four or more. Similarly, the traditional 4-detect test has 806, 481, 254 and 3228 SSL faults with effectiveness measures of 1, 2, 3, and ≥ 4 , respectively.

Finally, Table 5 provides a comparison between the average number of times an SSL fault is detected and the average effectiveness. Column “Avg. no. of detects” shows the average number of detects of each SSL fault obtained by the best traditional 4-detect test set and column “Avg. Effectiveness” shows the average number of unique states established by the same test set in the physical regions. The data in Table 5 indicates that many faults are detected a significant number of times. However, the average effectiveness is less than a third of the average detections for each benchmark. For example, b12 has a average detect of 129 but an average effectiveness of only 15. Although the data set here is small, we believe this example and the observations made early demonstrate the need to explore new approaches to N -detect test generation.

Benchmark	Avg. no. of detects	Avg. effectiveness
alu4	21	4
b10	39	8
b11s	48	14
b12	129	15

Table 5: Comparison between the average number of times an SSL fault is detected and the average effectiveness.

5 Conclusions

Tests for N -detect test sets are generated using a simple criteria, namely, each fault must be detected by

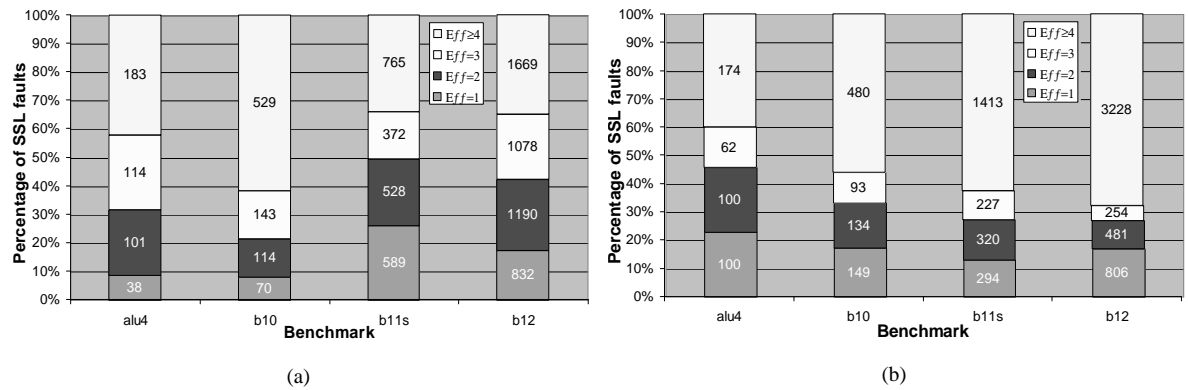


Figure 8: Effectiveness measures achieved by the best (a) exact and (b) traditional 4-detect tests sets for a neighborhood radius of $r = 1.28\mu\text{m}$.

at least N unique tests. In this paper, we described a new approach to measure the effectiveness of an N -detect test set based on the localized nature of most defect types. Simulations show that 4-detect test sets excite less than four value combinations in the localized regions surrounding the signal lines. If these results hold true in general, two important conclusions can be drawn. First, if the regions can have more value combinations, then there is opportunity to improve defect excitation and therefore reduce defect level. Second, if the regions are saturated and cannot support other unique value combinations, then the effort expended to generate and execute 4-detect test sets can be better utilized.

Our future work is exploring new, cost-effective techniques for N -detect test generation and, more importantly, analyzing the ability of the corresponding test sets to improve quality.

6 Acknowledgements

We would like to thank Mr. Amit Kumar for his help with the DEF layout parser. We would especially like to thank Mr. Sunil P. Motaparti for his help with creating the layouts and providing us with tools to reduce the region sizes through implication.

References

[1] S. C. Ma, P. Franco and E. J. McCluskey, “An Experimental Chip to Evaluate Test Techniques Experiment Results,” in *Proc. of International Test Conference*, pp. 663–672, Oct. 1995.

[2] I. Pomeranz and S. Reddy, “Stuck-at Tuple-Detection: A Fault Model Based on Stuck-at

Faults for Improved Defect Coverage,” in *Proc. of 16th VLSI Test Symposium*, pp. 289–294, April 1998.

[3] J. Dworak *et al.*, “Defect-oriented testing and defective-part-level prediction,” *IEEE Design and Test of Computers*, Vol. 18, No. 1, pp. 31–41, Jan./Feb. 2001.

[4] E. J. McCluskey and Chao-Wen Tseng, “Stuck-at Fault Tests Vs. Actual Defects,” in *Proc. of International Test Conference*, pp. 336–342, Oct. 2000.

[5] M. Abromovici, M. A. Breuer and A. D. Friedman, *Digital Systems Testing and Testable Design*, IEEE Press, Piscataway, NJ, 1990.

[6] K. N. Dwarakanath and R. D. Blanton, “Universal Fault Simulation Using Fault Tuples,” in *Proc. of 37th Design Automation Conference*, pp. 786–789, June 2000.

[7] R. Desineni, K. N. Dwarakanath and R. D. Blanton, “Universal Test Generation Using Fault Tuples,” in *Proc. of International Test Conference*, pp. 812–819, Oct. 2000.

[8] Chao-Wen Tseng and others, “An Evaluation of Pseudo Random Testing for Detecting Real Defects,” in *Proc. of 19th VLSI Test Symposium*, pp. 404–409, Apr. 2001.

[9] I. Polian and others, “Sequential n-detection criteria: Keep it simple!,” in *Proc. of Eight On-Line Testing Workshop*, pp. 189–190, 2002.

[10] I. Pomeranz and S. M. Reddy, “On N-Detection Test Sets and Variable N-Detection Test Sets for Transition Faults,” *IEEE Transactions on*

Computer-Aided Design of Integrated Circuits and Systems, Vol. 19, No. 3, pp. 372–383, Mar. 2002.

- [11] S. Davidson, “Panel 6: ITC’99 Benchmark Circuits - Preliminary Results,” in *Proc. of International Test Conference*, p. 1125, Sept. 1999.
- [12] H. Walker and S. W. Director, “VLASIC: A Catastrophic Fault Yield Simulator for Integrated Circuits,” *IEEE Trans. on Computer-Aided Design of Integrated Circuits*, Vol. 5, No. 4, pp. 541–556, Oct. 1986.
- [13] P. Nigh and W. Maly, “Layout-driven Test Generation,” in *Proc. of International Conference on Computer-Aided Design*, pp. 154–157, Nov. 1989.
- [14] H. Xue, C. Di and J. A. G. Jess, “A Net-Oriented Method for Realistic Fault Analysis,” in *Proc. of International Conference on Computer-Aided Design*, pp. 78–83, Nov. 1993.
- [15] T. Liu *et al.*, “Test Generation and Scheduling for Layout-Based Detection of Bridge Faults in Interconnects,” *IEEE Trans. of VLSI Systems*, Vol. 7, No. 1, pp. 48–55, March 1999.
- [16] S. T. Zachariah and S. Chakravarty, “A Scalable and Efficient Methodology to Extract Two Node Bridges from Large Industrial Circuits,” in *Proc. of International Test Conference*, pp. 750–759, Oct. 2000.
- [17] “Perl Interface to LEF/DEF,” <http://www.openeda.org>.
- [18] Texas Instruments, Dallas, TX, *TTL Databook*.
- [19] Y. Sato *et al.*, “A Persistent Diagnostic Technique For Unstable Defects,” in *Proc. International Test Conference*, pp. 242–249, Oct. 2002.
- [20] A. K. Stamper, T. L. McDevitt and S. L. Spuce, “Sub-0.25-micron Interconnection Scaling: Damascene Copper Versus Subtractive Aluminum,” in *Proc. of Advanced Semiconductor Manufacturing Conference and Workshop*, pp. 337–346, Sept. 1998.
- [21] R. D. Blanton *et al.*, “Fault Tuples in Diagnosis of Deep-submicron Circuits,” in *Proc. of International Test Conference*, pp. 233–241, Oct. 2002.
- [22] B. Keller, *Hierarchical Pattern Faults for Describing Logic Circuit Failure Mechanisms*, 1994, U. S. Patent 5,546,408.
- [23] Kumar N. Dwarakanath, “Fault Tuples: A Paradigm for Universal Test Analysis,” Tech. Rep. CMU-CAD 01-21, Carnegie Mellon University, Nov. 2001.