

Z-DFD: DESIGN-FOR-DIAGNOSABILITY BASED ON THE CONCEPT OF Z-DETECTION

Irith Pomeranz¹, School of ECE, Purdue University, W. Lafayette, IN 47907
Srikanth Venkataraman, Intel Corporation, Hillsboro, OR 97124
Sudhakar M. Reddy², ECE Dept., University of Iowa, Iowa City, IA 52242

Abstract

We address the problem of design-for-diagnosability, i.e., improving the accuracy of fault diagnosis or reducing its complexity through the insertion of observation points. To perform design-for-diagnosability efficiently, we use a procedure developed earlier for computing the number of fault pairs, N_p , that are not guaranteed to be distinguished by a given test set. By using the concept of z -detection, N_p can be computed efficiently without enumerating fault pairs and without performing non-fault dropping fault simulation. We study the possibility of increasing the diagnosability of a circuit by inserting observation points so as to reduce N_p . Our results include the following. (1) We find experimentally the number of observation points that need to be inserted in order to achieve a close-to-minimum value for N_p . (2) We describe an efficient procedure for inserting a given number of observation points so as to reduce N_p . We present experimental results for benchmark circuits to demonstrate the accuracy of using N_p to guide a design-for-diagnosability process.

1. Introduction

Design-for-diagnosability is the process of improving the accuracy of fault diagnosis or reducing its complexity through circuit modifications or the insertion of logic. This is important when diagnosis of the original circuit may not result in the desired level of diagnostic resolution or has a high complexity. In this work, we propose an efficient design-for-diagnosability procedure for combinational (or full-scan) circuits through the insertion of observation points. An important feature of the proposed procedure is that it does not require enumeration of fault pairs, and it can thus be applied efficiently to large circuits.

Fault diagnosis [1] is a process that requires direct or indirect consideration of pairs of faults. For example, the goal of diagnostic test generation is to ensure that for every pair of faults f_i and f_j there is at least one test t

such that the circuit-under-test produces a different output response depending on whether f_i or f_j is present in the circuit. Efficient procedures for diagnostic fault simulation and test generation rely on implicit consideration of fault pairs [1]-[12]. In [12], the concept of z -detection was introduced and used for identifying fault pairs that are guaranteed to be distinguished by a given test set. The same concept was also used for developing an efficient procedure for computing the number of fault pairs that are not guaranteed to be distinguished by a given test set. This number was denoted by N_p . The computation of N_p in [12] required fault simulation with fault dropping of a given test set, traversal of the circuit, set operations over pairs of fault sets, and arithmetic operations over set sizes. Thus, computation of N_p does not require explicit consideration of fault pairs, and has an overall complexity similar to that of fault simulation with fault dropping.

In this work we use N_p and the concept of z -detection to develop an efficient design-for-diagnosability procedure based on the insertion of observation points. Similar to the procedure from [12], the proposed procedure does not require consideration of fault pairs. The procedure uses a given test set to compute N_p . In our experiments, the test set is a fault detection test set. It is also possible to use a diagnostic test set, or a test set that was enhanced to increase the number of z -detected faults. By using a fault detection test set we avoid the need to consider fault pairs during diagnostic test generation before the diagnosability of the circuit is improved. We also avoid the need for more complex test generation that targets z -detection. At the same time, since the percentage of z -detected faults is very high even when fault detection test sets are used [12], we expect that the specific fault detection test set used will not have a significant effect on the insertion of observation points. Our results include the following.

We compute a value for N_p , which is close to the minimum value possible for the circuit, by inserting a large number of observation points. We also compute a close-to-minimum number of observation points required to achieve this value of N_p . We find that to achieve a close-to-minimum value of N_p , it is necessary to insert a large number of observation points. This is a plausible result since different fault pairs, in different parts of the

1. Research supported in part by a Grant from Intel Corp.

2. Research supported in part by NSF Grant No. CCR-0097905 and in part by SRC Grant No. 2001-TJ-949.

circuit, may need different observation points in order to be distinguished. Thus, it should not be expected to find a small number of observation points that will allow us to distinguish all or most of the fault pairs that are not distinguished without observation points.

We describe an efficient procedure for inserting a given number of observation points so as to reduce N_p , thus increasing the diagnosability of the circuit. In our experiments we consider numbers of observation points equal to at most 3%, 2%, 1% and 0.5% of the circuit lines. We verify that N_p is an appropriate guideline for design-for-diagnosability by performing diagnostic fault simulation with and without observation points, and comparing the results to those predicted based on N_p .

To our knowledge, this is the first time design-for-diagnosability is considered for facilitating fault diagnosis. We concentrate on the insertion of observation points since it fits naturally with the concept of z -detection, which is based on output information. Z -detection can also be enhanced by inserting control points, not considered here. However, observation points are less intrusive and are also used for other purposes, for example, to enhance the fault coverage achieved by functional tests [13]-[15].

The paper is organized as follows. In Section 2 we review the concept of z -detection and the computation of N_p , which is the number of fault pairs that are not guaranteed to be distinguished by a given test set. In Section 3 we evaluate the minimum value of N_p that can be achieved by inserting observation points. In Section 4 we describe a procedure for inserting a given number of observation points so as to reduce N_p . Experimental results are presented in Sections 3 and 4.

2. Z-detection

To define the concept of z -detection we need the following notation [12].

We consider a circuit with n outputs denoted by z_0, z_1, \dots, z_{n-1} . For a line g in the circuit, we denote by $Z(g)$ the set that contains every output z_i such that there is a (structural) directed path in the circuit from g to z_i . We refer to $Z(g)$ as the z -set of g .

For a single stuck-at fault f , which is associated with line g , we define the z -set $Z(f) = Z(g)$. The fault f can be the fault g stuck-at 0 or the fault g stuck-at 1. We denote by $F(Z)$ the set of faults that contains every fault f such that $Z(f) = Z$.

We say that a fault f is z -detected by a test t if t propagates the effects of f to every one of the outputs in $Z(f)$.

We demonstrate these definitions using the circuit shown in Figure 1. Line numbers are shown in square brackets in Figure 1.

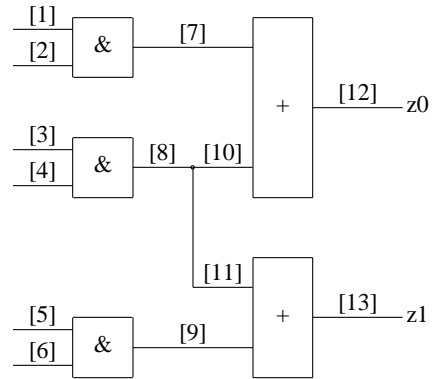


Figure 1: Example circuit [12]

Considering the collapsed set of single stuck-at faults in the circuit of Figure 1, each fault is associated with one of three z -sets, $Z_0 = \{z_0, z_1\}$, $Z_1 = \{z_0\}$, or $Z_2 = \{z_1\}$. Denoting the fault g stuck-at a by g/a we have $F(Z_0) = \{3/1, 4/1, 8/0, 8/1\}$ (these are the faults whose sites have paths to both outputs), $F(Z_1) = \{1/1, 2/1, 7/0, 10/0, 12/0, 12/1\}$ (these are the faults whose sites have paths only to z_0) and $F(Z_2) = \{5/1, 6/1, 9/0, 11/0, 13/0, 13/1\}$ (these are the faults whose sites have paths only to z_1).

The faults in $F(Z_1)$ and $F(Z_2)$ can only be propagated to the single output in their z -set. Consequently, if a test t detects a fault $f \in F(Z_1)$ (or $f \in F(Z_2)$), we can immediately say that t also z -detects f . For a fault $f \in F(Z_0)$, a test t that detects f may or may not z -detect the fault. For example, we show in Figure 2 a test for the fault line 3 stuck-at 0. This test does not z -detect the fault, since $Z(3) = Z_0 = \{z_0, z_1\}$, but the test only propagates the effects of the fault to z_1 . The test shown in Figure 3 z -detects the fault line 3 stuck-at 0, since it propagates the effects of the fault to both z_0 and z_1 .

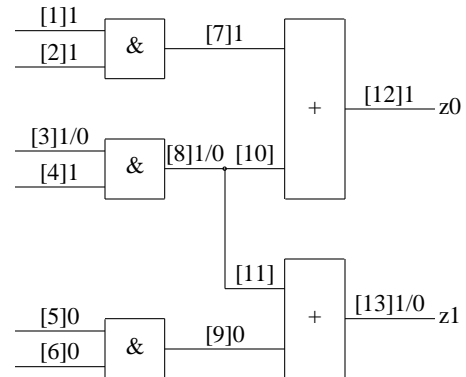


Figure 2: Example test 1

For a circuit with n outputs, the number of z -sets can be as high as $2^n - 1$. However, we are only interested in a z -set Z if there exists a fault f such that $Z(f) = Z$. Consequently, the number of z -sets cannot be larger than

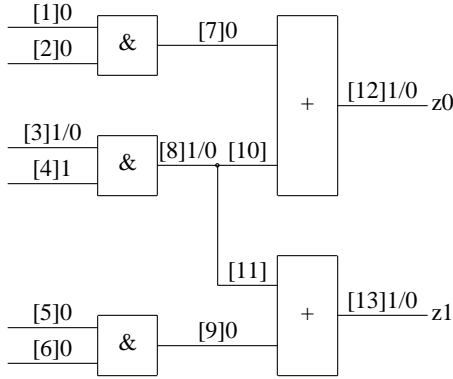


Figure 3: Example test 2

the number of single stuck-at faults.

To find the faults that are z -detected by a given test set T , we simulate T , dropping a fault f from further simulation only if it is z -detected by a test in T , i.e., only if it is detected on all the outputs in its z -set. As long as the fault is not z -detected, we continue to simulate it even if it is already detected. Experimental results presented in [12] show that fault simulation for z -detection is similar in complexity to n -detection fault simulation for $n \leq 3$ when T is a compact fault detection test set.

To demonstrate the importance of z -detections in fault diagnosis, let us consider the circuit of Figure 1 under a test set T that includes the test of Figure 3. This test z -detects the fault line 3 stuck-at 0. Consider an arbitrary fault $f \in F(Z_1) \cup F(Z_2)$. Even if f is detected or z -detected by the test of Figure 3, it can only be detected on a single output (z_0 if $f \in F(Z_1)$ or z_1 if $f \in F(Z_2)$). In either case, the test of Figure 3 is *guaranteed* to distinguish the fault 3 stuck-at 0 from f .

More generally, the knowledge that a fault is z -detected by a test set T *guarantees* that the fault will be distinguished by T from certain other faults. The following procedure was presented in [12] for counting (or enumerating) the fault pairs which are *not* guaranteed to be distinguished by a given test set T . In this procedure, N_p is the number of fault pairs that are *not* guaranteed to be distinguished by T . The set $F(Z_i)$ of faults with z -set equal to Z_i is partitioned in this procedure into two subsets. The subset $A(Z_i) \subseteq F(Z_i)$ contains the faults that are z -detected by T . The subset $B(Z_i) \subseteq F(Z_i)$ contains the faults that are not z -detected by T (however, they are detected by T). As demonstrated by the example above, a fault in $A(Z_i)$ will be distinguished from a fault in $F(Z_j)$ whenever $Z_i \not\subseteq Z_j$. In addition, if Z_i and Z_j are disjoint then all the faults in $F(Z_i)$ are guaranteed to be distinguished from all the faults in $F(Z_j)$. The procedure considers all the cases where a fault pair is not guaranteed to be distinguished.

Procedure 1: Computing the number of fault pairs N_p that are not guaranteed to be distinguished by T

- (1) Set $N_p = 0$.
- (2) For every pair of z -sets Z_i and Z_j (including the case where $Z_i = Z_j$):
 - (a) If $Z_i = Z_j$, set $N_p = N_p + |A(Z_i)| \cdot [|A(Z_i)| - 1] / 2 + |B(Z_i)| \cdot [|B(Z_i)| - 1] / 2$.
 - (b) If $Z_i \not\subseteq Z_j$ but $Z_j \subset Z_i$, set $N_p = N_p + |B(Z_i)| \cdot |F(Z_j)|$.
 - (c) If $Z_i \subset Z_j$ but $Z_j \not\subseteq Z_i$, set $N_p = N_p + |F(Z_i)| \cdot |B(Z_j)|$.
 - (d) $Z_i \not\subseteq Z_j$ and $Z_j \not\subseteq Z_i$ but $Z_i \cap Z_j \neq \emptyset$, set $N_p = N_p + |B(Z_i)| \cdot |B(Z_j)|$.

It should be noted that before Procedure 1 is applied, it is necessary to perform one pass over the circuit to identify z -sets for all the faults, and fault simulation with fault dropping to identify z -detected faults. The faults are then partitioned into sets $F(Z_i)$, and each set is partitioned into $A(Z_i)$ and $B(Z_i)$. Beyond these operations, Procedure 1 consists of set operations over pairs of fault sets and arithmetic operations over set sizes. Thus, Procedure 1 never considers fault pairs explicitly.

Application of Procedure 1 to benchmark circuits and to industrial circuits in [12] demonstrated that small percentages of fault pairs remain, which are not guaranteed to be distinguished by a fault detection test set. In the following sections we reduce this percentage further by inserting observation points.

3. Effect of observation points

In this section our goal is to investigate the potential effect that observation points can have on diagnosis. We measure the effect of an observation point by considering the number of fault pairs N_p computed by Procedure 1 based on a test set T . An effective observation point should reduce N_p , leaving fewer fault pairs that are not guaranteed to be distinguished by T .

We impose several rules regarding the inserted observation points. The rules serve two purposes. (1) They exclude observation points that do not help in reducing N_p . (2) They exclude observation points that may increase N_p . Although an observation point can never reduce the diagnosability of a circuit, it may increase N_p because of the way it is computed by Procedure 1. We prefer to omit such observation points.

After applying the rules, we insert all the observation points that satisfy the rules. Considering the circuit with the inserted observation points, we apply Procedure 1 to see the effect on the number of fault pairs, which are not guaranteed to be distinguished by T . In the next section we describe a procedure for selecting a given number

of observation points to achieve a maximal reduction in N_p .

An observation point can affect N_p for the following reason. Let f_i be the fault g_i stuck-at α_i and let f_j be the fault g_j stuck-at α_j . Suppose that f_i and f_j have the same z -sets. Suppose in addition that g_i drives a line s_i but not a line s_j ; and that g_j drives s_j but not s_i . This situation is demonstrated in Figure 4, where the boxes stand for logic blocks that may consist of one or more gates. The box marked f_i contains g_i , and the box marked f_j contains g_j . It can be seen that only g_i drives s_i and only g_j drives s_j , but both faults have the same z -set, $\{z_0, z_1\}$.

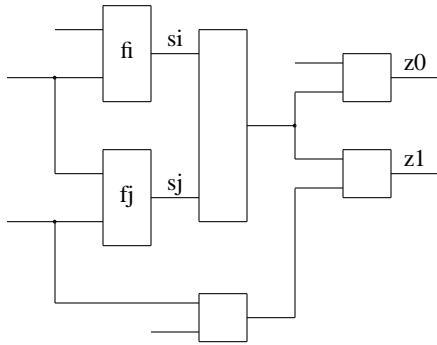


Figure 4: Effect of observation point

Without observation points on s_i and s_j , Procedure 1 will note that f_i and f_j are distinguished by T only if one of them is z -detected by T and the other one is not. If s_i and s_j are added to the set of circuit outputs, the z -set of f_i becomes $\{z_0, z_1, s_i\}$ while the z -set of f_j becomes $\{z_0, z_1, s_j\}$. In this case, Procedure 1 will note that the faults are distinguished even if both of them are z -detected. We summarize all the possible cases in Table 1. With the observation points on s_i and s_j there is one less case where the fault pair f_i, f_j will be counted by Procedure 1.

Table 1: Counting for f_i, f_j of Figure 4

z-detected		counted	
f_i	f_j	without s_i, s_j	with s_i, s_j
yes	yes	yes	no
yes	no	no	no
no	yes	no	no
no	no	yes	yes

It is important to note the following point. If an observation point is added on s_i but not on s_j , then the pair f_i, f_j will be counted by Procedure 1 in the case where f_i is not z -detected, even if f_j is z -detected. Thus, it is important to place observation points on both s_i and s_j to ensure that N_p will not increase due to this effect. We also note that the efficiency of Procedure 1 comes from the fact that it does not need to consider fault pairs

explicitly. To keep this advantage, we do not perform a detailed analysis of fault pairs in order to place observation points. Instead, to estimate the effect of observation points in this section, we place observation points on all the lines in the circuit except for certain lines as discussed below. In this way we attempt to capture all the observation points such as s_i and s_j in Figure 4.

The procedure we use in this section for placing observation points is as follows. Initially, the set of candidate observation points S includes every line in the circuit, which is the output of a multi-input gate but is not a primary output. The first rule given below excludes candidate observation points from S based on a purely structural analysis of the circuit. The last two rules use information about z -detected faults to exclude additional candidate observation points.

Rule 1: If $s \in S$ is a line that drives a primary output through single-input gates, an observation point on s will have a negligible effect beyond that of the primary output. We remove s from the set of candidates S .

Rule 2: Consider a fault f_i which is not z -detected by T . Let f_i drive a line s . If we add an observation point on s , f_i will still not be z -detected by T . As a result, f_i will continue to contribute the same count to N_p . If s is only driven by faults that are not z -detected by T , the addition of s will not affect N_p . Consequently, if s is only driven by faults that are not z -detected by T we remove s from S .

Rule 3: Consider a fault f_i which is z -detected by T . Let f_i drive a line s . If we add an observation point on s , the z -set of f_i will include s . Therefore, for f_i to continue to be z -detected, it must be detected on s as well as all the other outputs it drives. If f_i is not detected on s , we will lose the z -detection of f_i when we place an observation point on s . This can cause N_p to increase for the following reason. When f_i is z -detected, a pair f_i, f_j will not be counted in N_p as long as $Z_i \not\subseteq Z_j$. If f_i is not z -detected, there are cases where f_i, f_j will be counted in N_p even if $Z_i \not\subseteq Z_j$. Thus, we would like to avoid situations where we lose the z -detection of a fault by adding an observation point. We ensure this condition as follows.

If a fault f_i is z -detected by a test $t \in T$ and there is a path from f_i to s , we check whether f_i will still be z -detected by t after we add an observation point on s . In other words, we check whether t propagates the effect of f_i to s . If f_i will not be z -detected by t with an observation point on s , we remove s from the set of candidate observation points S .

In this check, we only consider a single test $t \in T$ that z -detects f . This allows us to keep to a minimum the simulation effort required for identifying z -detected faults.

Next, we report the results of the following experiment. We use a compact fault detection test set T , i.e., a compact test set that detects every detectable single stuck-at fault. We compute the set of candidate observation points S as described above. We insert all the observation points in S into the circuit, and apply Procedure 1. The results are reported in Tables 2 and 3 as follows.

Table 2: Candidate observation points

circuit	obs points			
	init	Rule1	Rule2	Rule3
s1423	416	414	276	224
s5378	952	932	640	387
s9234	2014	2000	1607	686
s13207	2573	2549	1895	992
s15850	3387	3358	2555	1647
s35932	10476	10476	7020	2930
s38417	8708	8650	6630	3435
b14	2928	2928	1676	1136
b20	6768	6768	3203	2296

Table 3: Remaining fault pairs

circuit	flt pairs	original	
		NP	%NP
s1423	1125750	364347	32.36
s5378	10408203	779089	7.49
s9234	20959575	2662558	12.70
s13207	46691616	3195550	6.84
s15850	64246780	6095945	9.49
s35932	616338495	2519720	0.41
s38417	480949605	9098292	1.89
b14	33722578	11329572	33.60
b20	194449060	49575658	25.50

circuit	obs points		
	NP	%NP	%reduc
s1423	334201	29.69	8.27
s5378	667552	6.41	14.32
s9234	1848917	8.82	30.56
s13207	3083202	6.60	3.52
s15850	4975287	7.74	18.38
s35932	2262474	0.37	10.21
s38417	7933511	1.65	12.80
b14	9508659	28.20	16.07
b20	43469985	22.36	12.32

In Table 2, following the circuit name we show the number of lines in S before applying Rules 1, 2 and 3, after applying Rule 1, after applying Rules 1 and 2, and after applying Rules 1, 2 and 3.

In Table 3, following the circuit name we show the total number of fault pairs. Considering the original circuit, we show under column *original* the number of fault

pairs N_p that are not guaranteed to be distinguished by the fault detection test set T , and the percentage of such fault pairs. Considering the circuit with observation points, we show under column *obs points* the value and percentage of N_p , as well as the percentage reduction in N_p due to the insertion of observation points. The percentage reduction is computed as the difference between the value of N_p with and without observation points, divided by the value of N_p without observation points.

From Table 2 it can be seen that there are many candidate observation points in benchmark circuits. From Table 3 it can be seen that these observation points allow non-trivial reductions in the numbers of fault pairs that are not guaranteed to be distinguished by a fault detection test set. Thus, the observation points can have a significant effect on diagnosis.

For comparison, we also consider the following sets of observation points. (1) The set of observation points that includes all the outputs of multi-input gates that are not primary outputs, i.e., before applying any rule. We denote this set by $S(-)$. (2) The set of observation points after applying Rule 1 but before applying Rules 2 and 3. We denote this set by $S(R1)$. (3) The set of observation points after applying Rules 1 and 2 but before applying Rule 3. We denote this set by $S(R1,R2)$. We insert the observation points in $S(-)$, in $S(R1)$ and in $S(R1,R2)$ into the circuit (each set separately) and find the reduction in the value of N_p relative to the case where no observation points are inserted. The results are reported in Table 4. For comparison we also show the results for S obtained after applying Rules 1, 2 and 3 (reported in Table 3). A negative number indicates that N_p increases as a result of not applying the corresponding rule(s).

Table 4: Comparison of sets of observation points

circuit	$S(-)$	$S(R1)$	$S(R1,R2)$	S
s1423	5.43	5.43	5.43	8.27
s5378	-27.72	-27.72	-27.72	14.32
s9234	28.49	28.49	28.50	30.56
s15850	6.19	6.19	6.19	18.38
b14	14.16	14.16	14.16	16.07

From Table 4 and from the motivation for Rules 1 and 2, these rules do not change N_p relative to the case where observation points are placed on all the outputs of multi-input gates that are not primary outputs. Rule 3 is important for excluding observation points that can increase the value of N_p . The effect of Rule 3 can be seen to be significant.

4. Inserting a given number of observation points

In this section we describe a procedure for inserting a given number of observation points so as to reduce N_p . The procedure starts from the solution that includes all the observation points in the set S considered in the previous section. It then removes observation points from S while ensuring that the number of fault pairs computed by Procedure 1 does not go above a target value for N_p . By increasing the target for N_p gradually, the number of observation points is reduced. The procedure stops when the target number of observation points is reached.

To describe the procedure we denote by $N_{p,none}$ the value of N_p computed by Procedure 1 without any observation points. We denote by S_{all} the set of observation points obtained in the previous section (after applying Rules 1, 2 and 3), and we denote by $N_{p,all}$ the value of N_p computed by Procedure 1 with all the observation points in S_{all} inserted into the circuit. We consider target values of N_p of the form

$$\hat{N}_{p,p} = N_{p,all} + (N_{p,none} - N_{p,all}) \cdot p / 100$$

for increasing percentages p . For each value of p we compute a set of observation points S_p by removing observation points from the previous set (the previous set is S_{p-1} if $p > 0$ or S_{all} if $p = 0$). We also compute intermediate sets between S_{p-1} and S_p as described later. Once S_p is found, Procedure 1 is used to compute the actual value of N_p obtained for S_p . This value of N_p is denoted by $N_{p,p}$, and it satisfies $N_{p,p} \leq \hat{N}_{p,p}$. With $p = 0$ we use $\hat{N}_{p,0} = N_{p,all}$, and we obtain $S_0 \subseteq S_{all}$. We may obtain $S_0 \subset S_{all}$ (with proper containment) if some of the observation points in S_{all} are unnecessary. We may also obtain $N_{p,0} \leq N_{p,all}$. For $p > 0$, we compute S_p such that $S_p \subseteq S_{p-1}$. As we increase p , the values of $\hat{N}_{p,p}$ and $N_{p,p}$ increase and the size of the set S_p decreases.

For a given value of p and a corresponding value of $\hat{N}_{p,p}$, we consider every observation point as a candidate for removal. We accept the removal of an observation point s if the value of N_p computed by Procedure 1 does not exceed the target value $\hat{N}_{p,p}$. The procedure for removing observation points from S_{all} is given next as Procedure 2. Additional details are given following the procedure.

Procedure 2: Finding sets of observation points

- (1) Apply Procedure 1 to compute $N_{p,none}$ for the case where no observation points are inserted.
- (2) Apply Procedure 1 to compute $N_{p,all}$ for the case where all the observation points in S_{all} are inserted into the circuit.
- (3) Set $p = 0$.
- (4) If $p = 0$ set $S_p = S_{all}$. Else set $S_p = S_{p-1}$. Set $\hat{N}_{p,p} = N_{p,all} + (N_{p,none} - N_{p,all}) \cdot p / 100$. Set $m = 0$.

- (5) For every $s \in S_p$, considering the observation points in a random order:
 - (a) Remove s from S_p .
 - (b) Apply Procedure 1 to compute $N_{p,p}$ for the case where the observation points in S_p are inserted into the circuit.
 - (c) If $N_{p,p} > \hat{N}_{p,p}$, add s back into S_p . Else set $S_{p,m} = S_p$, $N_{p,p,m} = N_{p,p}$ and $m = m + 1$.
- (6) Set $p = p + \Delta$, where Δ is a preselected constant. If $p < 100$ go to Step 4.

We consider the observation points for removal in a random order to avoid clustering of observation points. We found experimentally that avoiding clustering of observation points is important in ensuring that the actual number of indistinguished fault pairs by a given test set tracks the value of N_p . In this experiment, we compared the use of a random order to the use of a heuristic for determining the order in which observation points will be considered for removal. We report the results of this experiment later to support the use of a random order over the use of the heuristic.

It is important to note that fault simulation to identify z -detected faults is done only once, before Procedure 1 is applied for the first time. Fault simulation is needed again to apply Rule 3 of Section 3. After that, Procedure 1 (and Procedure 2) require only set operations over pairs of fault sets $F(Z_i)$, $A(Z_i)$ and $B(Z_i)$, and arithmetic operations over set sizes.

Between S_p and S_{p+1} , Procedure 2 computes a sequence of sets $S_{p,m}$, $m = 0, 1, \dots$, such that $N_{p,p,m} \leq \hat{N}_{p,p}$. Any one of these sets can be used as the set of observation points for the circuit.

Our goal is to obtain sets $S_{p,m}$ that include at most 3%, 2%, 1% and 0.5% of the circuit lines. To reach this goal with the least amount of computational effort, p should be varied in large steps, implying that Δ should be large. This is because every value of p requires that we consider all the lines in S_p , and larger steps will get us to the target size with fewer passes over the set of observation points. However, large values of Δ can also result in sets S_p that are larger than necessary, for the following reason. Suppose that S_p contains two lines s_1 and s_2 that can be removed. Suppose that s can be removed as well; however, once s is removed, s_1 and s_2 cannot be removed. In this case, we would like to remove s_1 and s_2 , and keep s . Typically in this situation, s causes a larger reduction in $N_{p,p}$ than either s_1 or s_2 . Therefore, a larger value of $\hat{N}_{p,p}$ (corresponding to a larger value of p and a larger value of Δ) may allow s to be removed; while increasing $\hat{N}_{p,p}$ in smaller steps may ensure that s will not be removed, allowing s_1 and s_2 to be removed.

As a compromise, we apply Procedure 2 with $\Delta = 5$ and $p = 0, 5, 10, 15, \dots, 95$. We report the results only for values of p and m that result in a set $S_{p,m}$ that includes at most 3%, 2%, 1% or 0.5% of the circuit lines. The results are reported in Tables 5-11. In the first row for every circuit we repeat the results obtained for S_{all} . In the second row we show the results for S_0 , i.e., for $p = 0$ and the last value of m obtained for $p = 0$. In the next rows we show the results for additional values of p and m , for which $S_{p,m}$ contains at most 3%, 2%, 1% or 0.5% of the circuit lines. In the last row we repeat the results when no observation points are placed. In each row we show the value of p , the number of observation points, the percentage of observation points out of all the circuit lines, the value of N_P computed by Procedure 1, and the percentage reduction in N_P compared to $N_{P,none}$. The values under columns $dsim$ and $\%dsim$ are explained later. The following points can be seen from Tables 5-11.

Table 5: Results of Procedure 2 (s1423)

p	obs	%obs	NP	%reduc	dsim	%dsim
all	224	15.74	334201	8.27	63	74.90
0%	211	14.83	334200	8.27	63	74.90
25%	42	2.95	350788	3.72	175	30.28
30%	28	1.97	351919	3.41	185	26.29
35%	14	0.98	355621	2.39	193	23.11
35%	7	0.49	357450	1.89	243	3.19
none	0	0.00	364347	0.00	251	0.00

Table 6: Results of Procedure 2 (s5378)

p	obs	%obs	NP	%reduc	dsim	%dsim
all	387	7.31	667552	14.32	514	13.76
0%	347	6.55	667551	14.32	517	13.26
20%	159	3.00	709679	8.91	546	8.39
25%	106	2.00	726816	6.71	556	6.71
35%	53	1.00	746093	4.24	569	4.53
40%	26	0.49	758318	2.67	585	1.85
none	0	0.00	779089	0.00	596	0.00

Table 7: Results of Procedure 2 (s9234)

p	obs	%obs	NP	%reduc	dsim	%dsim
all	686	7.43	1848917	30.56	1103	43.46
0%	536	5.80	1848917	30.56	1149	41.11
15%	277	3.00	1998564	24.94	1380	29.27
20%	185	2.00	2081323	21.83	1501	23.07
25%	92	1.00	2288614	14.04	1667	14.56
30%	46	0.50	2343054	12.00	1737	10.97
none	0	0.00	2662558	0.00	1951	0.00

Table 8: Results of Procedure 2 (s13207)

p	obs	%obs	NP	%reduc	dsim	%dsim
all	992	7.53	3083202	3.52	1647	35.71
0%	903	6.85	3083200	3.52	1671	34.78
20%	395	3.00	3117399	2.45	1977	22.83
25%	263	2.00	3133864	1.93	2108	17.72
30%	132	1.00	3156774	1.21	2379	7.14
35%	66	0.50	3164683	0.97	2465	3.79
none	0	0.00	3195550	0.00	2562	0.00

Table 9: Results of Procedure 2 (s15850)

p	obs	%obs	NP	%reduc	dsim	%dsim
all	1647	10.39	4975287	18.38	2306	35.21
0%	1492	9.42	4975287	18.38	2337	34.34
25%	475	3.00	5445219	10.67	2861	19.61
25%	317	2.00	5606225	8.03	2994	15.88
35%	158	1.00	5780714	5.17	3126	12.17
40%	79	0.50	5882627	3.50	3482	2.16
none	0	0.00	6095945	0.00	3559	0.00

Table 10: Results of Procedure 2 (b14)

p	obs	%obs	NP	%reduc	dsim	%dsim
all	1136	13.22	9508659	16.07	338	27.47
0%	1136	13.22	9508659	16.07	338	27.47
35%	258	3.00	10830724	4.40	438	6.01
35%	172	2.00	10968113	3.19	444	4.72
45%	86	1.00	11142488	1.65	459	1.50
50%	43	0.50	11225631	0.92	460	1.29
none	0	0.00	11329572	0.00	466	0.00

Table 11: Results of Procedure 2 (b20)

p	obs	%obs	NP	%reduc	dsim	%dsim
all	2296	11.93	43469985	12.32	1053	22.74
0%	2285	11.87	43469985	12.32	1053	22.74
30%	576	2.99	47682812	3.82	1300	4.62
35%	383	1.99	48220825	2.73	1312	3.74
40%	191	0.99	48826336	1.51	1331	2.35
45%	96	0.50	49068649	1.02	1335	2.05
none	0	0.00	49575658	0.00	1363	0.00

Table 12: Procedure 2 with a heuristic order (s1423)

p	obs	%obs	NP	%reduc	dsim	%dsim
all	224	15.74	334201	8.27	63	74.90
0%	210	14.76	334201	8.27	63	74.90
20%	42	2.95	346837	4.81	156	37.85
25%	28	1.97	350821	3.71	213	15.14
30%	14	0.98	352742	3.19	228	9.16
35%	7	0.49	356032	2.28	233	7.17
none	0	0.00	364347	0.00	251	0.00

Table 13: Procedure 2 with a heuristic order (s5378)

p	obs	%obs	NP	%reduc	dsim	%dsim
all	387	7.31	667552	14.32	514	13.76
0%	342	6.46	667551	14.32	517	13.26
15%	159	3.00	684031	12.20	564	5.37
15%	106	2.00	697198	10.51	590	1.01
20%	53	1.00	714055	8.35	596	0.00
25%	26	0.49	730214	6.27	596	0.00
none	0	0.00	779089	0.00	596	0.00

The value of $N_{P,0}$ is sometimes smaller than $N_{P,all}$; however, the difference between them is very small. During the computation of S_0 we obtain sets $S_{0,m}$, not reported in Tables 5-11, for which $N_{P,0,m} < N_{P,all}$. For example, for s 5378 the smallest value we obtain for N_P is $N_{P,0,7} = 667525$. However, the values $N_{P,0,m}$ obtained during the computation of S_0 are very close to $N_{P,all}$ as well. This indicates that $N_{P,all}$ is a good estimate of the minimum value that can be achieved for N_P .

The set S_0 is not significantly smaller than S_{all} . This indicates that almost all the observation points included in S_{all} are needed in order to achieve the maximum reduction in N_p . This is related to the fact that different fault pairs, in different parts of the circuit, may need different observation points. Thus, it should not be expected to find a small number of observation points that will affect all or most of the fault pairs.

If the number of observation points is restricted to 3% (2%, 1% or 0.5%) of the circuit lines, it is still possible to reduce N_p . However, not as dramatically as with higher numbers of observation points. This is again related to the nature of the problem, where different fault pairs require different observation points. More dramatic reductions with smaller numbers of observation points are obtained for circuits where the difference between $N_{p,all}$ and $N_{p,none}$ is high. This happens, for example, for $s9234$. We further demonstrate the reduction of N_p with the number of observation points in Figures 5-7. The reduction in N_p as a function of the percentage of lines with observation points is shown by the circles connected with a solid line. The x's connected with a dashed line will be explained later.

The advantage of using N_p to guide observation point insertion is that the computation of N_p does not require diagnostic fault simulation or analysis of fault pairs. To confirm the accuracy of performing design-for-diagnosability based on N_p , we perform diagnostic fault simulation of the test set T for some of the circuits considered in Tables 5-11. We find the numbers of fault pairs left indistinguished by T for the various numbers of observation points reported in Tables 5-11. The numbers of indistinguished fault pairs are reported under column $dsim$ of Tables 5-11. The percentage reduction in the number of indistinguished fault pairs due to the insertion of observation points (relative to the value obtained without any observation points) is shown under column $\%dsim$ of Tables 5-11. We also show in Figures 5-7 the reduction in the number of indistinguished fault pairs as a function of the percentage of lines with observation points. This is shown using x's connected by a dashed line.

It can be seen that the value of N_p tracks the actual number of indistinguished fault pairs. Except for $s5378$, the actual reduction in the number of indistinguished fault pairs is typically higher than the reduction in N_p . This justifies the use of N_p as an estimate (in most cases a conservative estimate) of the improvement in the diagnosability of a circuit.

Finally, to support the use of a random order for the removal of observation points in Procedure 2, we report the results obtained by using the following heuristic to determine the order by which observation points are con-

sidered in Procedure 2.

Let s be an observation point. We denote by $n_{z,in}(s)$ the number of faults inside the input cone of s , which are z -detected. We denote by $n_{z,out}(s)$ the number of faults outside the input cone of s , which are z -detected. The presence of s ensures that the z -detected faults inside its cone are distinguished from the z -detected faults outside its cone. Thus, these fault pairs do not contribute to N_p when the observation point on s is present. There are $n_{z,in}(s) \times n_{z,out}(s)$ such fault pairs. We use $n_{z,in}(s) \times n_{z,out}(s)$ as an estimate of the increase in N_p that will occur if the observation point on s is removed. In Procedure 2, we select observation points for removal by increasing value of $n_{z,in}(s) \times n_{z,out}(s)$.

We report the results obtained for $s1423$ and $s5378$ in Tables 12 and 13. The results show that for the same number of observation points, the heuristic achieves higher reductions in N_p than the random order used for Tables 5-11. However, the value of N_p does not track the number of indistinguished fault pairs that actually remain in the circuit well when the heuristic is used, especially in the case of $s5378$. Considering the observation points removed by Procedure 2 using the random order and using the heuristic, we found that the observation points tend to be clustered under the heuristic. This clustering can leave indistinguished fault pairs in certain areas of the circuit.

5. Concluding remarks

We considered the problem of design-for-diagnosability through the insertion of observation points. We used the concept of z -detection, which can be used for efficient computation of the number of fault pairs that are not guaranteed to be distinguished by a given test set. This number was denoted by N_p . Our goal of increasing the diagnosability of a circuit was translated through this concept into a reduction of N_p . Our results were as follows. (1) We showed that to achieve a close-to-minimum value for N_p it is necessary to insert a large number of observation points. This is related to the fact that different fault pairs require different observation points, and it is impossible to find a small number of observation points that will be beneficial for all the fault pairs. (2) We described a procedure for inserting a given number of observation points so as to reduce N_p . We reported results of this procedure for numbers of observation points equal to at most 3%, 2%, 1% or 0.5% of the circuit lines. We also demonstrated the accuracy of N_p in predicting the actual improvement in the circuit diagnosability by performing diagnostic fault simulation to find the actual number of indistinguished fault pairs for a given test set.

References

- [1] M. Abramovici, M. A. Breuer and A. D. Friedman, *Digital Systems Testing and Testable Design*, Computer Science Press, 1990.
- [2] J. A. Waicukauski and E. Lindbloom, "Failure Diagnosis of Structured VLSI", *IEEE Design and Test of Computers*, Aug. 1989, pp. 49-60.
- [3] P. Camurati, D. Medina, P. Prinetto and M. Sonza Reorda, "A Diagnostic Test Pattern Generation Algorithm", in *Proc. Intl. Test Conf.*, 1990, pp. 52-58.
- [4] T. Gruning, U. Mahlstedt and H. Koopmeiners, "DIATEST: A Fast Diagnostic Test Pattern Generator for Combinational Circuits", in *Proc. Intl. Conf. on Computer-Aided Design*, Nov. 1991, pp. 194-197.
- [5] K. Kubiak, S. Parkes, W. K. Fuchs and R. Saleh, "Exact Evaluation of Diagnostic Test Resolution", *29th Design Autom. Conf.*, June 1992, pp. 347-352.
- [6] E. M. Rudnick, W. K. Fuchs and J. H. Patel, "Diagnostic Fault Simulation of Sequential Circuits", *Intl. Test Conf.*, 1992, pp. 178-186.
- [7] F. Corno, P. Prinetto, M. Rebaudengo and M. Sonza Reorda, "GARDA: A Diagnostic ATPG for Large Synchronous Sequential Circuits", in *Proc. Europ. Design&Test Conf.*, March 1995, pp. 267-271.
- [8] S. Venkataraman, I. Hartanto, W. K. Fuchs, E. M. Rudnick, S. Chakravarty and J. H. Patel, "Rapid Diagnostic Fault Simulation of Stuck-at Faults in Sequential Circuits using Compact Lists", in *Proc. Design Autom. Conf.*, June 1995, pp. 133-138.
- [9] V. Boppana, I. Hartanto and W. K. Fuchs, "Full Fault Dictionary Storage Based on Labeled Tree Encoding", in *Proc. VLSI Test Symp.*, April 1996, pp. 174-179.
- [10] S. Venkataraman and W. K. Fuchs, "Distributed Diagnostic Simulation of Stuck-at Faults in Sequential Circuits", in *Proc. VLSI Design*, Jan. 1997, pp. 381-385.
- [11] I. Pomeranz and S. M. Reddy, "A Diagnostic Test Generation Procedure for Synchronous Sequential Circuits Based on Test Elimination", in *Proc. Intl. Test Conf.*, Oct. 1998, pp. 1074-1083.
- [12] I. Pomeranz, S. Venkataraman, S. M. Reddy and B. Seshadri, "Z-Sets and Z-Detections: Circuit Characteristics that Simplify Fault Diagnosis", in *Proc. Design Automation and Test in Europe*, Feb. 2004.
- [13] W. Needham and N. Gollakota, "DFT Strategy for Intel Microprocessors", in *Proc. Intl. Test Conf.*, 1995, pp. 157-166.
- [14] A. Carbine and D. Feltham, "Pentium(r) Pro Processor Design for Test and Debug", in *Proc. Intl. Test Conf.*, 1997, pp. 294-303.
- [15] D. Josephson, S. Poehlman and V. Govan, "Debug Methodology for the McKinley Processor", in *Proc. Intl. Test Conf.*, 2001, pp. 451-460.

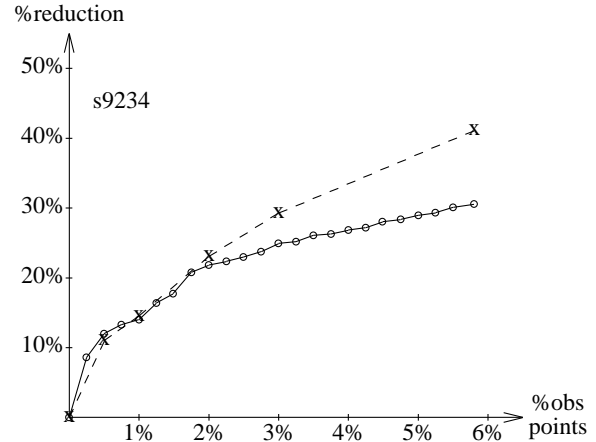


Figure 5: Results of Procedure 2 (s9234)

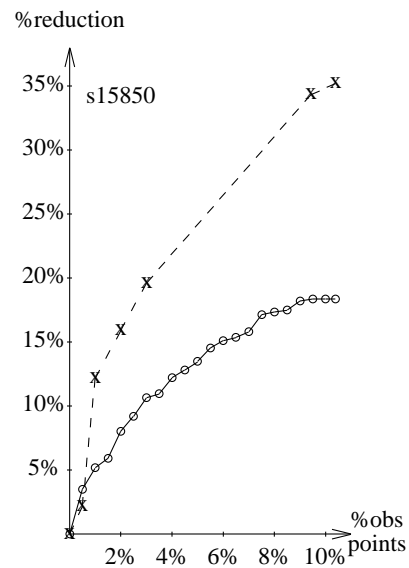


Figure 6: Results of Procedure 2 (s15850)

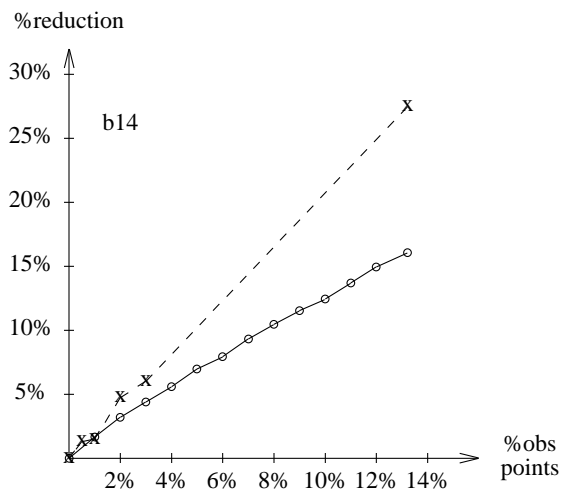


Figure 7: Results of Procedure 2 (b14)