

Independent Fault Sets

Ravi Chandra Paruchuri
Auburn University
Auburn , Alabama.

Abstract – This paper discusses the generation and use of independent fault sets in order to obtain the more global approach to automatic test generation. Independent faults were shown to be effective in computing small test sets for stuck at faults. In this paper, an efficient procedure for computing maximal set of independent stuck-at faults is proposed. Techniques are described for obtaining a series of sets which collectively give a global characterization of the testing requirements imposed by a given circuit. Procedures for using this information more effectively to guide the actual test generation process are developed. Examples involving benchmark circuits are included.

1. Introduction

While minimal test sets are highly desirable – particularly for VLSI designs –the primary concern of automatic test vector generation is not this, since the problem of generating a test set is computationally difficult. Most automatic test generation problem adopt a more local view of the overall test generation problem , by identifying a single target fault and generating a test vector for that fault.

A set of faults is said to be independent if no two faults in the set can be detected by the same test vector . The size of such a set will be a lower bound on the size of the required test set. Further, the identification of independent fault set requires an analysis of all the faults of interest. Since maximum independent fault set is as difficult as the test generation problem itself , we must content ourself with maximal independent fault set, let us say, F1. F1 will be located at very localized portion of given circuit. If the fault set F1 is temporarily removed from consideration and a second independent fault set F2 is derived from amongst the remaining faults, F2 will be typically found in an entirely different location[1].

The importance of independent faults is two fold. (1) They provide a lower bound on the size of minimum test set, thus making it possible to estimate the success of test pattern generators in generating small test sets. (2) Independent faults provide a method for ordering of target faults for test generation. Ordering has been

shown to be more important for obtaining small test sets and reducing test generation time [2].

In this paper I shall outline the process by which the independent fault set is identified. A preliminary discussion of independent fault set is done. Description of how and why another set of independent faults , F2, can be identified by temporarily removing the faults in F1, and the relationship between the faults in these two independent sets.

2. Independent Fault Sets

Fault independence is an orthogonal notion in that if a fault a is independent of fault b , if every test that detects a does not detect b and vice versa. For example , the stuck-at-1 faults on lead 14 and 15 are independent in Figure 1.

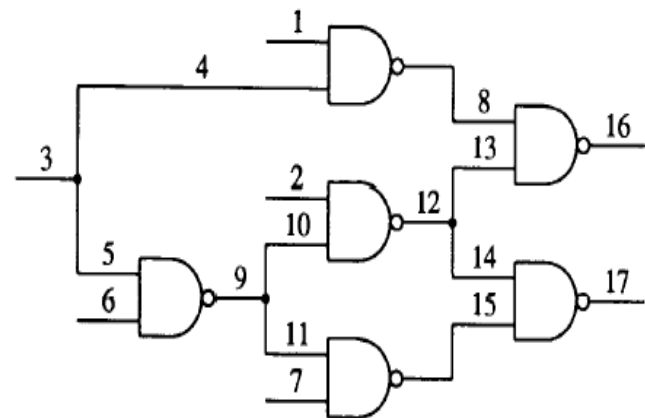


Figure 1: Circuit c 17

3. Test Values and their propogation

Any test generated for a given logic circuit is characterized by the set of logic values and also by the set of sensitivity of individual leads which appear on the circuit. Sensitivity is defined as whether or not a fault on a lead will be detected by the test. Each lead for a test, may have 4 values -0⁺, 0⁻, 1⁺, or 1⁻ where 0⁺ indicates that the normal logic value is 0 and that a single stuck at fault on that lead would be detected; 1⁺ indicates a logic value of 1 where a stuck-at-0 fault would not be detected , etc.

For example, let us take a 2 input AND gate; A and B being input leads and C being output lead. C may or may not be sensitive, (+ or - ; + being sensitive and - being insensitive) and also A and B may be at 1 or 0. Thus there are 8 cases as shown in Figure 2. However, knowing the sensitivity of C we can determine that of A and B and if the logic values of A and B are fixed the logic value of C can be determined. If C is insensitive, A and B are also. If C is sensitive the logic values of A and B which can change the value of C are also sensitive. This is shown in figure 3 results. In Figure 3, let us assume that A is at 0⁻. From Figure 3, we can see that C will be 0⁺ or 0⁻. Therefore, C must have a value of logic 0, though its sensitivity is undetermined.

A	B	C
0	0	-
0	1	-
1	0	-
1	1	-
0	0	+
0	1	+
1	0	+
1	1	+

Figure 2. Sensitivity values of output of AND gate

A	B	C
0-	0-	0-
0-	1-	0-
1-	0-	0-
1-	1-	0-
0-	0-	0+
0+	1-	0+
1-	0+	0+
1+	1+	1+

Figure 3. Test Values on an AND gate

This observation suggests that we can denote the “test value status” of every lead by a 4-bit binary number where each bit is 1 or 0 depending whether or not the corresponding test value can be assumed by that lead[1]. Figure 4 shows a table of 4 such bits and each 4-bit number is indicated by a symbol as shown. 0 indicates that a lead must have a logic value 0 (either 0⁺ or 0⁻); “-” indicates that lead is insensitive; “~1⁻” indicates that lead cannot assume 1⁻ (i.e., it must be at 0⁺ or 0⁻ or 1⁻). It will be easy now to develop rules for propagation of these symbols through the circuit.

TEST Value Symbol	Assignment			
	0 ⁻	1 ⁺	0 ⁺	1 ⁻
K	0	0	0	0
1 ⁻	0	0	0	1
(0)	0	0	1	0
1	0	0	1	1
1+	0	1	0	0
1	0	1	0	1
+	0	1	1	0
~0-	0	1	1	1
0-	1	0	0	0
-	1	0	0	1
(0)	1	0	1	0
~1 ⁺	1	0	1	1
0	1	1	0	0
~0 ⁺	1	1	0	1
~1 ⁻	1	1	1	0
X	1	1	1	1

Figure 4. Test value assignments

When the indicated symbol for a lead is derived it must be intersected with any (4-bit symbol) already present on that lead.

Let us discuss this with an example of how the fault propagation takes place. Consider the independent fault set, IFS_A of C17 and in particular the stuck-at-1 fault on lead 15. sa1 faults on lead 15 7 21 and sa0 fault on lead 15 comprises of IFS_A. To begin with we know that lead

15 must be at 0+ and that the outputs will be sensitive. The propagation process stops until all symbols stop changing. Every lead is specified, even though it might be partially as shown in figure. This gives us partial test if not full test of the values which the various leads may or may not have.

This process is repeated for other remaining 3 faults in the independent set. Four partial tests are now obtained. The first observation is that the fault independence depends on the lead values in various tests. There will be conflicts between the values of the tests for example, the sa1 fault on lead 15 requires a 0+ while in all others a value of 1 is assumed on this lead. The second observation is that none of these faults have a precise test specified. For most faults a number of different tests exists. For each of these partial tests a number of other faults are identified. For example, we can detect the faults on leads 16, 19, 21, 23 for the test on stuck at 1 fault on lead 15. A total of 12 additional faults can be identified as being detectable by these four tests.

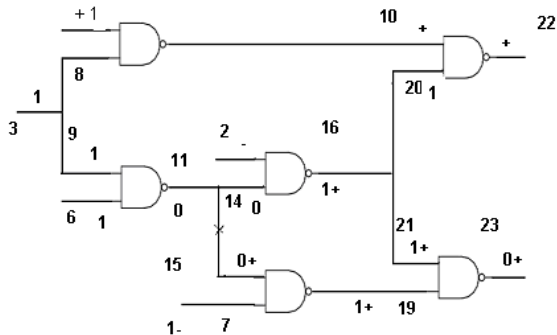


Figure 5. Propagated test values for a single fault

This test value propagation technique allows us to obtain a considerable amount of information about the test values which must be assumed for each of separate tests corresponding to the faults in an independent set. Moreover for each fault set a number of other faults are identified which can be detected. But this fault test generally involves leads in the localised portion. For this we need to consider a independent fault set preferably in the entirely different location.

The easiest way is to return to our original faultlist and simply remove from the list not only the faults in the previous independent fault set but also all the faults which their tests have been found to detect. Thus in the example involving C17, we would delete 16 faults from the original list .Now a second independent fault set, IFS_B is found. The sa0,sa1 on lead 1, sa1 on lead 8 and sa1 on lead 20 comprises of IFS_B. Test value

propagation is again performed on each of these faults to obtain a second partially specified test.

4. Fault Matching

Already a partial specification of test values are found which must be present in its test, we began looking for pairs of faults whose partial tests are in conflict. The test for lead 15 sa1 in IFS_A requires a 0 on lead 11 while the test for lead 8 sa1 in IFS_B requires a 1 on the same lead. This is done by intersecting their corresponding test assignments , we shall say that 2 faults are compatible and only if none of their intersected test assignments are null i.e., equal to K.

This information can be displayed by the compatibility matrix with 0 indicating that the corresponding pair is incompatible. Figure 6 shows that 7 pairs are incompatible between IFS_A and IFS_B. We need some sort of “measure of compatibility” which reflect the desirability of matching a particular pair. This can be done by considering the partial test which would result when a given pair is intersected. Figure 6 shows the compatibility matrix. Now we can invoke the ubiquitous linear assignment algorithm [4] to find the best matching between the faults in these 2 independent sets.

		IFS _B			
		1	8	1	20
		sa1	sa1	sa0	sa1
IFS _A	15				
	sa1	23	0	23	0
	7				
	sa1	21	21	21	0
	21				
sa1	0	0	0	27	
15					
sa0	22	21	23	0	

Figure 6. Compatibility Matrix

Thus this fault matching process involves four steps:

- (1) Elimination of already detected faults from the fault list
- (2) Generation of independent faultset and determination of partial tests
- (3) Construction of matrix showing the compatibility between the old and new partial test sets
- (4) Solving this new matrix to match the faults of IFSX onto the sets previously obtained.

Note that at each step in this process we not only obtain a new fault set, but also we increase the sizes of our compatible fault sets i.e., the sets of faults which may well be useful to look at some examples of the application of this process to larger circuits.

5. Computing Independent Fault Set

Independent faults are computed based on the line values that must be set for a fault under consideration to be detected. These values are called necessary assignments. Necessary assignments are illustrated in the following example.

Consider ISCAS-85 benchmark circuit c17 in Figure 1 and the fault line 11 stuck-at 1. Any test for the fault must set the following values.

- Line 11 must be set to 0.
- Consequently lines 9 and 10 are set to 0, line 12 is set to 1 and lines 13, 14 are set to 1.
- Lines 5 and 6 are set to 1 to set line 9 to 0
- Lines 3 and 4 are set to 1, consequently.
- To propagate the fault, line 7 must be set to 1
- Consequently, line 15 is set to 1, and line 17 is set to 0.

The complete set of necessary assignment is the values on each node. The following procedure allows the construction of independent fault sets.

- (1) Set F to be the set of all targets single stuck at faults. Compute the necessary assignments for every $f \in F$. Set $F^* = \{f \in F \mid f \text{ contains the independent fault set}\}$.
- (2) If F^* is empty, stop: F^* is a maximal independent fault set.
- (3) Select a fault f belongs to F^* . If the necessary assignments of f conflict with the necessary assignment of every fault $f' \in F^*$, add f to F^* .
- (4) Remove f from F and go to Step (2).

The size of the independent set strongly depends on the selection of faults in step (3). The criteria for selecting the fault in step(3) is the number of necessary assignments for the fault. The second criteria is the number of – values that are set to 0 or 1 when f is added. The third criteria used for the selection of fault is the larger number of 0 or 1 values in $V(f \cup F)$.

5. Conclusion

The over all objective in this investigation has been to obtain a more global approach to the general problem of test generation. The attention is limited to preprocessing steps which can be easily performed prior to formal initiation of the actual test generation and fault simulation.

Clearly, faults when conditions conflict will require separate tests while those where no conflict is apparent may be detectable by the same test. Accordingly, not only the methods have been described for finding independent faults, but also the techniques for looking across these sets to find compatible fault sets. An

algorithm for computing the independent fault set is also explained.

6. Acknowledgements

My sincere thanks to Mr. Raja Sandireddy for reviewing my paper and giving me some good suggestions.

7. References

- [1] Michael L. Bushnell and Vishwani D. Agrawal, Essentials of Electronic Testing For Digital, Memory And Mixed Signal Vlsi Circuits, Kluwer Academic Publishers, Boston, 2000.
- [2] S.B. Akers, Christie Joseph and Krishnamurthy. B, "On the Role of Independent Fault Sets in the Generation of Minimal Test Sets", in Proc. of the International Test Conf., Sept 1987, pp 1100-1107.
- [3] Irith Pomeranz and Sudhakar M. Reddy, "Generalisation of Independent Faults for Transition Faults", IEEE VLSI Test Symposium, 1992.
- [4] Even, S., "Graph Algorithms", Computer Science Press, 1979, pp 135-138.