Concurrent Test Generation

Vishwani D. Agrawal and Alok S. Doshi
Auburn University, Department of Electrical and Computer Engineering, Auburn, AL 36849, USA
vagrawal@eng.auburn.edu, *doshias@auburn.edu*

Abstract

We define a new type of test, called "concurrent test," for a combinational circuit. Given a set of target faults, a concurrent-test is an input vector that detects all (or most) faults in the set. When concurrent tests are generated for fault sets obtained from independence fault collapsing, minimal or near-minimal tests can be expected. This paper gives new simulation-based methods for independence fault collapsing and for deriving concurrent tests using single-fault ATPG.

1. Introduction

Consider two faults F1 and F2 in a combinational circuit and let T(F1) and T(F2) be the sets of all vectors that detect these faults (see Figure 1). Suppose an automatic test pattern generator (ATPG) targets F1 and finds the test vector v_1. Fault simulation will indicate that F2 should be targeted next. If we obtain the test v_3, *static compaction* will eliminate the vector v_1 and we will get just v_3 to cover both faults. However, if vector v_2 is obtained as a test for F2 then the compacted set will contain both vectors. Thus, static vector compaction cannot guarantee optimality because its outcome may be affected by a wrong vector selected for a single-fault target.

If v_1 has don't care bits, sometimes a *dynamic compaction* procedure may convert it into v_3, but this is not always guaranteed. Alternatively, dynamic compaction can try to iteratively replace the wrongly selected vectors [10]. This last method has been quite successful in achieving the optimum or near-optimum tests, but has a high time complexity.

Figure 1 shows a shortcoming of the single-fault ATPG algorithm, which must be overcome by compaction. The required test, v_3, would have been found if we targeted both faults F1 and F2 together and sought a common test. The problems of, (1) identifying suitable target fault sets and (2) concurrent test vector generation have been addressed in a recent paper [9], however, the solutions presented there are based on ATPG and hence are highly complex. We give alternative simulation-based solutions to these problems. Although, test generation for multiple target faults has been addressed in the literature [6, 7, 11], the algorithms and applications presented here are novel.

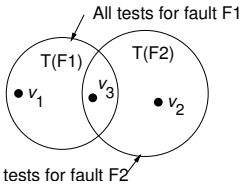


Figure 1. Venn diagram showing v_3 as a concurrent test for faults F1 and F2.

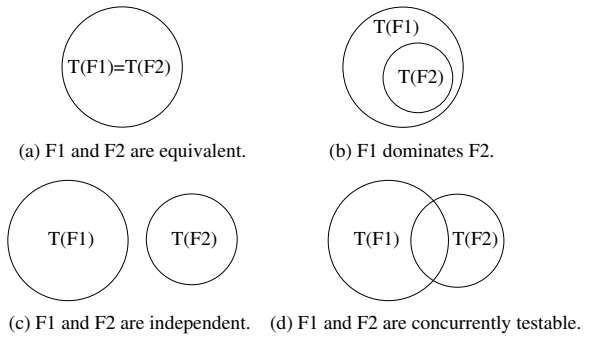
2. Concurrently-Testable Faults

Equivalence fault collapsing generally provides the target faults for ATPG. However, equivalence is only one among four possible test conditions that can exist between two faults. These are shown in Figure 2, where T(Fi) denotes the set of all test vectors for fault Fi.

For ATPG, faults are frequently collapsed via equivalence or dominance [5]. In equivalence collapsing, the faults are partitioned into disjoint equivalent sets and then one fault from each equivalent set is targeted by the ATPG. Detection of the targeted faults thus implies detection of all faults. In dominance collapsing, the target set is further reduced. When two faults, F1 and F2, in an equivalence collapsed set satisfy the relation T(F1) $\supset$ T(F2), meaning F1 *dominates* F2, fault F1 is dropped from the target fault list.

An equivalence collapsed fault set always, and a dominance collapsed set mostly, generates tests covering all or most faults. However, the number of test vectors generated is often significantly larger than the essential minimum number required. The reason for this is explained by Figures 2 (c) and (d). Two faults, F1 and F2, in the collapsed set can be either *independent* [3, 4], i.e., they have no common test, or *concurrently-testable*, i.e., they have common tests.

1081-7735/05 $20.00 ©2005 IEEE



(a) F1 and F2 are equivalent. (b) F1 dominates F2.

(c) F1 and F2 are independent. (d) F1 and F2 are concurrently testable.

Figure 2. Relations of faults F1 and F2 with test sets T(F1) and T(F2), respectively.

Concurrently-testable faults are also called "compatible." In the absence of any knowledge of these behaviors, we target both faults. If they are independent then we get two tests, which are essential. If they are concurrently testable then we may get one vector (if we were lucky) or two vectors, although only one would have been sufficient. Thus, independence and concurrently-testable properties can be useful in improving the efficiency of tests. We observe [9]:

- If two faults are independent, then no concurrent test is possible for them. A trivial case consists of two faults (with opposite polarity) of the same line.
- If two faults are equivalent, then any test for either fault is a concurrent test for both.
- If one fault dominates the other fault, then any test for the dominated fault is a concurrent test for both faults.
- Two faults having neither a concurrent test nor an exclusive test [1], are both redundant.

3. Independence Fault Collapsing

For the benchmark circuit c17, 11 s-a-1 faults, marked as 1 through 11 in Figure 3, form a functional dominance collapsed fault set [14]. A graph method of independence fault collapsing has been described in a recent paper [9]. These faults are represented as nodes of an *independence graph* (also knows as "incompatibility graph") and the independence of a pair of faults is expressed by an undirected edge between them. Independence fault collapsing groups faults into sets of concurrently-testable faults. The smallest number of such groups is the size of the largest *clique* or the *chromatic number* of the independence graph. The problem of finding the largest clique is NP-hard and is usually solved by

Table 1. Independence fault collapsing of c17 faults.

Fault Group No.	Faults (see Figure 3)	Concurrent test vector
1	1, 8	10010
2	2, 3, 9	01111
3	4, 6, 10	10101
4	5, 7, 11	x1010

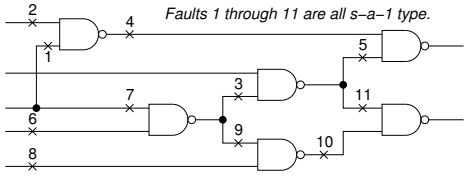


Figure 3. Functional dominance collapsed faults [14] of c17 circuit.

graph-coloring heuristics [2]. Simple greedy heuristics, however, result in groups of faults that are pairwise concurrently-testable but different pairs within a group may require separate tests. A "similarity heuristic" [9] tries to group those faults together that are likely to have a single concurrent test. That algorithm collapsed the 11 faults of c17 into four concurrent groups shown in Figure 4 and Table 1.

4. Concurrent Test Generation

There are several difficulties in implementing the procedures used in the example of Section 3. First, functional dominance fault collapsing [14], used prior to independence collapsing, is based on ATPG and is complex. Second, the independence graph generation procedure [9] is also based on ATPG. Finally, the use of concurrent D-algebra [8] requires a new ATPG program that may not be readily available to a user. In this section, we give an alternative procedure using the conventional fault simulator and single-fault ATPG programs for concurrent test generation. The only requirement is that the fault simulator should simulate without fault dropping, as is usually needed in diagnosis applications. The procedure is illustrated for the 4-bit ALU (74181) circuit.

Initial Fault Set. We begin with the structural equivalence collapsed fault set as obtained from any ATPG or fault simulation program. For the 4-bit ALU circuit this set contains 301 stuck-at faults including 8 redundant faults. Exclusive-OR gates in the circuit were expanded as four NAND gates each.

Independence Graph. Assuming no prior information about the concurrent detectability (compatibility) of faults, initially a fully connected independence graph was constructed for 301 faults.

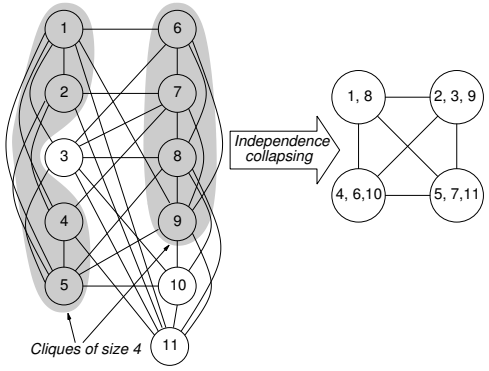


Figure 4. Independence graph (left) and fault collapsing for c17 circuit.

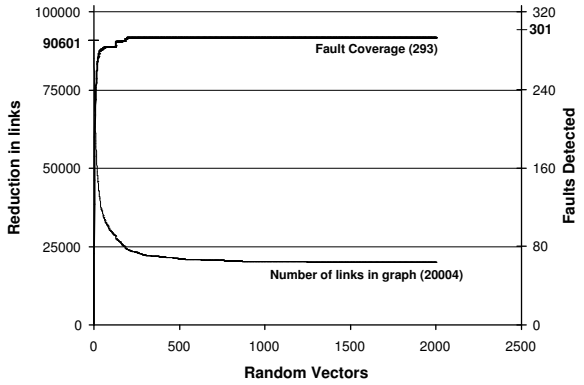


Figure 5. Random vector fault simulation to obtain independence graph of 4-bit ALU.

This graph contains $301 \times 301 = 90,601$ edges. The fault simulator Hope [13] was used to simulate random vectors without fault dropping. The upper curve in Figure 5 shows the fault coverage reaching 293 (all detectable faults) at vector number 193. All edges among the faults detected by each vector were deleted from the independence graph, reducing the number of edges down from 90,601 as shown by the lower curve in Figure 5. For example, the first vector detected 73 faults and caused the deletion of $73 \times 73 = 5,329$ edges. The simulation was stopped at 2,000 random vectors when it was found that about 200 vectors did not remove any new edge. This left 20,004 or about 22% edges. In this graph, there were 8 nodes that were still connected to all other nodes giving them a degree of independence 301. An ATPG was used to derive tests or prove redundancy. Since these faults were found to be redundant, the corresponding nodes and all edges attached to them were removed leaving 293 nodes.

The similarity-based independence collapsing algorithm [9] grouped 293 faults into 13 groups with sizes ranging from 6 to 81 faults (Table 2).

Concurrent Test Generation. Our objective is to generate a single test vector to cover all or most

Table 2. Simulation-based concurrent test generation for 4-bit ALU (74181) circuit.

Group no.	Number of faults	Test vector (for bit order see [8])
1	9	01100011111100
2	15	01101100000110
3	11	10100101111010
4	6	11011010100000
5	11	10110101011010
6	17	10100111101010
7	11	10010101001110
8	16	01000111101011
9	16	11100010010011
10	22	11011100110100
11	22	01010001100001
12	56	No test needed.
13	81	10101001110110

faults in each group. The independence collapsing algorithm [9] orders the faults within each group in order of their *degree of independence* (DI) [9]. DI is simply the number of edges attached to a fault node in the independence graph. A fault with higher DI is likely to be detectable by a smaller set of vectors.

For each group, we select the fault with highest DI and derive all test vectors for it. We then use a fault simulator to select a vector that detects most faults in the group. If more vectors than one detect the same number of faults within the group, then we select the one that detects most faults outside the group as well. The multiple test vector generation capability of Atalanta [12] and fault simulation without fault dropping in Hope [13] were used.

Since the number of test vectors for a fault can be very large and the vectors may contain don't care bits, for the 4-bit ALU we limited the number of vectors for a target fault to 250 and expanded each vector with don't cares into no more than 10 vectors. This heuristic produced one vector for each of the first 11 groups as shown in Table 2. All faults in group 12 were detected by these 11 vectors and the vector selected for group 13 detected all faults that were not detected by the previous vectors.

5. Results

Table 3 shows the results of independence fault collapsing (number of fault groups) and concurrent ATPG (number of vectors) for ripple-carry adders up to 32 bits, the 4-bit ALU and combinational benchmark circuits. For comparison, single-fault ATPG results are given. The column "Dyn*" gives one of the best reported result obtained by dynamic vector compaction [10]. "Min-Max" are the numbers of

Table 3. Concurrent ATPG test length.

Circuit	Indep. Collapse Groups	Number of vectors		
		Conc. ATPG	Single-fault ATPG Dyn*	Min-Max
1-b adder	5	5		5-7
2-b adder	5	5		7-9
4-b adder	5	5		8-11
8-b adder	7	7		10-15
16-b adder	7	9		13-22
32-b adder	7	11		17-25
4-b ALU	13	12		22-40
c17	4	4		6-9
c432	30	34	27	49-77
c499	52	52	52	54-68
c880	24	29	16	52-106
c1355	84	84	84	85-109
c1908	106	111	106	118-173
c2670	81	92	44	106-192
c3540	107	130	84	147-263
c5315	92	104	37	114-224
c6288	23	25	12	32-48
c7552	190	198	73	209-358

* Dynamic compaction (Hamzaoglu and Patel [10])

statically compacted and uncompacted vectors generated by Atalanta [12].

For ripple-carry adders, the minimum number of vectors is 5, irrespective of the adder size. The number of concurrent ATPG vectors grows, though at a slower rate than the compacted Atalanta vectors. The 4-bit ALU result is optimum. For several benchmarks, concurrent ATPG produced a minimal vector set. For others it produced more vectors than the best reported [10]. The number of collapsed groups indicates approximately how many vectors will be generated. For many circuits, the number of groups would be reduced if we simulated more random vectors causing deletion of additional independence links from the graph. Indeed, the graph generation procedure needs improvement. Our concurrent ATPG requires all vectors for a single fault and had to be restricted when there were too many such vectors. Alternative ATPG algorithms, such as concurrent D-algebra [8] and simulation-based directed-search [7], may be investigated in the future.

6. Conclusion

When combined with the independence fault collapsing, concurrent test generation can produce compact tests. Even with the future improvements of the procedures we cannot expect an absolute optimality for large circuits due to the complexity of the problem; ATPG and set covering problems have exponential time complexities. Still, independence fault collapsing and concurrent ATPG are novel concepts whose potential benefits are yet to be realized.

References

[1] V. D. Agrawal, D. H. Baik, Y. C. Kim, and K. K. Saluja, "Exclusive Test and Its Applications in Fault Diagnosis," in *Proc. 16th International Conf. VLSI Design*, Jan. 2003, pp. 143–148.

[2] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data Structures and Algorithms*. Reading, Massachusetts: Addison-Wesley, 1987.

[3] S. B. Akers, C. Joseph, and B. Krishnamurthy, "On the Role of Independent Fault Sets in the Generation of Minimal Test Sets," in *Proc. International Test Conf.*, 1987, pp. 1100–1107.

[4] S. B. Akers and B. Krishnamurthy, "Test Counting: A Tool for VLSI Testing," *IEEE Design & Test of Computers*, vol. 6, no. 5, pp. 58–77, Oct. 1989.

[5] M. L. Bushnell and V. D. Agrawal, *Esentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*. Boston: Springer, 2005.

[6] J.-S. Chang and C.-S. Lin, "Test Set Compaction for Combinational Circuits," *IEEE Trans. on CAD*, vol. 14, no. 11, pp. 1370–1378, Nov. 1995.

[7] K. T. Cheng and V. D. Agrawal, *Unified Methods for VLSI Simulation and Test Generation*. Boston: Kluwer Academic Publishers, 1989.

[8] A. S. Doshi, "Independence Fault Collapsing and Concurrent Test Generation," Master's thesis, Auburn University, Dept. of ECE, Auburn, Alabama, 2005. *In preparation.*

[9] A. S. Doshi and V. D. Agrawal, "Independence Fault Collapsing," in *Proc. 9th VLSI Design and Test Symp.*, Aug. 2005, pp. 357–364.

[10] I. Hamzaoglu and J. H. Patel, "Test Set Compaction Algorithms for Combinational Circuits," *IEEE Trans. on CAD*, vol. 19, no. 8, pp. 957–963, Aug. 2000.

[11] S. Kajihara, I. Pomeranz, K. Kinoshita, and S. M. Reddy, "Cost Effective Generation of Minimal Test Sets for Stuck at Faults in Combinational Logic Circuits," *IEEE Trans. on CAD*, vol. 14, no. 12, pp. 1496–1504, Dec. 1995.

[12] H. K. Lee and D. S. Ha, "Atalanta: An Efficient ATPG for Combinational Circuits," Tech. Report 93-12, Dept. of Electrical Eng., Virginia Polytechnic Inst. and State Univ., Blacksburg, Virginia, 1993.

[13] H. K. Lee and D. S. Ha, "HOPE: An Efficient Parallel Fault Simulator for Synchronous Sequential Circuits," *IEEE Trans. on CAD*, vol. 15, no. 9, pp. 1048–1058, Sept. 1996.

[14] R. K. K. R. Sandireddy and V. D. Agrawal, "Diagnostic and Detection Fault Collapsing for Multiple Output Circuits," in *Proc. Design, Autom. & Test in Europe (DATE) Conf.*, Mar. 2005, pp. 1014–1019.