Antitest, Exclusive Test, and Concurrent Test

Vishwani D. Agrawal
Agere Systems
600-700 Mountain Avenue
Murray Hill, NJ 07974
va@agere.com

Kewal K. Saluja
University of Wisconsin
Dept. of Electrical and Computer Eng.
Madison, WI 08854
saluja@engr.wisc.edu

February 22, 2004

Abstract

Any input vector that does not detect a fault is called an antitest for that fault. Given two faults an input vector that detects exactly one of those faults is an exclusive test. A concurrent test for two faults must detect both faults. The problems of generating these three types of tests involve detection of multiple-faults. We transform each of these problems into a single-fault ATPG problem. Several properties and applications of these tests are cited. A simple example illustrates the application of exclusive test in improving the diagnostic resolution of a fault dictionary.

1 Introduction

We define three new types of tests for a combinational circuit. These are "antitest", "exclusive test", and "concurrent test". Given a target fault, an antitest is an input vector that does not detect the target fault. Given a pair of target faults, a concurrent-test is an input vector that detects both faults. Similarly, for a pair of faults, an exclusive test detects any one (not both) fault. It is shown that the antitest problem can be translated into a single-fault detection problem by constructing an output modeling circuit (OMC). The OMC is a single-input single-output logic block with a single stuck-at-1 fault such that the fault-free OMC passes the input unchanged. The faulty OMC inverts the input. An antitest is a test for the multiple-fault consisting of the target fault and the s-a-1 fault in the OMC. A recently published method is used to model this multiple fault for generating an antitest by a single-fault ATPG program. To generate an exclusive test, we construct a miter circuit in which two copies of the circuit, each with a different single fault, feed an exclusive-OR gate that provides the primary output. A test for the double-fault is shown to be an exclusive test for the two faults. A concurrent test is derived as test for a double-fault in a double-miter circuit constructed with three copies of the circuit. One copy contains no fault and the other two copies contain one each of the two faults. The fault-free copy feeds into two exclusive-OR gates, whose other inputs come from the faulty copies. The exclusive-OR gates feed into an AND gate that generates the output of the double-miter.

Our use of multiple-faults should not cause confusion. They are used only to model the specific conditions imposed on the detection of single-faults. All tests we generate are aimed at detecting single stuck-at faults.

These three types of tests have applications in fault diagnosis. They may have other applications too. The purpose of this paper is to present novel definitions of tests and primary methods of their generation to the test community so as to initiate research on new test generation methods and applications.

2 Antitest

Figure 1 defines *test* and *antitest*. A fault-free digital circuit is shown as C_0. The other block C_1 is the same circuit with a fault f_1 permanently introduced. The circuit is assumed to be combinational and can have any number of inputs. For clarity of introducing a new topic, we will only consider single output functions in this paper. Any input vector that produces a 1 output in Figure 1(a) is a test for the fault f_1. This is the *Boolean satisfiability* version of the test generation problem, which can be solved by using two-valued (0,1) or three-valued (0,1,X) signals. The commonly used auto-

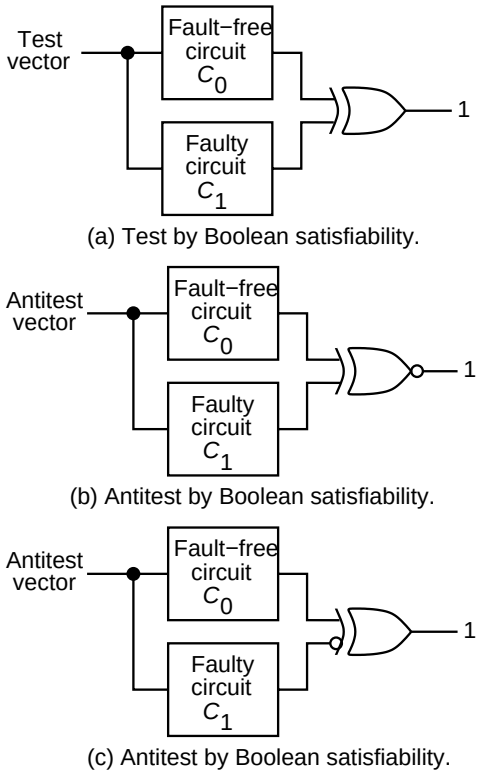

Figure 1: Boolean satisfiability definitions of test and antitest.

matic test pattern generation (ATPG) programs collapse C_0 and C_1 into one copy of the circuit and employ either a five-valued (0,1,D,$\overline{D}$,X) [9] or nine-valued (0,1,X,0/1,0/X,1/0,1/X,X/0,X/1) signal representations [7].

Two Boolean satisfiability formulations of the antitest problem are illustrated in Figures 1(b) and (c). In the next subsection, we will reformulate this problem for five or nine-valued signal ATPG. This definition covers all possibilities that prevent a vector from detecting the fault. These are:

1. Fault f_1 is not activated.
2. Fault f_1 is activated but no path from the fault site to the circuit output is sensitized.

The construction of Figure 1(a) is known as *miter* and is often used for equivalence checking [5]. It completely incorporates the conditions of test or antitest. We make the following observations from Figure 1(a):

- For any given fault, tests and antitests form disjoint sets that cover the entire input vector space.
- For a single stuck-at fault on a primary input line, the number of antitests must either equal or exceed half of all vectors.

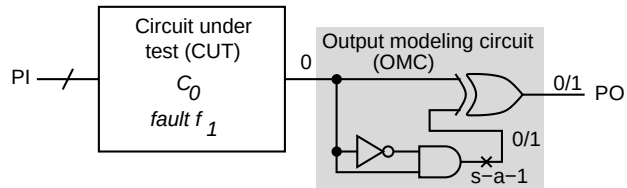

Figure 2: Output modeling circuit (OMC) to transform antitest to test problem.

- Every input is an antitest for a redundant fault.
- If f_1 and f_2 are two stuck-at faults on the same line (with opposite polarity), then any test for f_2 is an antitest for f_1. In general, however, f_1 may have additional antitests that do not detect f_2.
- For any test for a target fault, f_1, the two faults f_1 and output s-a-1 mask each other. Thus, the generation of an antitest is *equivalent to the generation of a test for the multiple-fault, f_1 and miter output s-a-1 in Figure 1(a).*

2.1 Antitest Generation

The use of the miter of Figure 1(a) for vector generation presents two problems. First, the isolation of the fault-free and faulty circuits requires the use of two-valued (0,1) or three-valued (0,1,X) logic. ATPG algorithms and programs, on the other hand, use a single copy of the circuit with either a five-valued or a nine-valued logic [3]. The second problem is that we must generate a test for a multiple-fault. Once again, ATPG programs can only handle single-faults. We, therefore, devise a model to generate antitests using any conventional ATPG program. This is done in two steps: output modeling circuit and a single-fault model for a multiple-fault.

Output Modeling Circuit. Figures 1(b) and (c) give two ways of expressing the antitest problem as Boolean satisfiability:

$$\overline{C_0 \oplus C_1} = 1 \quad \text{and} \quad C_0 \oplus \overline{C_1} = 1 \qquad (1)$$

The second equation can be realized by appending a one-input one-output *output modeling circuit* (OMC) with a single stuck-at fault to the circuit under test (CUT). In its fault-free state OMC propagates its input signal to the output. In the faulty state, it inverts the input signal at the output. Such a circuit is easily realized with three gates as shown in Figure 2. Irrespective of whether the input to the OMC is 0 or

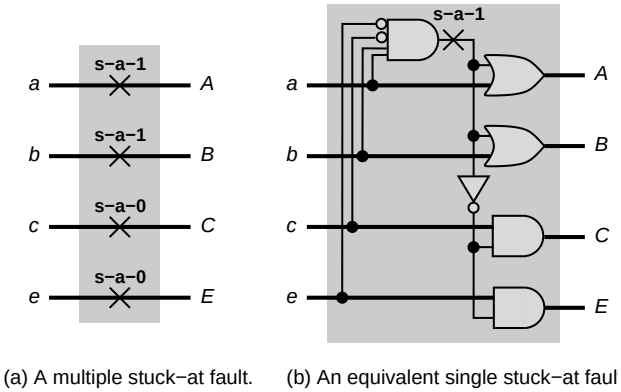

Figure 3: A multiple-fault modeled as a single fault [6].

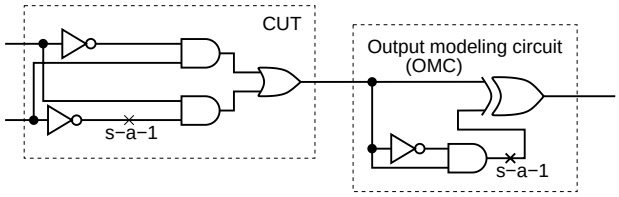

Figure 4: An example circuit with a target fault for antitest.

1, the s-a-1 fault is activated and propagated to the output. The example in the figure illustrates this for a 0 input, which produces 0/1. *We use the nine-value notation for signals so 0 means the signal is 0 for both fault-free and faulty cases. A value 0/1 refers to 0 for fault-free and 1 for faulty circuit.* Thus, the second Boolean satisfiability problem of Equation 1 can be written as,

$$OMC(C_0) \oplus OMC_{sa1}(C_1) = 1 \qquad (2)$$

This is the normal ATPG problem for detection of a multiple fault (f_1 in C_0 and s-a-1 in OMC). When f_1 is activated in C_0 and is propagated to the input of OMC, the fault in C_0 and the s-a-1 fault in OMC mask each other. The reader can easily verify that the OMC in such cases propagates the fault-free output of CUT. Any test for a multiple-fault that includes a target fault in CUT and the s-a-1 fault in OMC is an antitest for the target fault.

Multiple-Fault Modeling. We use a single-fault model for the multiple-fault as given in a recent paper [6]. This model is created by inserting two-input gates (OR for s-a-1 or AND for s-a-0) at the sites of all components of the multiple-fault. Further, the signals from these sites are ANDed, directly for s-a-1 faults and with inversion for s-a-0 faults. The output of this AND gate, which contains a single s-a-1 fault,

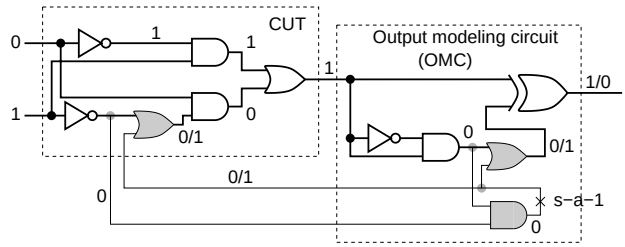

Figure 5: Derivation of antitest using single-fault ATPG.

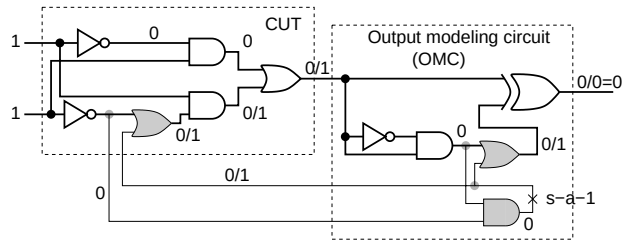

Figure 6: Masking of a test during antitest generation.

is fed to all in-line OR gates and with inversion to all in-line AND gates. An illustration of this model is shown in Figure 3, which is reproduced from a recent paper [6]. In this illustration a multiple stuck-at fault affects four lines of the circuit shown as aA, bB, cC and dD. The two circuits in Figure 3 are functionally identical if all faults are removed. A test for the single s-a-1 fault in Figure 3(b) will detect the multiple fault in (a). It is also shown [6] that the single-fault model remains functionally combinational (free from asynchronous feedback) even though the model in some cases may have structural feedback.

2.2 An Antitest Example

We now illustrate the antitest generation by an example. Figure 4 is an exclusive-OR function. We will derive an antitest for the target fault s-a-1 at the output of the second inverter. According to the forgoing discussion, any test for the multiple-fault consisting of the two faults shown in Figure 4 is an antitest for the target fault.

Figure 5 provides a single-fault model for the two faults in Figure 4. All gates inserted for fault modeling are shaded gray and their interconnects are drawn in thin lines. The signal values correspond to a test for the multiple-fault, which is also an antitest for the target fault. Notice that the target fault is activated but is not propagated to the output of CUT.

Figure 6 shows how the fault effect is masked by OMC. Here, the fault is activated and a 0/1 ap-

pears at the output of CUT. The exclusive-OR gate in OMC receives fault effects on both inputs, and produces a 0 at the output. Since an ATPG program will always try to generate an input vector that produces the fault effect at the output, the type of test in Figure 6 will never be generated.

3 Exclusive Test

The general exclusive test problem can be formulated for a pair of fault sets. For simplicity, however, we will study it for a pair of faults, f_1 and f_2. An exclusive test must detect exactly one of these faults. Once again, we denote the fault-free circuit function as C_0 and the two faulty functions as C_1 and C_2. The Boolean satisfiability formulation of the exclusive test problem is,

$$(C_0 \oplus C_1) \oplus (C_0 \oplus C_2) = 1 \tag{3}$$

which simplifies to,

$$C_1 \oplus C_2 = 1 \tag{4}$$

and may also be expressed as,

$$(C_0 \oplus C_0) \oplus (C_1 \oplus C_2) = 1 \tag{5}$$

Equation 4 is a condition for distinguishing between the two faults. The complementary condition, $C_1 \oplus C_2 = 0$, is known as the *indistinguishability condition* [3], and when satisfied by all inputs makes the two faults equivalent.

Equation 5 indicates that an exclusive test is a test for a double fault in two copies of the CUT producing a single output through an exclusive-OR gate, and each copy contains a different single fault. The Boolean satisfiability and multi-valued ATPG formulations of the exclusive test problem are illustrated in Figure 7.

We note the following observations about exclusive tests:

- If no exclusive test exists for two faults then either they are equivalent or both faults are redundant. Although not often recognized, all redundant faults of a combinational circuit form an equivalent fault set.
- If two faults are independent then any test that detects either one of them is an exclusive test for the pair. Two faults are defined as *independent* [1] if no vector can detect both of them.
- An exclusive test for a pair of faults is a test for one fault and antitest for the other.

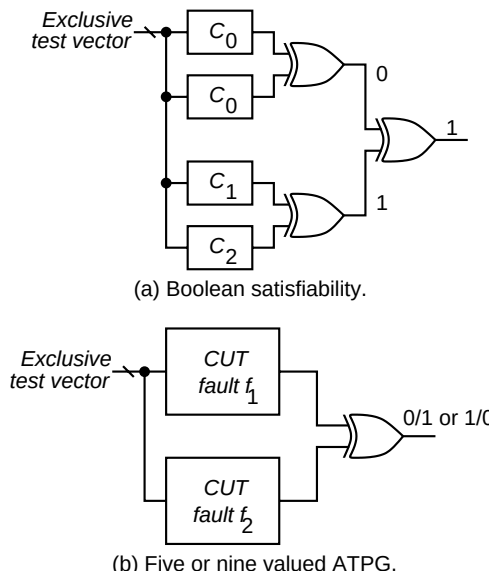

Figure 7: Generation of exclusive test.

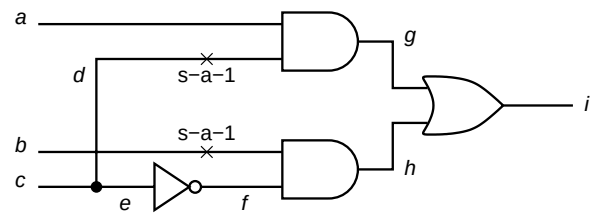

Figure 8: A multiplexer example for diagnostic tests.

3.1 Application of Exclusive Test to Diagnosis

We can illustrate the use of exclusive test in the generation of a diagnostic dictionary [3]. Consider the multiplexer circuit shown in Figure 8. Five tests, $T_1 = 001$, $T_2 = 010$, $T_3 = 011$, $T_4 = 100$ and $T_5 = 101$, were generated by an ATPG program [4]. All ten faults in the equivalence fault set $(a_1, b_1, c_0, c_1, d_1, f_1, g_0, h_0, i_1, i_0)$ were detected by this test set. The fault dictionary is given in Table 1. For the five tests, each fault has a test syndrome that is a five-bit vector. The jth bit t_j of this vector is 1 only if test T_j detects that fault. The table also gives a diagnostic number computed as,

$$DN(fault) = \sum_{j=1}^{n} t_j 2^{j-1}$$

where n is the number of tests. Thus, the fault a_1 (a s-a-1), which is detected by T_1 and T_3, has a test syndrome 10100 and its diagnostic number is 5.

We notice that this dictionary cannot distinguish between faults b_1 and d_1 since they have the same test

Table 1: Fault dictionary enhancement by exclusive test for the multiplexer of Figure 8.

Fault	100% coverage tests		Exclusive test added	
	Test syndrome	Diag. number	Test syndrome	Diag. number
a_1	10100	5	101000	5
b_1	**00010**	**8**	000101	40
c_0	00101	20	001010	20
c_1	01010	10	010100	10
d_1	**00010**	**8**	000100	8
f_1	00100	4	001000	4
g_0	00001	16	000010	16
h_0	01000	2	010000	2
i_0	01001	18	010010	18
i_1	10110	13	101101	45

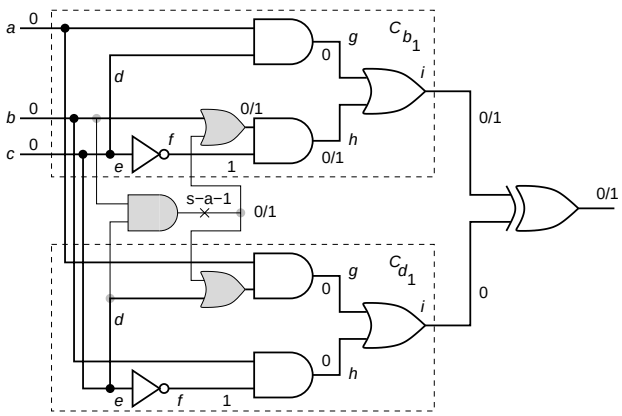

Figure 9: Exclusive test generation for fault pair (b_1, d_1).

syndrome and diagnostic number (shown in boldface in Table 1.) The procedure of the previous subsection is used in Figure 9 to generate an exclusive test for these faults. The modeling allows us to use a single-fault ATPG [4], which generates the vector $T_6 = 000$. This test detects two faults b_1 and i_1. When it is appended to the existing vector set, the resulting test syndromes and diagnostic numbers are shown in the two last columns of Table 1. Now every fault in the list can be uniquely diagnosed.

4 Concurrent Test

Although the general problem of concurrent test relates to a set of faults, for simplicity, we will consider two faults, f_1 and f_2. Figure 10 shows the Boolean satisfiability and multi-valued ATPG formulations of the concurrent test problem. As before, C_0 is the fault-free function and C_1 is the function with fault f_i permanently injected. The Boolean satisfiability

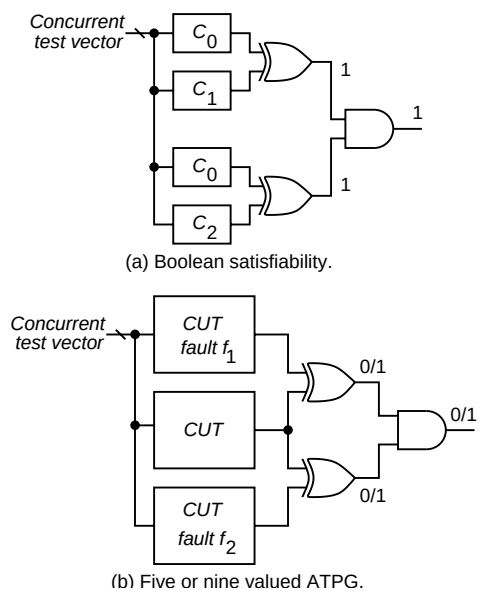

Figure 10: Generation of concurrent test.

problem is,

$$(C_0 \oplus C_1).(C_0 \oplus C_2) = 1 \tag{6}$$

Notice that the ATPG requires three copies of the circuit (CUT) and the detection of a multiple fault whose components, f_1 and f_2, are in two separate copies. We make the following observations:

- If two faults are independent [1], 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 two faults have neither a concurrent test nor an exclusive test, then both faults are redundant.

Table 2: Concurrent test generation for multiplexer.

Targeted faults	Concurrent test	New faults detected
a_1 and b_1	No test	
d_1 and b_1	100	$b_1,\ c_1,\ d_1,\ i_1$
a_1 and c_0	No test	
g_0 and c_0	101	$c_0,\ g_0,\ i_0$
a_1 and f_1	011	$a_1,\ f_1$
h_0	010	h_0

4.1 Application of Concurrent Test to Compact Test Generation

A compact fault detection test set, although not as good for the diagnosis application of Subsection 3.1, is sometimes desirable for reducing the test time, especially in scan testing. For compact test generation, fault targets are selected by heuristics and the resulting tests can be reduced by fault simulation based compaction techniques [8]. Alternatively, one can generate compact tests using an independent fault set (IFS) [1] or the method of test counting [2]. No two faults in an IFS can be detected by the same test. While finding a complete (or the largest) IFS is a complex problem, a subset can be easily found around a gate near primary inputs. Our illustration is based on such a partial IFS.

The reader can verify that faults a_1, d_1 and g_0 belong to an IFS for the multiplexer of Figure 8. We form fault pairs containing exactly one fault from this partial IFS. Concurrent tests are generated for these pairs. Whenever no test exists for a pair, the fault outside of IFS is successively paired with all faults in the IFS. If no pair succeeds in providing a test, then that fault is added to the IFS. The procedure terminates when all faults within and outside IFS are detected. This procedure applied to the multiplexer produced the result of Table 2.

The set of four vectors found here is the smallest for this circuit. In the next to the last step of Table 2, had we paired a_1 with h_0, we would find no test and would eventually add h_0 to the IFS. That would complete the IFS, which has a cardinality of 4.

5 Conclusion

This paper is an attempt to present new ideas that have not been discussed before. Our search of the literature has found no references to the three types of tests, namely, antitest, exclusive test, and concurrent test, introduced in this paper. We would like the reader to point us to references on similar prior work if they have seen any. Readers may find some of the ideas trivial, while others meaningful. We would like their remarks. The motivation for this article is two-fold. First, having given the definitions of the new types of tests, we attempt to find modeling and test generation algorithms. It is our belief that the techniques given here will have many future improvements. Our motivation is that some readers will contribute their expertise. Second, we are certain the material has novel applications not cited here. If readers cannot find them then the only reason would be that their imagination is restricted in the same way as ours. We unconditionally thank the present readers (reviewers) since their feedback is important to our research. Their decision will determine whether or not other readers benefit from this work.

References

[1] 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.

[2] 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.

[3] M. L. Bushnell and V. D. Agrawal, *Esentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits.* Boston: Kluwer Academic Publisher, 2000.

[4] W.-T. Cheng and T. J. Chakraborty, "Gentest: An Automatic Test Generation System for Sequential Circuits," *Computer*, vol. 22, no. 4, pp. 43–49, Apr. 1989.

[5] S.-Y. Huang and K.-T. Cheng, *Formal Equivalence Checking and Design Debugging.* Boston: Kluwer Academic Publisher, 1998.

[6] Y. C. Kim, V. D. Agrawal, and K. K. Saluja, "Multiple Faults: Modeling, Simulation and Test," in *Proc. 15th International Conf. VLSI Design*, Jan. 2002.

[7] P. Muth, "A Nine-Valued Circuit Model for Test Generation," *IEEE Trans. on Electronic Computers*, vol. C-25, no. 6, pp. 630–636, June 1976.

[8] I. Pomeranz, L. N. Reddy, and S. M. Reddy, "COMPACTEST: A Method to Generate Compact Test Sets for Combinational Circuits," *IEEE Trans. Computer-Aided Design*, vol. 12, no. 7, pp. 1040–1049, July 1993.

[9] J. P. Roth, W. G. Bouricius, and P. R. Schneider, "Programmed Algorithms to Compute Tests to Detect and Distinguish Between Failures in Logic Circuits," *IEEE Trans. on Electronic Computers*, vol. EC-16, no. 5, pp. 567–580, Oct. 1967.