# 7. TESTING FOR BRIDGING FAULTS

### About This Chapter

Shorts between normally unconnected signals are manufacturing defects that occur often in chips and on boards. Shorts are modeled as bridging faults. In this chapter we present fault simulation and test generation methods for bridging faults. These methods rely on an analysis of relations between single stuck faults and bridging faults.

## 7.1 The Bridging-Fault Model

*Bridging faults* (BFs) are caused by shorts between two (or more) normally unconnected signal lines. Since the lines involved in a short become equipotential, all of them have the same logic value. For a shorted line $i$, we have to distinguish between the value one could actually observe on $i$ and the value of $i$ as determined by its source element; the latter is called *driven value.* Figure 7.1 shows a general model of a BF between two lines $x$ and $y$. We denote such a BF by $(x.y)$ and the function introduced by the BF by $Z(x,y)$. The fanout of $Z$ is the union of the fanouts of the shorted signals. Note that the values of $x$ and $y$ in this model are their driven values, but these are not observable in the circuit.

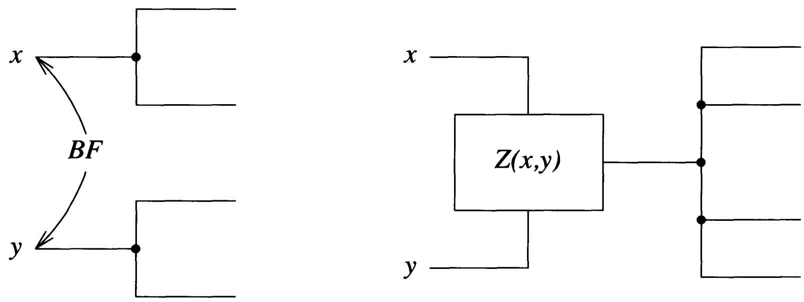



**Figure 7.1** BF Model

The function $Z$ has the property that $Z(a,a)=a$. What happens when $x$ and $y$ have opposite values depends on the technology. For example, in MOS the value of $Z(a,\bar{a})$ is, in general, indeterminate (i.e., the resulting voltage does not correspond to any logic value). In this section we will consider only BFs for which the values of the shorted lines are determinate. In many technologies (such as TTL or ECL), when two shorted lines have opposite driven values, one value (the *strong* one) overrides the other. If $c \in \{0,1\}$ is the strong value, then $Z(0,1)=Z(1,0)=c$, and the function introduced by the BF is AND if $c=0$ and OR if $c=1$.

Unlike in the SSF model, in the BF model there is no need to distinguish between a stem and its fanout branches, since they always have the same values. Thus the BFs we shall consider are defined only between gate outputs and/or primary inputs.

If there exists (at least) one path between $x$ and $y$, then a BF $(x.y)$ creates one or more feedback loops. Such a fault is referred to as a *feedback bridging fault* (FBF). A BF that does not create feedback is referred to as a *nonfeedback bridging fault* (NFBF). *An FBF transforms a combinational circuit into a sequential one* (see Figure 7.2). Moreover, if a feedback loop involves an odd number of inversions, the circuit may oscillate (see Figure 7.3). If the delay along the loop is small, the resulting oscillations have very high frequency and may cause the affected signals to assume indeterminate logic values. Such values may confuse the testing equipment.

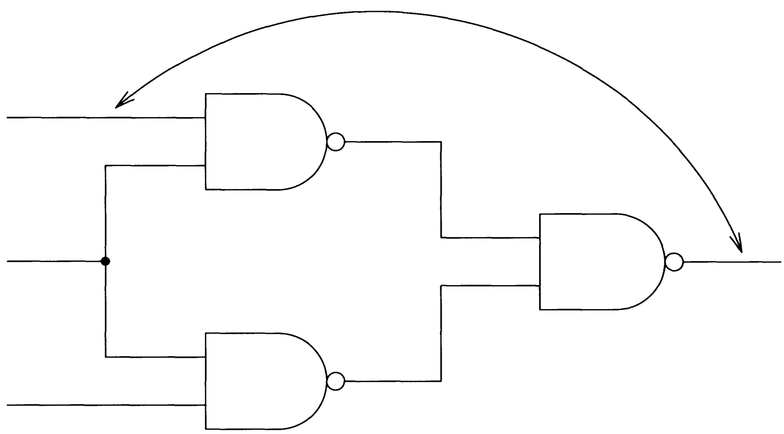


**Figure 7.2** Example of an FBF

The *multiple bridging-fault (MBF) model* represents shorts involving more than two signals. An MBF with $p>2$ shorted lines can be thought of as composed of $p-1$ BFs between two lines. For example, an MBF among lines $i$, $j$, and $k$ can be represented by the BFs $(i.j)$ and $(j.k)$. This model assumes that only one group of lines are shorted. For example, we cannot have both $(a.b)$ and $(c.d)$ present in the circuit. Although masking relations may occur among the components of an MBF [Mei 1974], most MBFs are detected by the tests designed to detect their component BFs. Also MBFs are less likely to occur. Thus in the following we shall consider only BFs between two lines.

The number of theoretically possible BFs in a circuit with $G$ gates and $I$ primary inputs is $b = \binom{G+I}{2}$. For $G>>I$, $b = G(G-1)/2$, so the dominant factor is $G^2$. However, this figure assumes that a short may occur between any two lines. In most circumstances a

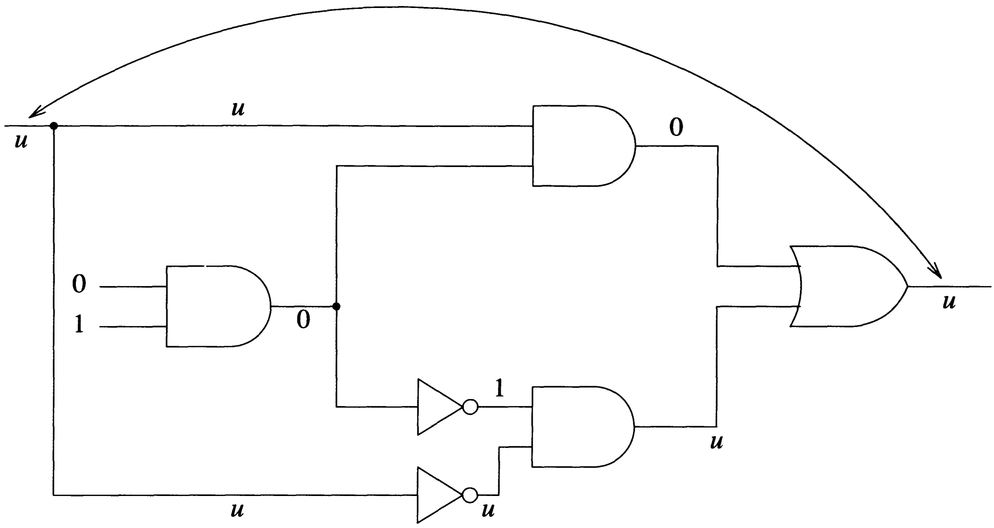



**Figure 7.3** Oscillation induced by an FBF

short only involves physically adjacent lines. Let $N_i$ be the "neighborhood" of $i$, i.e., the set of lines physically adjacent to a line $i$. Then the number of *feasible* BFs is

$$b = \frac{1}{2} \sum_i \mid N_i \mid$$

Denoting by $k$ the average size of $N_i$, we have

$$b = \frac{1}{2}(G+I)k$$

In general, $k$ increases with the average fanout count $f$. Let us assume that every stem and fanout branch has, on the average, $r$ distinct neighbors, where $r$ depends on the layout. Then

$$b = \frac{1}{2}(G+I)r(1+f)$$

Comparing this result with the number of SSFs derived in Chapter 4, we can conclude that *the number of feasible BFs is of the same order of magnitude as the number of SSFs in a circuit.* But since $r$ is usually greater than 2, the number of feasible BFs is usually greater. Moreover, the number of SSFs to be analyzed can be reduced by using structural equivalence and dominance relations, but similar collapsing techniques have not been developed for BFs.

However, the BF model is often used *in addition to* the SSF model. To take advantage of this, we will first analyze relations between the detection of SSFs and the detection of BFs. Then we will present fault simulation and test generation methods for BFs that exploit these relations, so that *the processing of the two fault models can* *be combined.* In this way BF testing can be done with a small additional effort beyond that required for testing SSFs.

The following analysis considers only combinational circuits. Although we discuss only AND BFs, the results can be extended to OR BFs by interchanging 0s and 1s.

## 7.2 Detection of Nonfeedback Bridging Faults

**Theorem 7.1:** [Williams and Angell 1973] A test $t$ detects the AND NFBF $(x.y)$ iff either $t$ detects $x$ *s-a*-0 and sets $y = 0$, or $t$ detects $y$ *s-a*-0 and sets $x = 0$.

**Proof**

a. Assume the first condition holds, namely $t$ detects $x$ *s-a*-0 and sets $y = 0$. Hence $t$ activates $x$ *s-a*-0 (by setting $x = 1$) and the resulting error propagates to a primary output. The same effect occurs in the presence of $(x.y)$, while the value of $y$ does not change. Therefore $t$ detects $(x.y)$. The proof for the second condition is symmetric.

b. If neither of the above conditions holds, then we shall prove that $t$ does not detect $(x.y)$. Two cases are possible:

   *Case 1:* $x$ and $y$ have the same value. Then $(x.y)$ is not activated.

   *Case 2:* $x$ and $y$ have different values. Assume (without loss of generality) that $y = 0$. Then an error is generated at $x$. But since $t$ does not detect $x$ *s-a*-0, this error does not propagate to any primary output.

   Therefore, in either case $t$ does not detect $(x.y)$ □

We emphasize that lines $x$ and $y$ involved in a BF are gate outputs and/or primary inputs, *not* fanout branches. As illustrated in Figure 7.4, Theorem 7.1 does not hold if applied to fanout branches. Although both $h$ *s-a*-0 and $m$ *s-a*-0 are undetectable, the AND BF $(h.m)$ is detected by the test 110, because the fault effect on $h$ propagates from the stem $b$ (Theorem 7.1 applies for $b$ and $c$).

In general, there is no guarantee that a complete test set for SSFs will satisfy the conditions of Theorem 7.1 for every NFBF. However, detection can be guaranteed for certain types of BFs involving inputs of the same gate [Friedman 1974]. This is important because shorts between inputs of the same gate are likely to occur.

**Corollary 7.1:** Let $x$ and $y$ be signals without fanout. If $x$ and $y$ are inputs to the same OR or NOR gate, then the AND BF $(x.y)$ dominates both $x$ *s-a*-0 and $y$ *s-a*-0.

**Proof:** From Figure 7.5 we can observe that the values required to detect $x$ *s-a*-0 (and similarly, those required to detect $y$ *s-a*-0) satisfy the conditions of Theorem 7.1. Thus the AND BF $(x.y)$ dominates both $x$ *s-a*-0 and $y$ *s-a*-0. □

Note that, in general, this dominance relation does not extend to sequential circuits. For example, in the circuit of Figure 7.6 (starting with $y$=1), consider the OR BF $(x_1.x_2)$, which would dominate $x_2$ *s-a*-1 in a combinational circuit. Although the test sequence shown detects $x_2$ *s-a*-1, it does not detect $(x_1.x_2)$.

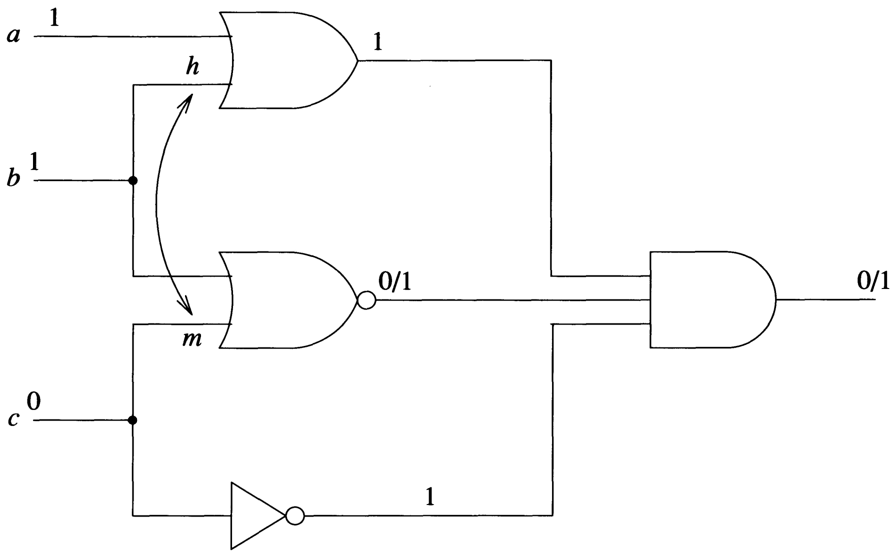



Figure 7.4

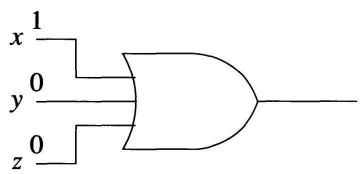


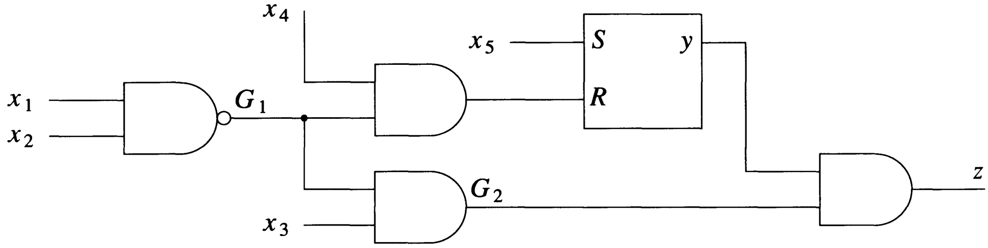

Figure 7.5

Now consider an AND BF between inputs of an OR gate, with only one of the shorted inputs having fanout. An argument similar to that used to prove Corollary 7.1 (refer to Figure 7.7) leads to the following result.

**Corollary 7.2:** Let $x$ be a line with fanout and $y$ a line without. If $x$ and $y$ are inputs to the same OR or NOR gate, the AND BF $(x.y)$ dominates $y$ *s-a-*0. □

BFs between inputs of the same gate that do not satisfy the condition of Corollaries 7.1 or 7.2 are not guaranteed to be detected by complete test sets for SSFs.

[Friedman 1974] has conjectured that in an irredundant combinational circuit, all BFs between inputs of the same gate, where both inputs have fanout, are detectable. To our knowledge, no counterexample to this conjecture has yet been found.




| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
|---|---|---|---|---|---|
| $t_1$ | 1 | 0 | 0 | 1 | 0 |
| $t_2$ | 1 | 0 | 0 | 0 | 0 |
| $t_3$ | 0 | 1 | 0 | 0 | 0 |
| $t_4$ | 0 | 1 | 1 | 0 | 0 |

**Figure 7.6**

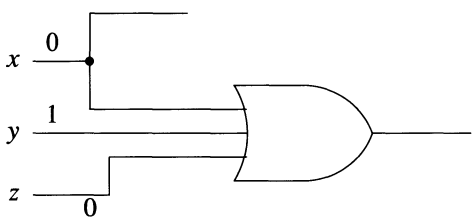



**Figure 7.7**

## 7.3 Detection of Feedback Bridging Faults

An FBF is created when there exists at least one path between the two shorted lines. We refer to the two lines involved in an FBF as the *back line* $b$ and the *front line* $f$, where $b$ is the line with the lower level (i.e., is closer to the primary inputs). Since a combinational circuit becomes a sequential one in the presence of an FBF, in general we need a test sequence to detect an FBF. However, we shall show that *in many cases an FBF can be detected by only one test* [Abramovici and Menon 1985].

**Theorem 7.2:** A test $t$ that detects $f$ *s-a-*0 and sets $b = 0$ detects the AND FBF $(b.f)$.

**Proof:** Independent of the state of the circuit before test $t$ is applied, $b$ is driven to 0, which is the strong value for the AND BF (see Figure 7.8). The driven value of $f$ is 1 and hence an error is generated at $f$. Since the value of $b$ is the same as in the fault-free circuit, the error propagates on the same path(s) as the error generated by $f$ *s-a*-0. Therefore $t$ detects $(b.f)$. □

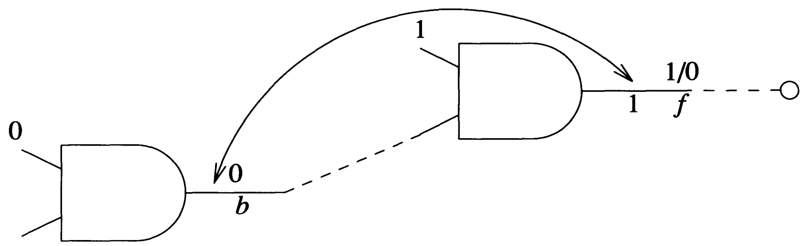



**Figure 7.8**

In the next theorem the roles of $b$ and $f$ are reversed.

**Theorem 7.3:** A test $t$ that detects $b$ *s-a*-0, and sets $f$=0 without sensitizing $f$ to $b$ *s-a*-0, detects the AND FBF $(b.f)$.

**Proof:** Since $f$ is not sensitized to $b$ *s-a*-0 by $t$, the value of $f$ does not depend on the value of $b$ (see Figure 7.9). Independent of the state of the circuit (with the FBF present) before $t$ is applied, the driven values of $b$ and $f$ are, respectively, 1 and 0. Then $t$ activates $(b.f)$ and the error propagates from $b$ along the path(s) sensitized by $t$ to detect $b$ *s-a*-0. Therefore $t$ detects $(b.f)$. □

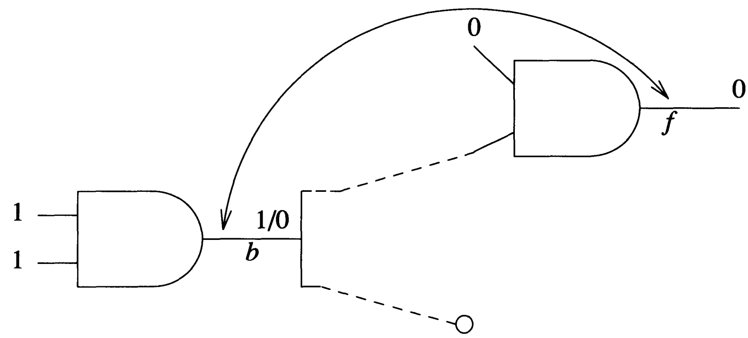



**Figure 7.9**

Note that a test $t$ that propagates $b$ *s-a*-0 through $f$ and sets $f$=0 induces an oscillation along the loop created by $(b.f)$ (see Figure 7.3); we will refer to such a BF as *potentially oscillating*.

Theorems 7.2 and 7.3 are valid independent of the number of paths between $b$ and $f$ and of their inversion parity. Now we shall prove that when all the paths between $b$ and $f$ have even parity, the conditions for the detection of $(b.f)$ can be relaxed.

**Corollary 7.3:** Let $(b.f)$ be an AND FBF such that all paths between $b$ and $f$ have even inversion parity. A test $t$ that either detects $f$ *s-a*-0 and sets $b = 0$, or detects $b$ *s-a*-0 and sets $f = 0$, also detects $(b.f)$.

**Proof:** The first condition is covered by Theorem 7.2. Let us analyze the second condition, namely $t$ detects $b$ *s-a*-0 and sets $f = 0$. We shall prove that line $f$ is not sensitized to the fault $b$ *s-a*-0 by $t$. Let us assume the contrary. Then, according to Corollary 4.1, the value of $f$ in $t$ should be $1 \oplus 0 = 1$; but this contradicts the assumption that $t$ sets $f = 0$. Hence $f$ does not lie on any path sensitized to $b$ *s-a*-0 by $t$. Then this case is covered by Theorem 7.3, and therefore $t$ detects $(b.f)$. □

All the preceding results in this section only give sufficient conditions for the detection of FBFs by one test. But an FBF can also be detected by a sequence of tests, none of which individually satisfies these conditions. The next example shows such a case.

**Example 7.1:** Consider the AND FBF $(a.z)$ in the circuit in Figure 7.10, and the tests $t_1 = 1010$ and $t_2 = 1110$. The only path between $a$ and $z$ has even inversion parity. Neither $t_1$ nor $t_2$ satisfies the conditions of Corollary 7.3. Nevertheless the sequence $(t_1, t_2)$ detects $(a.z)$, because the 0 value of $z$ in $t_1$ overrides the 1 value applied to $a$ in $t_2$, and the resulting error propagates to $z$. □

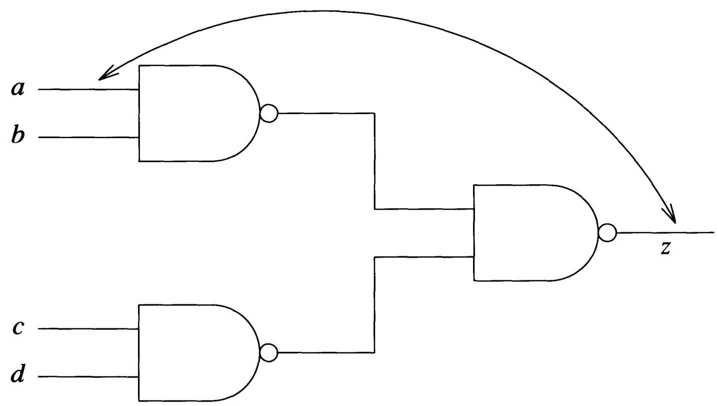



**Figure 7.10**

For certain cases we can derive conditions that are both necessary and sufficient for the detection of an FBF by one test.

**Corollary 7.4:** Let $(b.f)$ be an AND FBF such that any path between $b$ and a primary output goes through $f$. A test $t$ detects $(b.f)$ iff $t$ detects $f$ *s-a*-0 and sets $b$=0.

**Proof:** Sufficiency follows from Theorem 7.2. To prove necessity, we show that if $t$ does not detect $f$ *s-a*-0 or does not set $b$=0, then it does not detect $(b.f)$.

*Case 1:* Suppose that $t$ does not detect $f$ *s-a*-0. Since any path between $b$ and a primary output goes through $f$, an error generated by $(b.f)$ can propagate only from $f$. Then $t$ must detect $f$ *s-a*-1. Hence, $t$ sets $f$=0. Since 0 is the strong value for an AND BF, in the presence of $(b.f)$, $f$ can either maintain value 0 or it can oscillate. If $f = 0$, no error is generated. If $f$ oscillates, no definite error is generated.

*Case 2:* Suppose that $t$ detects $f$ *s-a*-0 but sets $b$=1. Again no error is generated at $f$.

Therefore in either case $t$ does not detect $(b.f)$. □

**Corollary 7.5:** Let $(b.f)$ be an AND FBF where $b$ and $f$ are such that $f$=1 whenever $b$=0 (in the fault-free circuit). A test $t$ detects $(b.f)$ iff $t$ detects $f$ *s-a*-0 and sets $b$=0.

**Proof:** Sufficiency follows from Theorem 7.2. To prove necessity, two cases must be considered.

*Case 1:* Suppose that $t$ sets $b$=1. If $t$ sets $f$=1, then no error is generated. If $t$ sets $f$=0, this would make $b$=0 because of the BF; but in the fault-free circuit $b$=0 implies $f$=1, and this is also true in the presence of $(b.f)$. Hence the loop created by $(b.f)$ oscillates and no definite error is generated.

*Case 2:* Suppose that $t$ sets $b$=0 but does not detect $f$ *s-a*-0. By assumption $b$=0 implies $f$=1. This means that $t$ activates $f$ *s-a*-0 but does not propagate its effect to a primary output. Similarly, the error generated by $(b.f)$ appears only at $f$ but is not propagated to a primary output.

Therefore, in either case $t$ does not detect $(b.f)$. □

An example of an FBF satisfying Corollary 7.5 is an FBF between an input and the output of a NAND gate.

The following result is a consequence of Corollary 7.4 and allows us to identify a type of FBF that is *not* detectable by one test.

**Corollary 7.6:** No single test can detect an AND FBF $(b.f)$ such that every path between $b$ and a primary output goes through $f$, and $f$=0 whenever $b$=0.

For example, if $b$ is an input of an AND gate whose output is $f$, and $b$ does not have other fanout, then $(b.f)$ is not detectable by one test.

## 7.4 Bridging Faults Simulation

In this section we discuss simulation of BFs. It is possible to explicitly simulate BFs by a process similar to the simulation of SSFs, based on determining when BFs are activated and on propagating their fault effects. This approach, however, cannot efficiently handle large circuits, because

- A BF is both structurally and functionally more complex than an SSF.
- The number of all feasible BFs is larger than that of SSFs.

Thus explicit fault simulation of BFs would be much more expensive than that of SSFs.

In the following we present an *implicit simulation method for BFs* in combinational circuits [Abramovici and Menon 1985]. This method uses the relations between the detection of SSFs and that of BFs to *determine the BFs detected by a test set without explicitly simulating BFs*. This is done by monitoring the occurrence of these relations during the simulation of SSFs.

Since we analyze the detection of BFs only by single tests, the method is approximate in the sense that it will not recognize as detected those BFs detected only by sequences of tests. Hence the computed fault coverage may be pessimistic; that is, the actual fault coverage may be better than the computed one.

We assume that layout information is available such that for every signal $x$ (primary input or gate output) we know its neighborhood $N_x$ consisting of the set of all lines that can be shorted to $x$. We consider only AND BFs. Note that for every BF $(x.y)$ we have both $x \in N_y$ and $y \in N_x$.

All the results in the previous sections relate the detection of AND BFs to the detection of *s-a-*0 faults. If a test that detects $x$ *s-a-*0 also detects the BF $(x.y)$, we say that $(x.y)$ is *detected based on* $x$. To simplify the simulation of BFs, we construct a *reduced neighborhood* $N_x'$ obtained by deleting from $N_x$ all lines $y$ such that $(x.y)$ is undetectable or undetectable based on $x$. If $x$ and $y$ are inputs to the same AND or NAND gate and they do not have other fanouts, then $(x.y)$ is undetectable and we remove $y$ from $N_x$ (and $x$ from $N_y$). The same is done if $x$ and $y$ satisfy the conditions of Corollary 7.6. If $(x.y)$ is an FBF with $y$ being the front line, and the conditions of Corollary 7.4 or Corollary 7.5 apply, then $(x.y)$ can be detected only based on $y$, and we remove $y$ from $N_x$.

Next we partition the remaining lines in $N_x'$ into two sets, $M_x$ and $M_x^*$. The set $M_x^*$ contains all lines $y$ that are successors of $x$, such that at least one path between $x$ and $y$ has odd inversion parity, and $M_x$ contains all other lines in $N_x'$. The reason for this partitioning is that if $y \in M_x$, detecting $x$ *s-a-*0 when $y$=0 is sufficient for the detection of $(x.y)$. But if $y \in M_x^*$, then $(x.y)$ is a potentially oscillating FBF that is detected based on $x$ only if the effect of $x$ *s-a-*0 does not propagate to $y$.

The processing required during SSF simulation to determine the detected BFs can be summarized as follows. After simulating each test $t$, analyze every fault $x$ *s-a-*0 detected by $t$. For every line $y \in M_x$ with value 0, mark $(x.y)$ as detected. For every line $y \in M_x^*$ with value 0, mark $(x.y)$ as detected if the effect of $x$ *s-a-*0 does not propagate to $y$.

Determining whether the effect of $x$ *s-a-*0 propagates to $y$ depends on the method used by SSF simulation. In deductive simulation we can simply check whether $x$ *s-a-*0 appears in the fault list of $y$. In concurrent simulation we must also check whether the values of $y$ in the good circuit and in the circuit with $x$ *s-a-*0 are different. In critical path tracing we have to determine whether $x$ is reached by backtracing critical paths from $y$.

A BF $(x.y)$ may be discarded as soon as it is detected. This means removing $y$ from $M_x$ or $M_x^*$ *and* removing $x$ from $M_y$ or $M_y^*$. A fault $x$ *s-a-*0, however, should be retained until all BFs that are detectable based on $x$ are detected, i.e., when $M_x$ and $M_x^*$ become empty. If equivalent SSFs have been collapsed, whenever a SSF fault is detected, all the *s-a-*0 faults equivalent to it should be checked. A SSF fault should be discarded only after all the BFs whose detection may be based on its equivalent *s-a-*0 faults have been detected.

Delaying fault discarding beyond first detection increases the cost of explicit fault simulation algorithms, such as deductive and concurrent. Therefore, the most suitable SSF simulation technique for determining the detected BFs is critical path tracing, which is an implicit fault simulation method that does not discard or collapse faults. Although the additional checks for fault-effect propagation are more expensive in critical path tracing than in the explicit fault simulation methods, the experimental results presented in [Abramovici and Menon 1985] show that in practice these checks are not needed, because most of the potentially oscillating FBFs are detected based on their front lines.

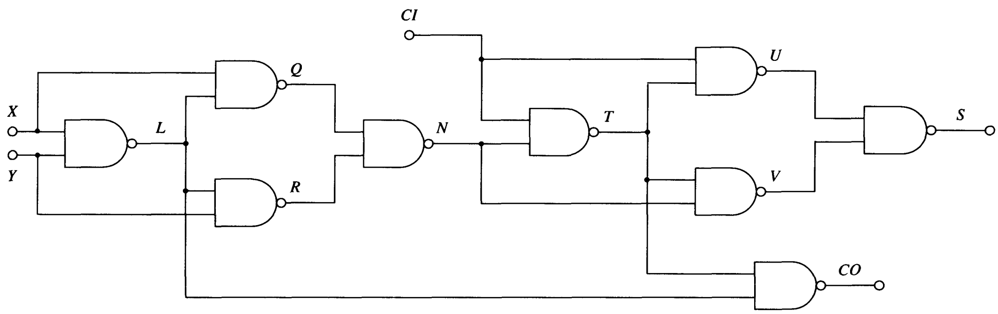



**Figure 7.11**

**Example 7.2:** Consider the adder in Figure 7.11. The rows of the array in Figure 7.12(a) show the sets $M$ and $M*$ of every line under the assumption that any two lines are neighbors. This is an unrealistic worst case, but it has the merit that it does not bias the example toward any particular layout. Neighbors in $M$ are denoted by "•" and those in $M^*$ by "*". For example, $M_R = \{X,Y,CI,L,T\}$ and $M_R^* = \{U,V,S,CO\}$. Note that $N$ does not appear in $M_R^*$ because all paths between $R$ and a PO pass through $N$. However, $R \in M_N$. Similarly, $Q$ is not included in $M_R$ because $(Q.R)$ is obviously undetectable, and $Q$, $R$, $T$, $V$, and $CO$ do not appear in $M_L^*$ because they are set to 1 whenever $L$=0. Figure 7.12(b) shows the applied tests (which detect all SSFs) and the values in the fault-free circuit.

The detection of BFs is summarized in Figure 7.12(c). A number $i$ in row $x$ and column $y$ shows that the SSF $x$ *s-a-*0 and the BF $(x.y)$ are detected by the test $t_i$. An "×" indicates that $(x.y)$ is detected based on $y$ (this test is found in row $y$ and column $x$). For example, $t_1$ detects $L$ *s-a-*0. Since $X,Y,CI$, and $N$ have value 0 in $t_1$, we write 1 in the corresponding entries. Since $S$ has value 0 and the effect of $L$ *s-a-*0 does not reach $S$, we also write 1 in the entry for $(L.S)$. We also enter an "×" in the entries for $(CI.L)$, $(N.L)$, and $(S.L)$.

| | $X$ | $Y$ | $CI$ | $L$ | $Q$ | $R$ | $N$ | $T$ | $U$ | $V$ | $S$ | $CO$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $X$ | | • | • | | | • | * | * | * | * | * | * |
| $Y$ | • | | • | | • | | * | * | * | * | * | * |
| $CI$ | • | • | | • | • | • | • | | | • | * | • |
| $L$ | • | • | • | | | | • | | • | | * | |
| $Q$ | • | • | • | • | | | | • | * | * | * | * |
| $R$ | • | • | • | • | | | | • | * | * | * | * |
| $N$ | • | • | • | • | • | • | | | • | | * | • |
| $T$ | • | • | • | • | • | • | • | | | | • | |
| $U$ | • | • | • | • | • | • | • | • | | | | • |
| $V$ | • | • | • | • | • | • | • | • | | | | • |
| $S$ | • | • | • | • | • | • | • | • | • | • | | • |
| $CO$ | • | • | • | • | • | • | • | • | • | • | • | |

(a)

| | $X$ | $Y$ | $CI$ | $L$ | $Q$ | $R$ | $N$ | $T$ | $U$ | $V$ | $S$ | $CO$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $t_1$ | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 |
| $t_2$ | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 |
| $t_3$ | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| $t_4$ | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| $t_5$ | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |

(b)

| | $X$ | $Y$ | $CI$ | $L$ | $Q$ | $R$ | $N$ | $T$ | $U$ | $V$ | $S$ | $CO$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $X$ | — | 2 | 2 | — | — | × | × | × | × | × | | 2 |
| $Y$ | × | — | 4 | — | × | — | × | × | × | × | × | |
| $CI$ | × | × | — | × | × | × | × | — | — | × | × | × |
| $L$ | 1 | 1 | 1 | — | — | — | 1 | — | × | — | 1 | — |
| $Q$ | 1 | 1 | 1 | 4 | — | — | — | × | | | × | 1 |
| $R$ | 1 | 1 | 1 | 4 | — | — | — | | × | × | | 2 |
| $N$ | 3 | 2 | 2 | × | 2 | 3 | — | — | × | — | × | 2 |
| $T$ | 1 | 1 | 1 | 5 | 2 | | 1 | — | — | — | 1 | — |
| $U$ | 1 | 1 | 1 | 4 | | 3 | 1 | 3 | — | — | — | 1 |
| $V$ | 1 | 1 | 1 | 4 | | 3 | 1 | 3 | — | — | — | 1 |
| $S$ | | 2 | 2 | × | 2 | | 5 | × | 5 | 2 | — | 2 |
| $CO$ | × | | 4 | 4 | × | × | × | 3 | × | × | × | — |

(c)

**Figure 7.12**

In this example 91 percent (58 out of 64) of the considered BFs are detected by single tests. The fault coverage is actually higher, because two of the six BFs declared as undetected — $(X.S)$ and $(Y.CO)$ — are in fact detected, respectively, by the sequences $(t_1,t_2)$ and $(t_2,t_3)$. □

The experimental results presented in [Abramovici and Menon 1985], involving simulation of up to 0.5 million BFs, show that

1. Tests with high coverage for SSFs (about 95 percent) detected, on the average, 83 percent of all possible BFs. The actual BF coverage may be higher, because BFs detected by sequences and potentially oscillating BFs detected based on back lines were not accounted for.

2. Although for most circuits, tests with high SSF coverage also achieved good BF coverage, this was not always the case. For example, a test set with 98 percent SSF coverage obtained only 51 percent BF coverage. Thus the BF coverage does not always follow the SSF coverage, and it must be separately determined.

3. On the average, 75 percent of the potentially oscillating BFs were detected based on their front lines. Thus, the pessimistic approximation introduced in the BF coverage by ignoring their detection based on back lines is negligible.

4. In large circuits, the additional time required for the implicit simulation of the feasible BFs is a small fraction of the time needed for SSF simulation (by critical path tracing).

## 7.5 Test Generation for Bridging Faults

The relations established in Sections 7.2 and 7.3 can also be used to extend a TG algorithm for SSFs so that it will generate tests for BFs as well.

Often when generating a test for a SSF, there are lines whose values are not constrained. We can take advantage of this freedom by setting values that contribute to the detection of BFs. Namely, for every fault $x$ *s-a*-0 detected by the partially specified test, we can try to find lines $y \in M_x$ such that $(x.y)$ is still undetected and the value of $y$ is unspecified. Then justifying $y$=0 creates the conditions for detecting $(x.y)$.

After the test generation for SSFs is complete, we may still have undetected BFs. Let $(x.y)$ be one of these BFs. If $y \in M_x$, we try to derive a test for $x$ *s-a*-0, with the added constraint $y$=0. (This, of course, should be attempted only if $x$ *s-a*-0 has been detected during SSF test generation.) Similarly, if $x \in M_y$, we try to derive a test that detects $y$ *s-a*-0 and sets $x$=0. These types of operations require only minor changes to a test generation program for SSFs.

When $y \in M_x^*$, the potentially oscillating FBF $(x.y)$ can be detected by trying to generate a test for $x$ *s-a*-0 while setting $y$=0, with the additional restriction that the effect of $x$ *s-a*-0 should not propagate to $y$. However, because most potentially oscillating FBFs can be detected based on their front lines, we can overlook the case $y \in M_x^*$ without any significant loss in the fault coverage.

**Example 7.3:** Let us continue Example 7.2 by generating a test for $(Q.U)$, which is one of the BFs not detected by the analyzed tests. Since $Q \in M_U$, we try to generate a test for *U* *s-a*-0 while setting *Q*=0. The unique solution to this problem is the test (*X*,*Y*,*CI*) = 101, which also detects the previously undetected BFs (*Q.V*) and (*Y.CO*). □

Although most FBFs can be detected by one vector, some FBFs require a sequence for their detection. Detection of FBFs by sequences has been studied by [Xu and Su 1985].

## 7.6 Concluding Remarks

The BF model is a "nonclassical" fault model. BFs cannot be "represented" by SSFs, because, in general, BFs are not equivalent to SSFs. Nevertheless, the methods presented in this chapter show that by analyzing the dominance (rather than equivalence) relations between these two types of faults, the processing of the two fault models can be combined, and BFs can be dealt with at a small additional cost over that required for testing SSFs.

## REFERENCES

[Abramovici and Menon 1985] M. Abramovici and P. R. Menon, "A Practical Approach to Fault Simulation and Test Generation for Bridging Faults," *IEEE Trans. on Computers*, Vol. C-34, No. 7, pp. 658-663, July, 1985.

[Friedman 1974] A. D. Friedman, "Diagnosis of Short-Circuit Faults in Combinational Logic Circuits," *IEEE Trans. on Computers*, Vol. C-23, No. 7, pp. 746-752, July, 1974.

[Malaiya 1986] Y. K. Malaiya, "A Detailed Examination of Bridging Faults," *Proc. Intn'l. Conf. on Computer Design*, pp. 78-81, 1986.

[Mei 1974] K. C. Y. Mei, "Bridging and Stuck-At Faults," *IEEE Trans. on Computers*, Vol. C-23, No. 7, pp. 720-727 July, 1974.

[Williams and Angell 1973] M. J. Y. Williams and J. B. Angel, "Enhancing Testability of Large-Scale Integrated Circuits via Test Points and Additional Logic," *IEEE Trans. on Computers*, Vol. C-22, No. 1, pp. 46-60, January, 1973.

[Xu and Su 1985] S. Xu and S. Y. H. Su, "Detecting I/O and Internal Feedback Bridging Faults," *IEEE Trans. on Computers*, Vol. C-34, No. 6, pp. 553-557, June, 1985.

## PROBLEMS

**7.1** For the circuit in Figure 7.13 determine the BFs detected by each of the tests 00, 01, 10, and 11.

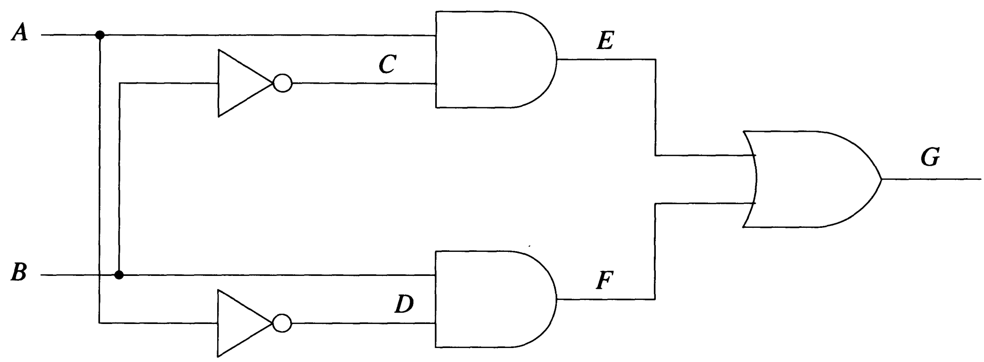



**Figure 7.13**

**7.2** Find a single test that detects the FBF $(a.z)$ of Example 7.1.

**7.3** For Example 7.2, determine the potentially oscillating BFs that are detected based on their front lines.

**7.4** For the circuit in Figure 7.14 show that the AND FBF $(A.Z)$ is not detectable by one test. Find a two-test sequence that detects $(A.Z)$.

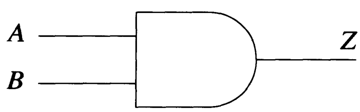



**Figure 7.14**