Improved cutting algorithm

by J. Savir

The cutting algorithm allows computation of bounds on signal probabilities and detection probabilities in combinational networks. These bounds can be used to determine the necessary pseudorandom test length needed to test a network. One of the problems with the cutting algorithm is that it may compute loose bounds which translate into unnecessarily long test lengths. The object of this paper is to improve the cutting algorithm so that the computed bounds become satisfactory. The improved cutting algorithm is a careful combination of the original cutting algorithm and the Parker–McCluskey algorithm. The tightness of the computed bounds may vary depending on which portion of the circuit is handled with the cutting algorithm and which with the Parker–McCluskey algorithm. Thus, the user of the improved cutting algorithm can actually control and trade off the accuracy of the results against the computational effort needed to achieve them.

Introduction

The continuous increase in chip density and digital system complexity has created an unprecedented crisis in testing. Traditional test generation and fault simulation have been proven to be too slow, too computationally intensive, and far from being a practical solution.

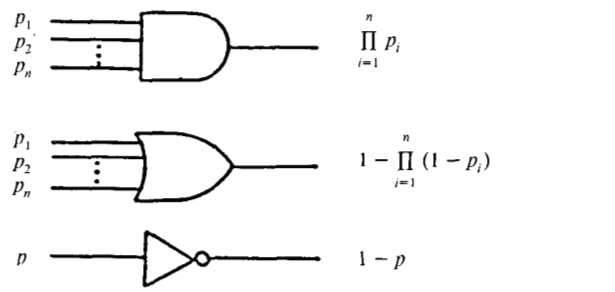
Built-in self-test (BIST) [1] has emerged as a possible solution to this crisis. Many BIST architectures have been proposed over the last ten years. These architectures differ in test methodology and in the amount of test hardware actually built into the product.

One of the most promising methodologies is based on pseudorandom patterns. As the name implies, this type of BIST applies pseudorandom patterns to the product until either the faults are exposed or the product is declared fault-free. The product test responses to the pseudorandom patterns are usually compressed in some form of a signature analyzer. Products either pass or fail the test depending on whether or not their measured signature matches that of the fault-free one. The fault-free signature is acquired either by simulation or by actual measurement of a network known to be good.

One of the problems associated with pseudorandom-pattern-based BIST is determining its quality. This quality is usually measured by the fault coverage (against single stuck-at faults) achievable by the pseudorandom test. If the actual patterns used during the test are known, this fault coverage can be determined by simulation. If the pseudorandom patterns are not known, or if simulation methods take too long, one can resort to a probabilistic evaluation of the fault coverage. The use of probabilistic measures leads to statistical determination of the fault coverage. Statements such as "a pseudorandom pattern test of length T will detect $P\%$ of the single stuck-at faults with $C\%$ confidence" are quite common.

Two of the methods used to statistically qualify the fault coverage are the cutting algorithm [1, 2] and the Parker–McCluskey algorithm [1, 3]. Both algorithms use signal probability computations to compute detection probabilities of faults. The *signal probability* of a line is defined as the probability that a randomly

Table 1 Rules of propagating signal probabilities according to the Parker–McCluskey algorithm.



$p^i = p$ for all $i > 1$ (exponent suppression)

selected pattern will propagate a value 1 to that line. The *detection probability* of a fault is the probability that a randomly selected pattern will expose the fault to at least one of the circuit outputs. To statistically determine the fault coverage of the pseudorandom test, it is necessary to obtain the *detection probability profile* of all faults at question. This profile is usually an ordered list of all the detection probabilities from largest to smallest.

The Parker–McCluskey algorithm manipulates symbolic expressions to compute signal and detection probabilities. This algorithm has the advantage of computing exact signal and detection probability figures, and the disadvantage of being too complex for large circuits. The cutting algorithm, on the other hand, computes bounds on the signal and detection probabilities rather than the exact values. The idea used in this algorithm (and the origin of its name) is to cut a significant number of fanout branches to turn the circuit into a tree from which the signal and detection probabilities can be easily computed. The penalty associated with these cuts is the abandoning of exact figures and, therefore, the acceptance of bounds. The disadvantage of the cutting algorithm, however, is its inability to deliver acceptable bounds in all instances. These loose bounds may sometimes lead to unacceptable test lengths.

The improved cutting algorithm described in this paper is a combination of the cutting algorithm and the Parker–McCluskey algorithm. It calls for cutting only a subset of the total fanout branches and the processing of the remaining fanout stems with symbolic expressions using the Parker–McCluskey algorithm. There is complete freedom as to which branches to cut and which to leave in. The improved cutting algorithm allows the penetration of symbolic expressions into bounds. The complexity of the algorithm and the accuracy of the results it produces depend upon the number of cuts made and the number of symbolic expressions manipulated by it. The general trend is that the algorithm becomes more complex, and therefore more accurate, as the number of symbolic expressions increases, with an opposite trend appearing as the number of symbolic expressions decreases. Thus, the user of the improved cutting algorithm can actually decide for himself how much extra complexity is warranted by the added accuracy.

The paper is divided into several sections. Beginning with a brief description of the Parker–McCluskey and cutting algorithms, we then describe the improved cutting algorithm. Most of the paper is devoted to the computation of signal probabilities, with one section describing detection probabilities that are derivable from signal probabilities.

The improved cutting algorithm discussed in this paper is based upon the *full-range* cutting algorithm [1, 2]. Everything stated and described in this paper also holds true for the *partial-range* cutting algorithm [1, 2]. The pseudorandom test discussed here is assumed to be *unbiased*; i.e., all input vectors are assumed to be equally likely. The biased case can be treated similarly by adjusting the signal probabilities of the primary inputs [1]. The discussion in this paper is restricted to combinational circuits (or level-sensitive scan designs, LSSD) [1, 4].

It is worthwhile to mention that there have been other probabilistic algorithms aimed at evaluating the quality of BIST designs. Some of these are described in [1].

Parker–McCluskey algorithm

The Parker–McCluskey (PM) algorithm can compute the *exact* signal probability of a line. The PM algorithm involves manipulation of symbolic expressions, as described in the following procesure.

- *PM procedure*

Step 1: Identify the stems of the reconvergent fanout branches and mark them $p_1, p_2, \cdots, p_k$.

Step 2: Assign a signal probability of 1/2 to the primary inputs and compute the signal probabilities of the lines downstream using the propagation rules of **Table 1**. When a fanout stem is reached, record its value (or expression if it is a function of other p_i symbols upstream). Continue to propagate signal probabilities according to Table 1 until all the stems marked in Step 1 have been evaluated.

Step 3: Using the symbols $p_1, p_2, \cdots, p_k$ and the other primary inputs, compute the signal probabilities in the circuit using the propagation rules of Table 1.

Step 4: Substitute the values of $p_1, p_2, \cdots, p_k$ into the signal probability expressions of the lines. Perform exponent suppression whenever necessary as described in Table 1. The numeric values thus obtained constitute the exact signal probabilities of the lines in the circuit.

We illustrate the PM algorithm with an example.

Example 1

Consider the circuit of **Figure 1**. We use the PM algorithm to compute the signal probability of the output F.

There are two stems in this circuit; the signal probabilities of these lines are marked p_1 and p_2 in Figure 1. Since p_1 is the signal probability of a primary input, $p_1 = 1/2$. The value of the signal probability p_2 can be computed from Table 1: $p_2 = [1 - (1/2)]^2 = 1/4$. The rest of the signal probability expressions are as follows:

$$E_1 = E_2 = [1 - (1/2)](1 - p_1) = (1/2)(1 - p_1),$$

$$E_3 = 1 - [1 - (1/2)](1 - p_2) = (1/2)(1 + p_2),$$

$$E_4 = 1 - p_2 E_1 E_2 = 1 - p_2[(1/2)(1 - p_1)]^2$$
$$= 1 - (1/4)p_2(1 - 2p_1 + p_1^2).$$

After suppressing the second power of p_1, we get

$$E_4' = 1 - (1/4)p_2(1 - p_1),$$

$$E_5 = E_3 E_4' = (1/2)(1 + p_2)[1 - (1/4)p_2(1 - p_1)]$$
$$= (1/2)[1 + p_2 - (1/4)(p_2 + p_2^2)(1 - p_1)].$$

After suppressing the second power of p_2, we get

$$E_5' = (1/2)[1 + (1/2)p_2 + (1/2)p_1 p_2].$$

Substituting $p_1 = 1/2$ and $p_2 = 1/4$ in the expression of E_5' yields the signal probability of the output F, $SP(F) = 19/32$. □

As stated earlier, the PM algorithm computes exact signal probability values. In the worst case, however, its complexity is exponential, which hinders its use in large circuits.

Cutting algorithm

The cutting algorithm (CA) was devised to alleviate the complexity problem of the PM algorithm. The idea was to turn the combinational circuit into a tree by "cutting" some of the reconvergent fanout branches. Since signal probability computation is very simple in tree networks (linear complexity), the "cut" version of the circuit can be analyzed very easily. Notice that no structural changes are made to the circuit. The so-called "cuts" are software modifications to the circuit model that are used solely for the purpose of analyzing the circuit. But when the circuit is converted into a tree, an analysis penalty must be paid. In general, the CA does not compute exact signal probability values as the PM algorithm does. In the context of the following algorithm, a *tree line* is a line whose cone of logic is free of reconvergent gates (a reconvergent gate is a gate at which reconvergent fanout branches converge). A *nontree line* is a line whose cone of logic includes at least one reconvergent gate. The CA computes exact signal probability values for tree lines and signal probability bounds for nontree lines. A more detailed description of the CA follows.

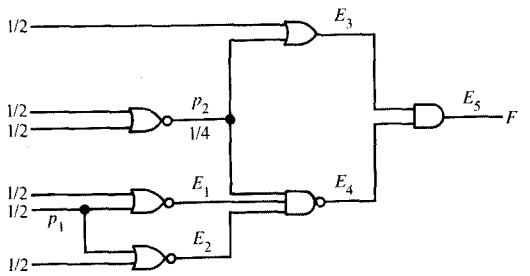


Figure 1

Example circuit.

• *CA procedure*

To compute the signal probability of a tree line, follow the PM algorithm. To compute the signal probability bound of a nontree line, do the following:

Step 1: Identify the reconvergent fanout branches in the circuit.

Step 2: Cut enough reconvergent fanout branches to turn the circuit into a tree. Assign the signal probability of 1/2 to the primary inputs and signal probability range [0, 1] to each of the cut branches.

Step 3: Using the CA propagation rules of signal probability bounds (**Table 2**), compute the signal probability bounds of the nontree lines.

The following example illustrates the use of the CA.

Example 2

We compute the signal probability bound of the output F of Figure 1. **Figure 2** is a cut version of Figure 1, in which two reconvergent branches have been cut in order to turn the circuit into a tree. These cut branches have

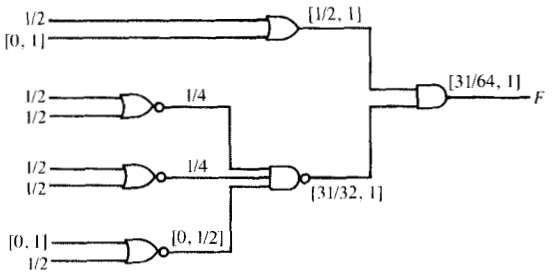


Figure 2

Cut version of the example circuit of Figure 1.

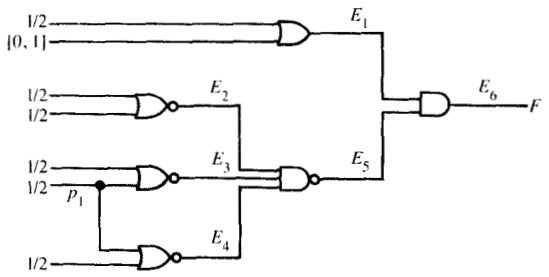


Figure 3

Example circuit of Figure 1 with one fanout branch cut.

Table 2 Cutting algorithm rules for propagating signal probability bounds.

$[l_1, u_1]$ $[l_2, u_2]$ $[l_n, u_n]$	$\left[\prod_{i=1}^{n} l_i, \prod_{i=1}^{n} u_i\right]$
$[l_1, u_1]$ $[l_2, u_2]$ $[l_n, u_n]$	$\left[1 - \prod_{i=1}^{n} (1 - l_i), 1 - \prod_{i=1}^{n} (1 - u_i)\right]$
$[l, u]$	$[1 - u, 1 - l]$

been assigned a [0, 1] bound. The propagation of signal probability values and bounds is shown in Figure 2. As is evident, the signal probability bound of the output F has been computed by the CA to be [31/64, 1], which in fact encloses the exact value, 19/32, computed earlier by the PM algorithm. □

Although the CA has a relatively low complexity (polynomial of low degree), it sometimes fails to deliver acceptable bounds. For example, the best signal probability bound the CA can compute for an Exclusive-Or gate implemented by four NAND gates (see Figure 4, shown later) is the [0, 1] bound.

Improved cutting algorithm

The improved cutting algorithm (ICA) is aimed at combining the advantages of the CA and PM algorithms. It takes advantage of both the simplicity of the CA algorithm and the accuracy of the PM algorithm. The ICA will still, in general, compute bounds of signal probabilities, but these bounds will be substantially narrower than those computed by the CA.

In the ICA combination of CA and PM algorithms, some of the reconvergent fanout branches are cut according to the CA algorithm, and some fanout stems are processed symbolically according to the PM algorithm. The user must decide which portion of the circuit will be cut and which will be processed symbolically; he can therefore custom-tailor the ICA according to his accuracy/complexity trade-off objectives.

• *ICA procedure*

Step 1: Determine which fanout branches should be cut and which fanout stems should be processed symbolically.

Step 2: (Bound consolidation rule) Propagate signal probability expressions and bounds following both the CA and PM algorithms. Whenever a bound of bounds of the form $[[L_1, U_1], [L_2, U_2]]$ is encountered, replace it with the bound $[L_1, U_2]$.

We illustrate the use of the ICA with some examples.

Example 3

Consider the circuit of **Figure 3**, which has been derived from the example circuit of Figure 1 by cutting one of the fanout branches.

As can be seen in Figure 3, only one symbol (p_1) is used to compute the signal probability bound of the output F. The internal expressions and bounds are as follows:

$$p_1 = 1/2, \quad E_1 = [1/2, 1],$$

$$E_2 = 1/4, \quad E_3 = E_4 = (1/2)(1 - p_1),$$

$$E_5 = 1 - E_2E_3E_4 = 1 - (1/4)[(1/2)(1 - p_1)]^2.$$

After suppressing the second power of p_1 in E_5, we get

$E_5' = (1/16)(15 + p_1)$.

The output signal probability bound expression is

$E_6 = E_1 E_5' = (1/16)(15 + p_1)[1/2, 1]$.

After substituting $p_1 = 1/2$ in the expression of E_6, we get the signal probability bound of the output F, $SPB(F) = [31/64, 31/32]$. Notice that this bound is narrower than the one computed by the CA. The reader can verify that if the p_1 stem is cut, rather than p_2 (see Figure 1), the output signal probability bound achieved is even tighter ($[19/32, 5/8]$). □

As can be seen in the following example, the symbols $p_1, p_2, \cdots, p_k$ can themselves be bounds. This is the case in which the bound consolidation rule of Step 2 of the ICA procedure must be invoked.

Example 4

Consider the Exclusive-Or implementation of **Figure 4**. **Figure 5** shows the circuit of Figure 4 with two fanout branches cut.

The symbol p assigned to the fanout stem is now a bound, $p = [1/2, 1]$. Following the steps of the ICA, we get

$E_1 = 1 - (1/2)p, \quad E_2 = [1 - p, 1],$

$E_3 = 1 - E_1 E_2 = 1 - [1 - (1/2)p][1 - p, 1]$

$= [(1/2)p, p]$ (after exponent suppression).

Substituting the value of p in E_3, we get a bound of bounds,

$E_3 = [(1/2)[(1/2), 1], [(1/2), 1]]$.

The resolution of this bound of bounds leads to the signal probability bound of the output F, $SPB = [1/4, 1]$ (the exact value is 1/2).

Note that other cutting patterns are possible in the circuit of Figure 4. Each cutting pattern may result in a different signal probability bound for the output F. The bound achieved by the cutting pattern of Figure 5 is not necessarily the best one possible. □

As shown in Example 4, the symbols $p_1, p_2, \cdots, p_k$ are used as if they were simple numeric parameters. Whenever a signal probability bound expression must be numerically evaluated, the bound is compressed according to the bound consolidation rule (Step 2 of the ICA procedure).

The next example is concerned with multiple-symbol use of the ICA.

Example 5

Consider the circuit of **Figure 6**. **Figure 7** is a cut version of the circuit shown in Figure 6 (two branches cut).

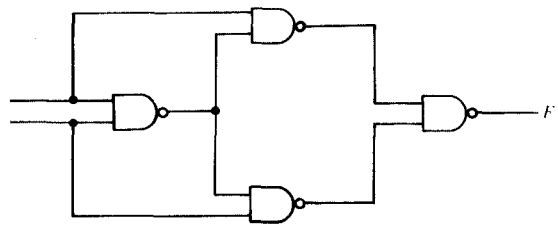


Figure 4

Exclusive-Or implementation.

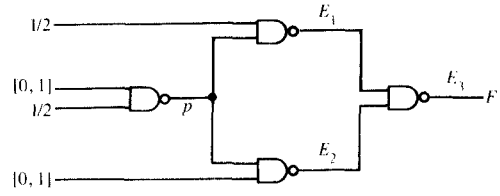


Figure 5

Cut version of the circuit shown in Figure 4.

Three symbols are used in Figure 7 to compute the signal probability bound of the output F.

The application of the ICA procedure yields

$p_1 = [1/2, 1], \quad p_2 = 3/4, \quad p_3 = [0, 1/2],$

$E_1 = 1 - (1 - p_1)(1 - p_3), \quad E_2 = p_1 p_2,$

$E_3 = (1 - p_2)(1 - p_3),$

$E_4 = 1 - (1 - E_1)(1 - E_2)(1 - E_3)$

$= 1 - (1 - p_1)(1 - p_3)(1 - p_1 p_2)(p_2 + p_3 - p_2 p_3).$

Suppressing high-order exponents in E_4 results in

$E_4' = 1 - (1 - p_1)p_2(1 - p_3).$

The signal probability bound of the output F is

$SPB(F) = 1 - [0, 1/2](3/4)[1/2, 1]$

$= 1 - [0, 3/8][1/2, 1]$

$= 1 - [0, 3/8] = [5/8, 1].$

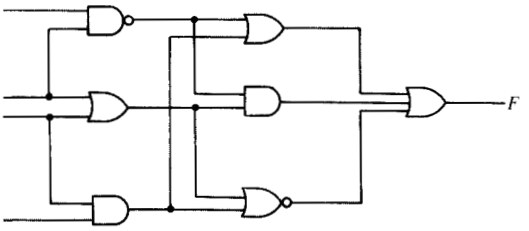


Figure 6

Circuit of Example 5.

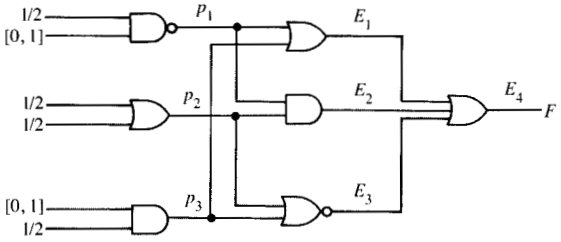


Figure 7

Cut version of the circuit shown in Figure 6.

The reader can verify that this bound encloses the exact value of 13/16. It is also important to note that different cutting patterns of Figure 6 will lead to different signal probability bounds, and the bound computed according to the cutting pattern of Figure 7 is not necessarily the best one possible. □

Computing lower bounds of detection probabilities

As is shown in [1, 2], it is possible to turn the detection probability computation problem into a signal probability problem. One such method adds *auxiliary gates* to the circuit model, so that the lower bounds of the signal probabilities computed at the outputs of those auxiliary gates constitute lower bounds on the detection probabilities at question. The auxiliary gates are usually And-gates, which are fed from a number of circuit lines in such a way that the output of the And-gate is 1 if and only if a fault is detected by propagating its effect along a preselected path to one of the primary outputs.

The addition of auxiliary gates to the circuit model allows computation of lower bounds on the detection probabilities of faults. Using these lower bounds, instead of the exact detection probability of faults, to compute the random-pattern test length needed to detect all faults with a certain degree of confidence, results in conservative test length figures [1, 5].

Since the CA sometimes delivers a lower bound of 0 on detection probabilities of faults, it may cause this conservative test length figure to be unacceptable. The ICA, on the other hand, can deliver nonzero lower bounds at the expense of a higher (but still reasonable) complexity cost. The next example is aimed at clarifying these points.

Example 6

Consider the circuit of **Figure 8(a)**. We are interested in computing a lower bound on the detection probability of the fault w stuck-at-0 ($w/0$). We choose to propagate the effect of $w/0$ along the boldface path shown in Figure 8(a). **Figure 8(b)** shows the circuit of Figure 8(a) with the auxiliary gate (AG) added. The connections to AG are made so that its output $H = 1$ if and only if the fault $w/0$ is detected by propagating its effect along the boldface path. For that to happen, gates G_2 and G_4 must be in a sensitized state, and gate G_3 in a blocked state. Thus, points A, B, and D are connected directly to the inputs of the AG, and point C is connected with inverted polarity to the AG.

Figure 8(c) shows the auxiliary gate with its cone of logic. The gates which do not influence the output H have been removed. We use Figure 8(c) to show that the CA cannot compute a nonzero lower bound for the detection probability of $w/0$ along the selected boldface path. Note that the lower bound of the signal probability of the output H constitutes a lower bound on the detection probability of $w/0$.

To use the CA to compute a signal probability bound of the output H, it is necessary to cut enough fanout branches in Figure 8(c) to turn it into a tree. To accomplish this, two fanout branches must be cut. There are four possible cutting patterns: {ac, ad, bc, bd}. Note that if any pair from {ac, ad, bc} is chosen for cutting, an immediate [0, 1] signal probability bound appears at one of the inputs of AG, resulting in a 0 lower bound for the detection probability in question. Thus, there is only one more possibility to consider, cutting branches b and d. However, if branches b and d are cut, line e receives a signal probability bound of [0, 1], causing the lower bound of the detection probability in question to be zero again. The CA, therefore, cannot compute a nonzero lower bound for the detection probability of $w/0$.

Figure 9 shows the circuit of Figure 8(c) with only fanout branch d cut. We next apply the ICA to the circuit of Figure 9:

$p = 1/2, \quad E_1 = 3/4, \quad E_2 = [1 - p, 1],$

$E_3 = (1/2)(1 - p)E_1E_2 = (1/2)(1 - p)(3/4)[1 - p, 1],$

$E_3 = (3/8)[(1 - p)^2, 1 - p].$

Suppressing high-order exponents in E_3 results in

$E_3' = (3/8)[1 - p, 1 - p].$

Thus, the lower bound of the detection probability of $w/0$ is

$LB(w/0) = (3/8)(1 - p) = (3/8)(1/2) = 3/16.$

The reader can verify that the exact detection probability of $w/0$ along the boldface path of Figure 8(a) is 3/16. The unrestricted (in the sense that no limitation is imposed on propagation paths) detection probability of $w/0$ is 9/16. □

Conclusions

The improved cutting algorithm is a combination of the original cutting algorithm and the Parker–McCluskey algorithm. The accuracy of the results delivered by the improved cutting algorithm may range from loose bounds to exact figures. The complexity of the improved cutting algorithm may range from quadratic to exponential.

Both accuracy and complexity are controlled by the user of the algorithm. The user has complete freedom with respect to analyzing his design: He can use very few cuts, in which case the results are more accurate, or he may use more cuts, in which case the algorithm is less complex. The user always has the option of increasing the accuracy of the results by running the algorithm again with fewer cuts. The accuracy of the results can thus be dynamically adjusted.

Experience with the improved cutting algorithm suggests that even processing of very few symbols greatly enhances the accuracy of the results, in which case the complexity of the improved cutting algorithm is very close to that of the original cutting algorithm.

If a given choice of cuts results in unsatisfactory bounds, the user may rerun the procedure to achieve better results. The improved cutting algorithm is amenable to incremental analysis. If the rerun requires only an incremental change in the set of branches that need be cut, only incremental work is needed to compute the effect of these incremental changes. More specifically, only bounds on lines reachable from the changed branches must be recomputed by the procedure. Expressions and bounds computed for other lines remain the same after the change. This property is important because the cost of improvements is usually less than the cost of the original job.

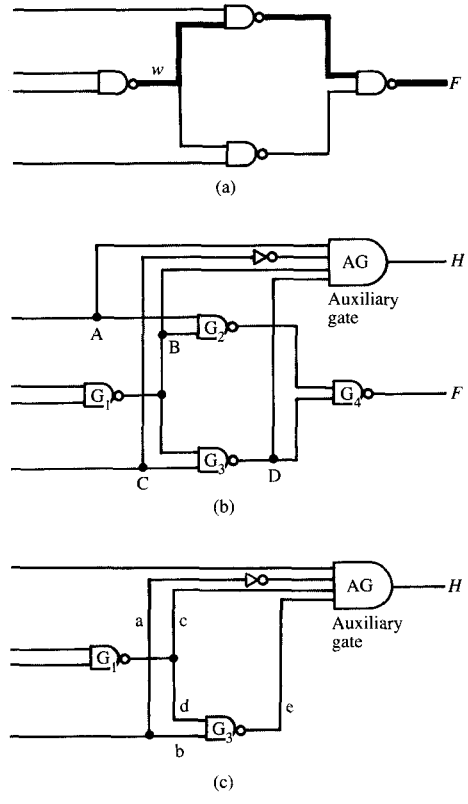


Figure 8

(a) Circuit of Example 6. (b) Circuit with auxiliary gate to detect $w/0$. (c) Cone of logic of the auxiliary gate.

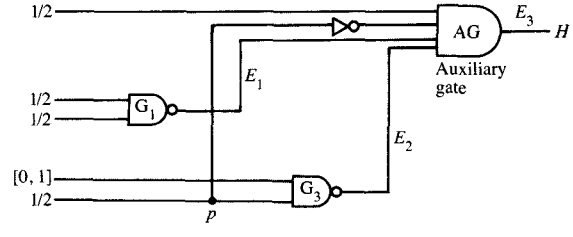


Figure 9

Circuit of Figure 8(c) with one cut branch.

References

1. P. H. Bardell, W. H. McAnney, and J. Savir, *Built-In Test for VLSI: Pseudorandom Techniques*, John Wiley & Sons, Inc., New York, 1987.
2. J. Savir, G. S. Ditlow, and P. H. Bardell, "Random Pattern Testability," *IEEE Trans. Computers* **C-33,** 79–90 (1984).
3. K. P. Parker and E. J. McCluskey, "Probabilistic Treatment of General Combinational Networks," *IEEE Trans. Computers* **C-24,** 668–670 (1975).
4. E. B. Eichelberger and T. W. Williams, "A Logic Design Structure for LSI Testability," *J. Design Automat. Fault-Tol. Comput.* **2,** 165–178 (1978).
5. J. Savir and P. H. Bardell, "On Random Pattern Test Length," *IEEE Trans. Computers* **C-33,** 467–474 (1984).

Received January 6, 1989; accepted for publication September 27, 1989

Jacob Savir *IBM Data Systems Division, P.O. Box 950, Poughkeepsie, New York 12602.*

Dr. Savir holds a B.Sc. and an M.Sc. degree in electrical engineering from the Technion, Israel Institute of Technology, and an M.S. in statistics and a Ph.D. in electrical engineering from Stanford University. He is currently a Senior Engineer in the IBM Data Systems Division in Poughkeepsie, and was previously a Research Staff Member at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York. He is also an Adjunct Professor of Computer and Information Systems at Pace University, New York. Dr. Savir's research interests lie primarily in the testing field, where he has published numerous papers and coauthored the text *Built-In Test for VLSI: Pseudorandom Techniques* (Wiley, 1987). Other research interests include design automation, design verification, design for testability, statistical methods in design and test, fault simulation, fault diagnosis, and manufacturing quality. Dr. Savir has received two IBM Invention Achievement Awards. He is a member of the Institute of Electrical and Electronics Engineers and Sigma Xi.