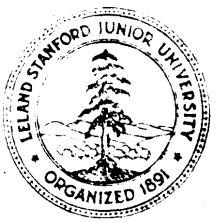
COMPUTER SYSTEMS LABORATORY

STANFORD ELECTRONICS LABORATORIES
DEPARTMENT OF ELECTRICAL ENGINEERING
STANFORD UNIVERSITY · STANFORD, CA 94305



SYNDROME-TESTABLE DESIGN OF COMBINATIONAL CIRCUITS

Jacob Savir

Technical Report No. 157

October 1978

This work was supported by an IBM post-doctoral fellowship.

SYNDROME-TESTABLE DESIGN OF COMBINATIONAL CIRCUITS

Jacob Savir

Technical Report No. 157

October 1978

Center for Reliable Computing
Computer System Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, California

This work was supported by an IBM post-doctoral fellowship

SYNDROME-TESTABLE DESIGN OF COMBINATIONAL CIRCUITS

Jacob Savir

Technical Report No. 157

October 1978

Center for Reliable Computing
Computer Systems Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, California

AESTRACT

Classical testing of combinational circuits requires a list of the fault-free response of the circuit to the test set. For most practical circuits implemented today the large storage requirement for such a list makes such a test procedure very expensive.

In this paper we describe a method of designing combinational circuits in such a way that their test procedure will require the knowledge of only one characteristic of the fault-free circuit, called the syndrome. This solves the storage problem asscciated with the test procedure. It is shown that the syndrome-testable design is inexpensive and can be easily implemented by the logic designer.

INDEX TERMS:

Combinational circuit; Stuck-at fault; Minterm; Prime implicant; Fanout-free circuit.

1. INTRODUCTION

In this paper we restrict ourselves to the class of permanent stuck-at-faults, i.e. a faulty line appears logically as if it is either stuck at a constant 0, or at a constant 1. The classical approach [6] to the problem of testing combinational circuits was to design a test set which detects all fauits from a prescribed set, and store both the test set and the expectea output to the individual test vectors . Whenever a circuit was tested , the actual response and the expected output were compared to determine whether or not the circuit under test (CUT) was fault-free. it was already conjectured in the past [2] that for most of the possible combinational circuits, the minimal test length for single faults is very large, for instance 2^{n-1} for two level circuits where n is the number of binary input lines. Thus, it seems that not much can be done in general to reduce the test length. However, it is possible to reduce the storage requirement considerably at a very low cost.

Recently, a few other approaches to the testing problem have been presented in the literature. In [3] the method of transition counts is described. This method has the advantage of drastically reducing the storage requirement, but it is difficult to determine a proper test input sequence. In [5] a referenceless testing method is described. Its disadvantage lies in requiring very long test patterns to achieve a reasonable detection probability. Because the test generation is probabilistic , all realistic test lengths are considerably larger than the number of all possible input combinations.

In this paper we show a method of designing combinational circuits such that the storage requirement wili be restricted to only one number , no matter how large the circuit is. This characteristic number , which wili be called the syndrome of the circuit, is based on the number of minterms realized by the switching function. Since a fault-free and faulty circuit do not necessarily have different syndromes, special design of the combinational circuit is required in order to make it syndrome-testable. We say that a circuit is syndrome-testable if the syndrome of any faulty version of a circuit does not equal the syndrome of the fault-free circuit. Thus, the test procedure consists of applying all input combinations 50 the CUT and recording its syndrome (usually implemented by a counter) . if the actual syndrome equais the expected syndrome, the circuit is fauit-free ; otherwise a fault i s detected and the procedure stops. This approach has both the advantage of requiring only one reference and the advantage of having a test length which does not exceed the maximum number of input combinations. In order to reduce the test length for circuits with large number of inputs (for instance more than 20), the combinational circuit i s partitioned into subcircuits and designed so that each subcircuit is syndrome-testable. In this way, the testing time for any size combinational circuit will never exceed 1 second.

The penalty paid for producing a syndrome-testable design is a slight increase in the number of pins. It is shown that for most practical circuits, the number of extra pins is no more than 1.

In section 2 various syndrome properties of combinational circuits are described. Section 3 describes the proposed testable design. The paper concludes with a brief summary.

2. SYNDROME-PRGPERTIES OFCOMBINATIONALCIRCUITS

Definition 1: The syndrome of a Eooiean function is defined as $S = \frac{K}{2^n}$, where K is the number of minterms realized by the function and n the number of binary input iines.

The syndrome is a functional property. Thus, various realizations of the same function wiii have the same syndrome. Eowever, since different realizations have different testing properties, it is possible to find syndrome-testable realizations. Clearly, $0 \leq S \leq 1$, where the boundaries are attained by the constant functions.

The syndrome cf various n-input gates is shown in Fig. 1.

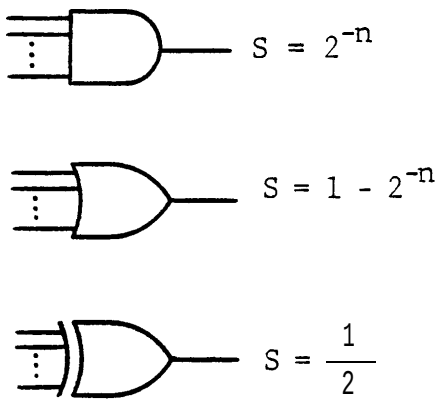


FIGURE 1: Syndrome of various n-input gates.

It is useful to find the input-output syndrome relations between various interconnected sections of a logic circuit. These input-cutput relations are especially simple when the associated sections have unshared (disjoint) inputs.

Lemma 1: When the inputs are unshared, the input-output syndrome reiations of the networks terminating in an INVERTER, m-gate, AND-gate, or EXCLUSIVE-OH gate are given in Fig. 2.

Proof: We will prove the input-output syndrome relation of the network terminating in an OR gate. The other proofs are similar.

Denote by (n, K, S) the three-tuple describing the number of inputs, the number of minterms, and the syndrome realized by a given logic block. Let (n_1, K_1, S_1) and (n_2, K_2, S_2) be the three-tuples describing the two interconnected blocks in Fig. 2(b).

Since the network is terminating in an OR gate, its output is 1 whenever any one of its inputs is 1. Thus, the number of minterms, K, realized by the network is

$$K = K_1 2^{n_2} + K_2 2^{n_1} - K_1 K_2$$

Hence,

$$S = \frac{K}{2^{n_1+n_2}} = \frac{K_1}{2^{n_1}} + \frac{K_2}{2^{n_2}} - \frac{K_2}{2^{n_1}} \cdot \frac{K_2}{2^{n_2}}$$

$$S = S_1 + S_2 - S_1 S_2$$

Q.E.D.

Lemma 1 is useful when determining the syndrome of a fanout-free combinational circuit. It also provides an algebraic tool to find the number of minterms realized by a fanout-free network. Traditionally, one had to use mapping tools or equivalent methods to find the number of minterms realized by a given function, which were quite cumbersome.

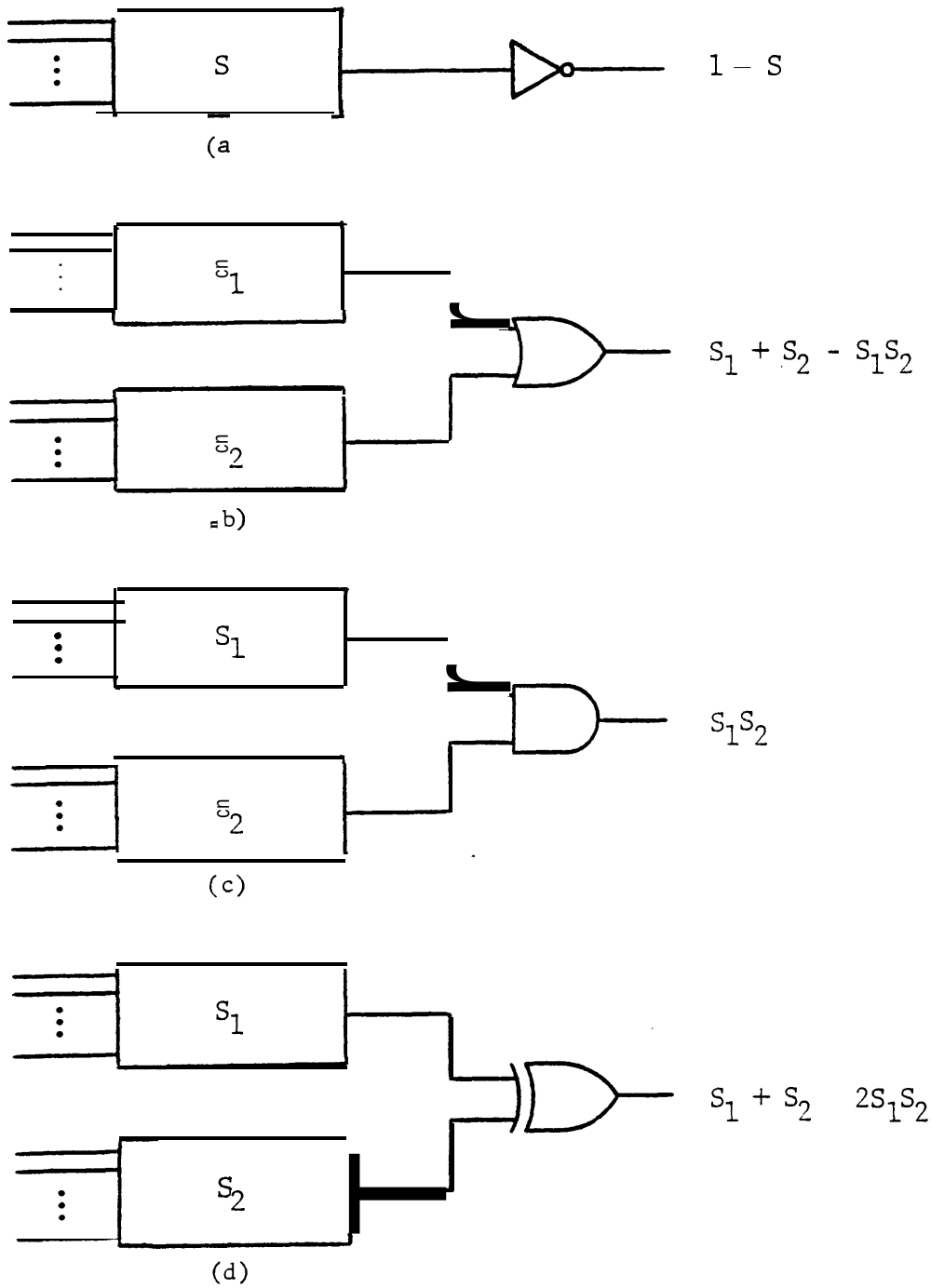


FIGURE 2: Input-output syndrome relations.

Example 1 : Find the syndrome and number of minterms realized by the fanout-free circuit of Fig.3 without using a Karnaugh map or equivalent-normal-form.

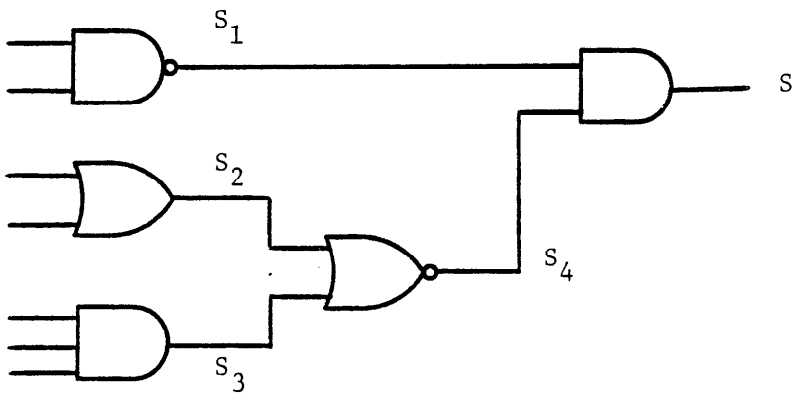


Figure 3: Example 1.

be have

$$S_1 = 1 - 2^{-2} = \frac{3}{4} \text{ , } S_2 = 1 - 2^{-2} = \frac{3}{4} \text{ , } S_3 = 2^{-3} = \frac{1}{\quad} \text{ .}$$

Hence,

$$S_4 = 1 - (S_2 + S_3 - S_2S_3) = \frac{7}{32} \text{ , } S = S_1S_4 = \frac{21}{128} \text{ ,}$$

$$K = S \cdot 2^n = 21 \text{ .}$$

Let K(F) and S(F) denote the number of minterms and the syndrome of a switching function F, respectively. In the case of interconnected blocks with shared (conjoint) inputs, the foliowing Lemma is very useful.

Lemma 2: Let two blocks be interconnected as shown in Fig. 4. Let the Boolean functions realized by blocks B_1 and B_2 be F and 2 respectively. Then

$$S(F + G) = S(F) + S(G) - S(FG) \qquad (1)$$

Proof: Consider the network of Fig. 4. Let the number of inputs to the blocks realizing the functions F and G be n_1 and n_2 respectively. Also, let, p be the number of inputs which are common to both blocks.

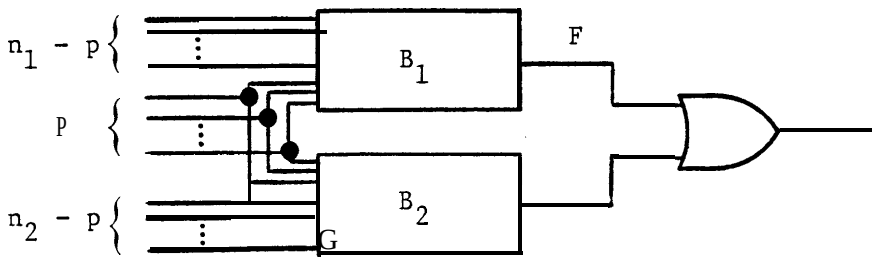


FIGURE 4: Two blocks having shared inputs.

Thus,

$$K(F + G) = K(F)\cdot 2^{n_2-p} + K(G)\cdot 2^{n_1-p} - K(FG) \ ,$$

$$S(F + G) = \frac{K(F + G)}{2^{n_1+n_2-p}} = \frac{K(F)}{2^{n_1}} + \frac{K(G)}{2^{n_2}} - \frac{K(FG)}{2^{n_1+n_2-p}}$$

$$S(F + G) = S(F) + S(G) - S(FG) \ .$$

Q.E.D.

Note, that Lemma 2 reduces to Lemma 1 in the disjoint case.

From Lemma 2 the following relations are implied:

$$S(FG) = 1 - S(\overline{FG}) = S(F) + S(G) + S(\bar{F}\bar{G}) - 1 \qquad (2)$$

$$S(F \oplus G) = S(F\bar{G}) + S(\bar{F}G) \qquad (3)$$

Example 2: Find the syndrome of the function

$$F = \bar{x}\bar{y}z + wxz + xy + \bar{v}\bar{w}y\bar{z}$$

Solution:

Let $A=\bar{x}\bar{y}z$, $B=wxz$, $C=xy$, $D=\bar{v}\bar{w}y\bar{z}$

Note that $S(x_1^* x_2^* \ldots x_n^*) = 2^{-n}$, where $x_i^* \varepsilon \{x_i, \bar{x}_i\}$, $i=1,2,\ldots,n$

Hence

$$S(A) = \frac{1}{8}, \quad S(E) = \frac{1}{8}, \quad S(C) = \frac{1}{4}, \quad S(D) = \frac{1}{16}$$

Using (1) we get

$$S(A + B) = \frac{1}{4}, \quad S(A + E + D) = \frac{5}{16}$$

$$S(F) = S(C) + S(A + B + D) - S(AC + EC + CD)$$

$$= \frac{9}{16} - S(wxyz + \bar{v}\bar{w}xy\bar{z}) = \frac{9}{16} - S(wxyz) - S(\bar{v}\bar{w}xy\bar{z})$$

$$= \frac{9}{16} - \frac{1}{16} - \frac{1}{32} = \frac{15}{32}$$

3. THE TESTABLE DESIGN

We are given a switching function to be realized with logic gates. Our purpose is to reach a syncrome-testable realization, namely, to have a design such that no fault can cause the circuit to have the same syndrome as the fauit-free circuit. he would like to achieve this goal while inserting the minimum number of extra inputs.

The class of faults we consider in this paper is the single stuck-at type. However, many multiple faults will be covered by the proposed design. In subsection 3.1 we consider two-level circuits which are the most simpie to design. Here , whenever we refer to a two-level circuit we implicitly impiy an AND-OR circuit. The results can be easily extended to CR-AND circuits. As a consequence, our design will be applicable to the important class of Programmable Logic Arrays (PLA's). In subsection 3.2 we treat the class of general combinational circuits.

The test procedure of the syndrome-testabie circuits is shown in Fig. 5. Each possible input combination is applied to the CUT exactly once. The syndrome-register is a counter which counts the

number of ones appearing on the output of the CUT, The equality checker checks the register's contents with the expected syndrome. If the syndromes are equal , the CUT is reported to be fault-free; otherwise a fault is detected and the CUT is declared faulty.

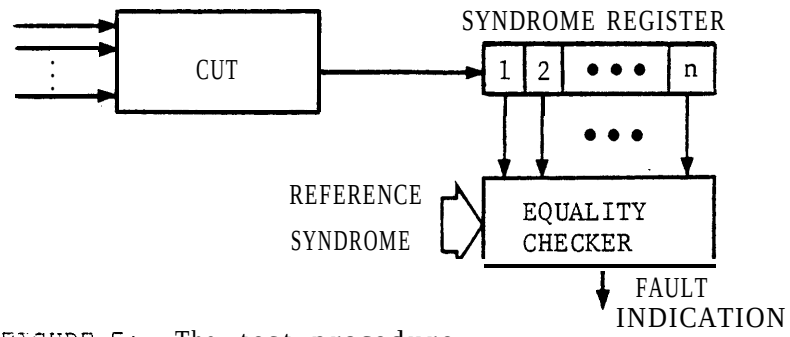


FIGURE 5: The test procedure.

3.1 Syndrome-Testable Design of' Twc-Level Circuits.

Definition 2: The function $F(x_1, x_2, .., x_i, .., x_n)$ is said to be positive (negative) in the variable x_i, if' there exists a disjunctive or conjunctive expression for it in which x_i appears only in uncompiemented (complemented) form.

Definition 3: The function $F(x_1, x_2, ..., x_i, ..., x_n)$ is said to be unate in x_i, if it is either positive or negative in x_i.

Lemma.3: A two-level irredundant circuit which realizes a unate function in all its variables is syndrome-testable.

Proof It is sufficient to consider the set of stuck-at faults at the inputs to the AND gates and the fanout branches. Any stuck-at 0 fault at the input to an AND gate causes the prime implicant (PI) realized by the AND gate to vanish. Any stuck-at 1 fault at the

input of an AND gate corresponds to a growth term which covers the original PI and its adjacent neighbor. Since we assume that the circuit is irredundant both faults change the number of minterms realized by the function , and thus change its syndrome.

In order to complete the proof we have to consider the influence of stuck-at faults at the fanout branches on the syndrome. Assume without loss of generality that the function is positive in x_i, and that x_i is a fanout branch. Thus, the function F can be expressed in the form

$$F = Ax_i + B$$

where both A and E do not depend on x_i. The expression A has two or more terms. A stuck-at 0 at the input x_i causes ail the PI's associated with x_i to vanish. A stuck-at 1 at input x_i causes all the PI's associated with x_i to cover a greater number of minterms . Ey similar argument to that presented earlier we conclude that a fault at input x_i changes the syndrome of the function. Q.E.D.

Definition 4: Two paths that emanate from a common line and reconverge at some forward point are said to be reconverging paths [1].

Definition 5: The inversion parity [4] of a reconverging path is equal to the number of inversions modulo 2 in the path, i.e., the number of inversions modulo 2 between the point of divergence and the point of reconvergence.

In the case of two-level circuits , the points of divergence can only be at the input branches , and the point of reconvergence at the output line. The inverters can appear only at the inputs to the AND gates.

Lemma 4 There exist two-level irredundant circuits which are not syndrome-testable.

Proof: By Lemma 3, a circuit which is not syndrome-testable can not be unate in all its variables. Hence, there exists at least one input, say x_i, from which at least two reconverging paths emanate with non-equal inversion parities. Thus, the function F can be expressed in the form

$$F = Ax_i + B\bar{x}_i + C$$

where A, E and C do not depend on variable xi, and both A and B include at least one term. Thus, a stuck-at fault (stuck-at 0, or at 1) at the fanout branch, x_i, will cause the faulty syndrome to be identical to the fault-free syndrome if and only if

$$S(A\bar{C}) = S(B\bar{C}) \qquad (4)$$

i.e., if 2nd only if the number of minterms added by the growth terms equal the number of ainterms eiiminated by the vanishing terms. Only stuck-at faults at input fanout lines which have property (4) can make the circuit syndrome-untestable. Q.E.D.

Example 3: Consider the function

$$F = xz + y\bar{z} .$$

The fault-free syndrome is $\frac{1}{2}$. The faulty syndrome induced by the fault z/0 (we denote by b/O line b stuck-at 0, and by b/l line b stuck-at 1) is also $\frac{1}{2}$. Thus, F is not syndrome-testable.

it is already obvious at this point that. two-level combinational circuits can be made syndrome-testable by controlling the "size" of the PI's with extra input insertion. For example, the function F of Example 3 may be made syndrome-testable by inserting one extra input, w, to realize the new function $F = wxz + y\bar{z}$.

During normal operation the input w is fed with-a constant logic 1 , while fbr testing purposes it is used as a valid input. The function F is now syndrome-testable since $S(wx) \neq S(y)$.

Lemma 5: See seperate sheet.

From Lemmas 3 and 4 it is evident that the candidates for syndrome-untestability are the non-unate input lines. In order to keep track of the actual lines in which the function F is syndrome-untestable wc define the set

$$T = \{ x_i \mid F \text{ is syndrome-untestable in } x_i \}$$

Clearly,

$$T \subseteq \{ x_i \mid F \text{ is non-unate in } x_i \}$$

Our goal is to reach a syndrome-testabir realization of a given function while adding tne minimum number of extra inputs (control inputs). We choose to add the control inputs in an uncomplemented form (complemented inputs are acceptable as well). By Lemma 3 the modified function will always be syndrome-testable in the control inputs.

Procedure 1 describes an algorithm of designing syndrome-testabie two-level combinational circuits . The syndrome-testable design is achieved by modifying the original irredundant sum of products. The modification requires an introduction of a nearly-minimai number of control inputs. We use the following notation in describing the algorithm:

C = set of control iines.

PI_j = prime impiicant number j.

PI = set of prime implicants.

j = a number specifying how many times has the function been modified so far.

Lemma 5: Every two - levelirredundant combinational circuit can be made syndrome - testable by attaching extra inputs to the AND gates.

Proof: Attach extra inputs such that condition 4 no longer holds.

Q.E.D.

$F^{(j)}$ = the j-th modified function.

$T^{(j)}$ = the set T which corresponds to function $F^{(j)}$.

FLAG = an identifier describing whether the function should be modified with an existing control variable or with a new one.

Procedure 1:

Step 1: Initialization:

$j = 0$

$\mathbf{C} = \phi$

FLAG = 0

$\mathbf{PI} = \{PI_1, PI_2, \ldots, PI_k\}$

$F^{(j)} = F = \sum_{i=1}^{k} PI_i$ ($\sum$ denotes boolean sum)

Step 2: Derive $T^{(j)}$ for the function $F^{(j)}$.

if $T^{(j)} = \phi$ - stop, $F^{(j)}$ is syndrome-testable.

If $\mathbf{C} = \phi$ go to step 4.

Step For each $c \in \mathbf{C}$ and $PI_j \in \mathbf{PI}$ evaluate $T^{(nj)}$ for function

Let $\left| T^{(\alpha\beta)} \right| = \underset{n,j}{\text{Min}} \left\{ \left| T^{(nj)} \right| \right\}$ $\sum_{i \neq j} PI_i + c_n \cdot PI_j$ (when the minimum is attained by several members choose one arbitrarily)

If $\left| T^{(\alpha\beta)} \right| = \left| T^{(j)} \right|$ go to step 4.

otherwise FLAG =0, go to step 5.

Step 4: Add new input: $\mathbf{C} \leftarrow \mathbf{C} \cup \{c_\delta\}$.

For each $PI_j \in \mathbf{PI}$ evaluate $T^{(\delta j)}$ for the function

$$\sum_{i \neq j} PI_1 + c_\delta \cdot PI_j$$

Let $\left|T^{(\delta\mu)}\right| = \underset{j}{\text{Min}} \left\{ \left|T^{(\delta j)}\right| \right\}$

If $\left|T^{(\delta\mu)}\right| = \left|T^{(j)}\right|$ then FLAG = 0, otherwise FLAG = 1

Step 5:

$j \leftarrow j+1$

if FLAG = 0 then

$\left|T^{(j)}\right| = \left|T^{(\alpha\beta)}\right|$; $PI_\beta \leftarrow c_\alpha \cdot PI_\beta$

If FLAG = 1 then

$\left|T^{(j)}\right| = \left|T^{(\delta\mu)}\right|$; $PI_\mu \leftarrow c_\delta \cdot PI_\mu$

Go to step 2.

By Lemma 5, Procedure 1 is guaranteed to halt. Although, at each iteration of the algorithm the optimal input insertion (or one of the optimal insertions in the case of multiple choices) is obtained, an overall optimal solution is not guaranteed. The reason for that is that local optimization does not necessary lead to global optimization.

Example 4: Find the syndrome-testable design with the minimum number of control inputs for the function

$$F = x_1\bar{x}_2 + \bar{x}_1x_3 + x_2\bar{x}_3 + x_4x_5 + \bar{x}_4\bar{x}_5$$

Solution: Let $PI_1 = x_1\bar{x}_2$, $PI2 = \bar{x}_1x_3$, $PI_3 = x_2\bar{x}_3$, $PI_4 = x_4x_5$ $PI_5 = \bar{x}_4\bar{x}_5$.

First pass of Procedure 1 yields

$$T^{(0)} = \left\{x_1, x_2, x_3, x_4, x_5\right\}$$
$$F^{(0)} = F$$
$$T^{(1)} = \left\{x_3, x_4, x_5\right\}$$
$$F^{(1)} = c_1x_1\bar{x}_2 + \bar{x}_1x_3 + x_2\bar{x}_3 + x_4x_5 + \bar{x}_4\bar{x}_5$$

Second pass of the algorithm gives

$$T^{(2)} = \{x_3\}$$

$$F^{(2)} = c_1x_1\bar{x}_2 + \bar{x}_1x_3 + x_2\bar{x}_3 + c_1x_4x_5 + \bar{x}_4\bar{x}_5$$

Third pass of Procedure 1 yields

$$T^{(3)} = \phi$$

$$F^{(3)} = c_1x_1\bar{x}_2 + c_2\bar{x}_1x_3 + x_2\bar{x}_3 + c_1x_4x_5 + \bar{x}_4\bar{x}_5$$

The testable design is shown in Fig. 6..

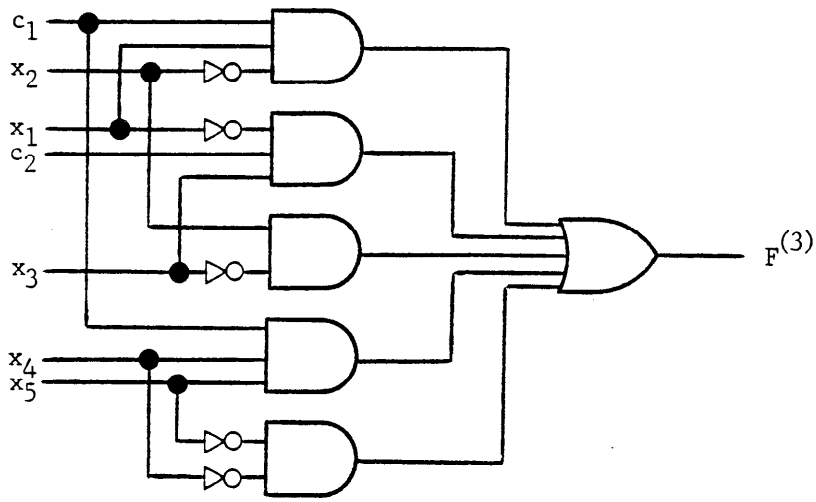


FIGURE 6: The syndrome-testable design for example 4.

The oniy difficulty invoiving the design arises when n is large. Since the test procedure requires the application of all 2^n input combinations, the design will be worthwhile for $n \leq 20$ (which is equivalent to about 1 second of testing with a 1 MHZ machine). However, for $n > 20$, the problem can be overcome by designing the circuit with subcircuits , as shown in Fig. 7 , such that every subcircuit has no more than 20 inputs. This partition enables simultaneous testing of each subcircuit. This improvement will cost in$\lceil \frac{q}{20} \rceil$ extra outputs, where q is the number of inputs to the AND gates (the extra OR gates do not constitute a problem in present day

technology) . Each subcircuit must be designed to be syndrome-testable. Note that the different subcircuits do not need to have disjoint inputs in order to be able to test them in parallel. In order to observe this important fact, consider the simple example of two subcircuits having the input sets $\{x_1, x_2\}$ and $\{x_2, x_3\}$. The following 4 test vectors apply all input combinations to both subcircuits:

	t_1	t_2	t_3	t_4
x_1	0	1	0	1
x_2	0	0	1	1
x_3	0	1	0	1

This proposed partition reduces the syndrome-test length from 2^n to $2^{\text{Max}_i\{n_i\}}$, where n_i is the number of inputs to the i-th subcircuit.

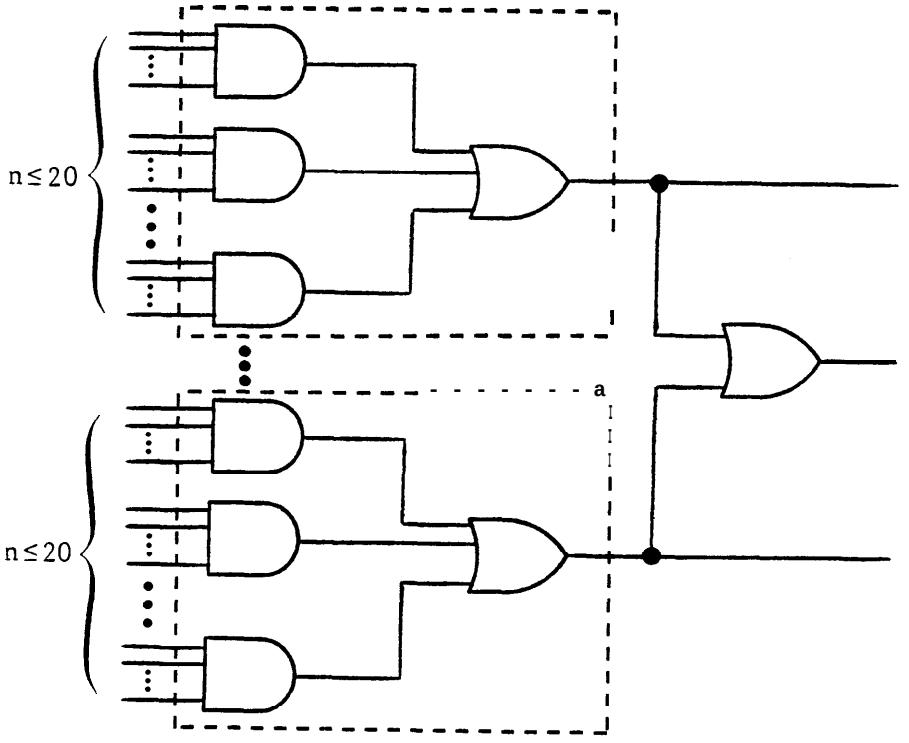


FIGURE 7: The proposed partition for large circuits.

3.2 Syndrome-Testable Design of General Combinational Circuits

Lemma 6: Every fanout-free irredundant combinational circuit, composed of AND, CR, NAND, NCR and NOT gates is syndrome-testable.

Proof: In a fanout-free circuit there is a unique path from any line to the circuit output. Thus, if we label a given line by g , the output F will be either positive or negative in g, depending on the inversion parity of the path originating at g and terminating at the output line. Assume without loss of generality that F is positive in g. We can , therefore, represent F as

$$F = Ag + B$$

where $g = g(x_{i_1}, x_{i_2}, \ldots, x_{i_n})$,

and both A and B do not oepend on $x_{i_1}, x_{i_2}, \ldots, x_{i_n}$.

A stuck-at 0 fault on line g eliminates all the minterms associated with Ag , and does not create any new ones. Therefore, the circuit is syndrome-testable with respect to g/0. in order to prove that g/1 is syndrome-testable, we use a contra-positive approach. Assume that g/l is syndrome-untestable. Thus, all the minterms covered by $A\bar{g}$ are already covered by F. Therefore F can be represented by

$$F = Ag + A\bar{g} + B = A + B$$

which is independent of g, contradicting the previous assumption that F is positive in g. Q.E.D.

Lemma 7: Let $g = g(x_{i_1}, x_{i_2}, \ldots, x_{i_n})$ be a line in a general combinational circuit. Let the equivalent sum of products of the function F with respect to line g be

$$F = Ag + B\bar{g} + C$$

then,

The fault g/0 is syndrome-untestable if and only if

$$S(A\bar{C}g) = S(B\bar{C}g) \tag{5}$$

and, the fault g/1 is syndrome-untestable if and only if

$$S(A\bar{C}\bar{g}) = S(B\bar{C}\bar{g}) \tag{6}$$

Proof: We prove relation (5). The proof of (6) is similar. The fault g/0 is syndrome-untestabie if and only if

$$S(F) = S(B + C)$$

Using Lemmas 1 and 2 , we have

$$S(F) = S(A\bar{C}g) + S(B\bar{C}\bar{g}) + S(C) \quad .$$

$$S(E + C) = S(B\bar{C}) + S(C)$$

Thus the fault g/0 is syndrome-untestable if and only if

$$S(A\bar{C}g) + S(B\bar{C}\bar{g}) = S(B\bar{C}) \quad , \text{ or}$$

$$S(A\bar{C}g) = S(B\bar{C}g)$$

Q.E.D.

Corollary 1: Let g be any line in an irredundant combinational circuit composed of AND ,OR, NAND, NCR and NOT gates. If there exists only one path from g to the circuit output, the function F is syndrome-testable with respect to faults in g.

Proof: Directly from Lemmas 6 and 7.

EXCLUSIVE-OR and EQUIVALENCE gates are source of problems to syndrome testing because of their inherent non-unateness. The following Lemma displays the conditions necessary so that the faults on the inputs of an EXCLUSIVE-OR gate be syndrome-untestable. The corresponding conditions for an EQUIVALENCE gate can be obtained from Lemmas 1 and 8.

Lemma 8: Let $F = g \oplus h$, where $g = g(x_1, x_2, \ldots, x_n)$ and $h = h(x_1, x_2, \ldots, x_n)$. The fault h/0 is syndrome-untestable if and only if

$$S(gh) = S(\bar{g}h) \tag{7}$$

and the fault h/1 is syndrome-untestable if and only if

$$S(gh) = S(\bar{g}\bar{h}) \tag{8}$$

Proof: We prove (7). The proof of relation (8) is similar. The fault h/0 is syndrome-untestable if and only if

$$S(g \oplus h) = S(g)$$

Thus,
$$S(g\bar{h}) + S(\bar{g}h) = S(g)$$

$$S(\bar{g}h) = S(gh)$$

Q.E.D.

Corollary 2: Let $F = g \oplus h$, where $g = g(x_{i_1}, x_{i_2}, \ldots, x_{i_u})$, $h = h(y_{i_1}, y_{i_2}, \ldots, y_{i_v})$ such that $\{x_{i_1}, \ldots, x_{i_u}\} \cap \{y_{i_1}, \ldots, y_{i_v}\} = \phi$. Then, the fault h/0 is syndrome-untestable if and only if

$$S(g) = 1/2 \tag{9}$$

and the fault h/1 is syndrome-untestable if and only if

$$S(h) + S(g) = 1 \tag{10}$$

Proof: Directly from Lemmas 1 and 8.

It is evident from Corollary 2 that attaching an extra input to an EXCLUSIVE-OR gate will not fix the syndrome-untestable condition since relation (9) will hold for the new input. There are two basic approaches to fix the syndrome-untestable condition with respect to inputs of an EXCLUSIVE-OR gate. One way of handling the problem is by breaking the inherent symmetry of the gate. This can be done when the actual implementation of the gate is known. For example, in the case where the EXCLUSIVE-OR gate is implemented by NAND gates, the output can be made syndrome-testable with respect to faults at the inputs by adding a control input, c, as shown in Fig. 8.

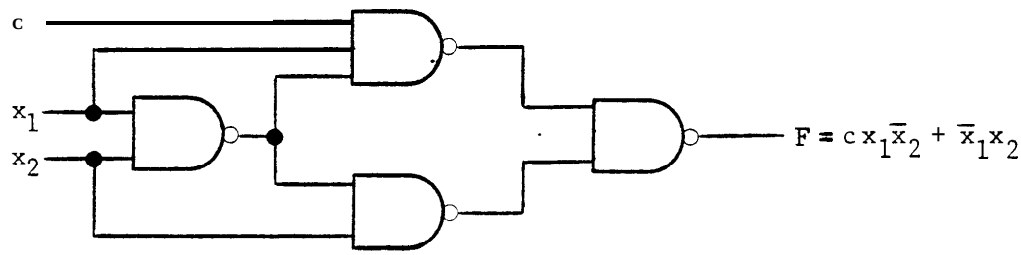
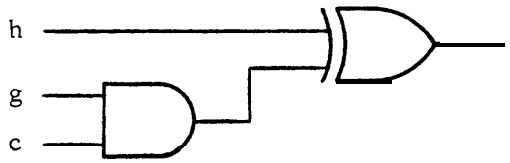
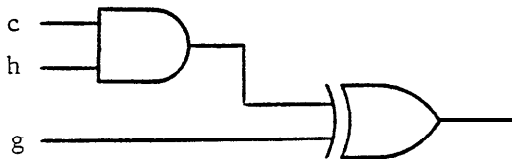


FIGURE 8: A syndrome-testable modification of an EXCLUSIVE-OR gate. For normal operation c = 1.

The second approach is to add a control input and an extra AND gate to the input lines of the EXCLUSIVE-OR gate as shown in Fig. 9. However, one should verify that the syndrome-untestable condition has been invalidated since this modification does not work in general.



9.a



9.b

FIGURE 9: 9.a - Using an extra AND gate and an extra input to fix the syndrome-untestabie condition in h/0. The fault h/O is syndrome-testable if $S(g) \neq S(h)$ and $\bar{g}h \neq 0$.

9.b - Using an extra AND gate and an extra input to fix the syndrome-untestable condition in h/1. The fault h/l is syndrome-testable if $S(g\bar{h}) \neq S(gh)$ and $S(\bar{g}h) \neq S(gh)$.

The problem is how to modify the design of a general combinational circuit so that. it will be syndrome-testable. We try to do it by means of control input insertion and/or a small amount of extra logic. The control inputs connected to AND and NAND gates should be applied with a constant 1 under normal operation, while the control inputs attached to OR and NOR gates should be applied with a

constant 0 under normal operation.

From the previous Lemmas and Corollaries it is clear that the candidate lines for syndrome-untestability are the fsncut lines (including input fanout) and inputs and outputs of EXCLUSIVE-OR gates. The following procedure describes the modification algorithm.

Procedure 2:

Step 1:

Find the set of lines in which the function is syndrome-untestable by finding the equivalent sum of products with respect to the candidate lines for syndrome-untestability.

Step 2:

Use Procedure 1 with the following modifications:

$PI \rightarrow$ Set of AND, NAND, OR, NOR gates.

$\sum_{i \neq j} PI_i + c \cdot PI_j \rightarrow$ The function obtained by attaching the control input c to gate number j.

For the faults at the inputs of EXCLUSIVE-OR gates use either the method of Fig. 8, or Fig. 9.

Since the conditions of syndrome-untestability are very strict (i.e. relations (3)-(6)) it is believed that , in general, very few lines will meet them. Thus, only few extra inputs will be required to achieve a syndrome-testable design. Table 1 displays few MSI combinational logic, their number of pins and the number of extra pins needed to make them syndrome-testable. As seen from Table 1, the number of extra pins do not exceed 1 (or 5%) for these

functions. This is not surprising because one extra input can invalidate the syndrome-untestable condition in various portions of the circuit . Because of the strictness of the syndrome-untestable condition, it is feasible to correct untestable conditions of several faults by means of only one additional input. Figure 10 shows a syndrome-testable modification of SN74 181 . The modification requires one extra input and two extra AND gates. Notice that the extra AND gates could have been avoided if we assumed a certain realization of the EXCLUSIVE-GR gates (see Fig. 8).

TABLE 1: The number of control inputs necessary in order to make the circuits syndrome-testable as a function of number of pins for various MSI combinational logic.

Function	# Pins	# Control inputs
Dual 4-line-to-1-line Data selectors/Mux SN74 153	16	1
Data selectors/Mux SN74 150	24	1
Mux SN74151	16	1
Dual carry save adders SN74183	14	
Look-ahead carry generators SN74182	16	0
Arithmetic logic units SN74181	24	1
Total	110	5

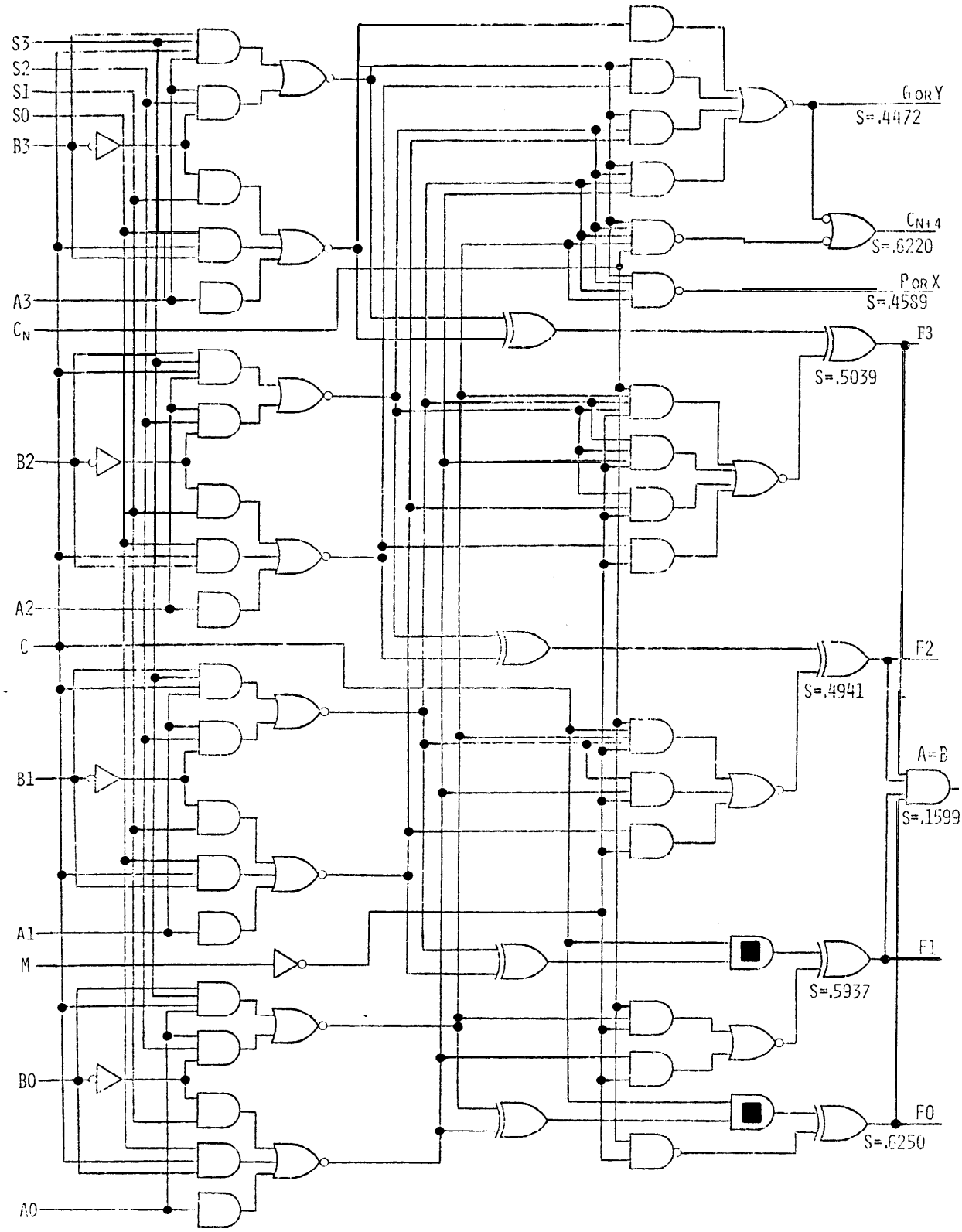


FIGURE 10: Syndrome-testable design of Arithmetic Logic Units, SN74181, requiring me extra input, C, and two extra gates (marked ■). The output syndromes are rounded to four decimal digits.

4. SUMMARY AND CONCLUSIONS

A new approach to the design of' testable combinational circuits was presented in this paper. The design method is to modify given realizations by inserting extra inputs sc that the final circuit will be syndrome-testabie. A procedure that produces a nearly minimal number of extra input insertions was described. It was al so shown how to partition very large two-level combinational circuits to subcircuits such that all subcircuits can be tested simultaneously to reduce the total testing time. Although we restricted ourselves, in this paper, to single-output networks? the ideas easily generalize to multiple-output networks as well.

It is well known that the most severe restriction on IC manufacturers is the number of pins per chip. Although the testable design requires an increase in the number of pins , it is believed that the pin overhead is very low because of the strictness of tne syndrome-untestable condition. Up to this time, we were unable to find 3 combinat ionai circuit that requires more than two control inputs in order to make it syndrome-testable.

For the cases were an increase in the number cf pins is considered an impractical solution, one could combine syndrome and classical testing in the following way: run a syndrome-test procedure first and then test for the syndrome-untestable faults by using a classical fault detection experiment. Since the syndrome-test

procedure will probably cover most of the stuck-at faults, the classical fault detection experiment, will need to be designed to cover only the previously uncovered faults. This combined procedure will require, therefore, a restricted dictionary of the expected response and will not be contingent upon modifying the logic design by adding control inputs.

REFERENCES

[1] Armstrong, D.E., "On Finding a Nearly Minimal Set of Fault Detection Tests for Combinational Logic Nets," IEEE Trans. Electron. Comout., Vol. EC-15, pp. 66-73, Feb. 1966.

[2] Hayes, J.P., "On Realization of Boolean Functions Requiring a Minimal or Near-Minimal Number of Tests," IEEE Trans. Comput., Vol. c-20, pp. 1506-1513, Dec. 1971.

[3] Hayes, J.P., "Transition Count Testing of Combinational Circuits," IEEE Trans. Comput., Vol.C-25, pp. 613-620, June, 1976.

[4] Rohavi, I., and Z. Kohavi, "Detection of Multiple Faults in Combinationai Logic Networks," IEEE Trans. Comput., Vol. C-21, pp. 556-568. June 1972.

[5] Losq, J., "Referenceless Random Testing," Proc. Sixth Ann. Symp. on Fault-Toierant Comput., Pittsburgh, PA, June 21-23, 1976, pp. 108-113.

[6] Roth, J.P., W.G. Bouricius, and P.R. Schneider, "Programmed Algorithms to Compute Tests to Detect and Distinguish Eetween Failures in Logic Circuits," IEEE Trans. Comput., Vol. EC-16, pp. 567-580, Oct. 1967.

[7] Savir, J., "Fault Detection in Modular Combinational Networks," Proc. Ninth Convention of Electrical and Electronics Eng. in Israel, April 1975, B-2-4.

[8] Tzidon, A., I. Berger and M. Yoeli, "A Practical Approach to Fault Detection in Combinational Networks," IEEE Trans. in Computers, Vol. C-27, No. 10, pp. 968-971, October 1978.