Fault Detection Techniques

By
Donnamaie E. White, Ph.D.

Chap 12 of *Advanced Logical Circuits Design Techniques*

Table of Contents

List of Figures

12 Fault Detection Techniques

12.1 Faults

12.1.1 Fault Definition

In any circuit composed of logic gates there is the possibility of the occurrence of a fault. A fault is defined to have occurred when any circuit variable assumes a value (1, 0, or X) which differs from that expected, that is, which violates the original circuit equations.

12.1.2 Masking a Fault

The presence of an internal or input fault may not be observable at the circuit output, in which case, the fault is considered to be ***masked***.

A single fault may be masked as a result of:

1) Reconvergent fan-out, where unequal parity changes have occurred;
2) Circuit redundancy;
3) Previous occurrence of an undetectable fault.

Masked faults are undetectable by their definition since the observed circuit behavior is correct. However, the occurrence of a second fault may uncover a previously undetectable fault. To be complete, the test set must include tests for this case.

Figure 12-1 presents a redundant circuit and its ***Marquand Map***. The circuit implements three terms to cover the eight points on the map when the two terms, $\underline{x}_3x_0$ and x_3x_1 are sufficient. *(**Note**: Underscore used for negation.)*

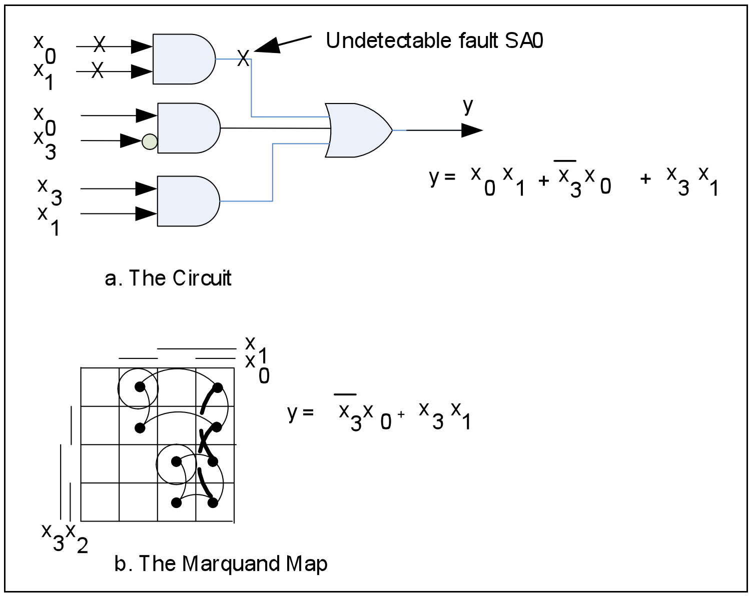


Figure 12-1 Redundant Circuit

12.1.3 Fault Types

Faults may be ***indeterminate*** in value (suspended between logical "1" and logical "0"), or ***determinate*** in value (exhibiting a "0" or a "1").

Faults may be ***transient*** (indeterminate, time-varying), in which case they are elusive and difficult to detect.

Faults may be ***permanent*** (considered "hard" or "solid"), in which case they are easy to detect if they are not masked and if a proper test is used.

Faults may be multiple in occurrence, which has always been considered a rare event. (With higher circuit densities, this event has increased somewhat in probability.)

Faults may occur singly, which is considered to be the most likely event.

Further, multiple faults can occur in such a manner that there is an equivalent single fault for them. A test which detects the presence of this equivalent [single] fault will be sufficient to detect the presence of [the multiple] faults. It should be noted the fault ***identification*** is not possible [in this circumstance].

12.1.4 Fault Equivalencies

There are several equivalencies that exist which are useful in fault detection and which make fault location considerably more difficult. Some of these equivalencies are:

1) One or more inputs to an OR gate stuck at "1" (SA1) is equivalent to the output of the OR gate being SA1.
2) One or more inputs to an AND gate stuck at "0" (SA0) is equivalent to the output of the OR gate being SA0.
3) All inputs to the OR gate SA0 is equivalent to the output of the OR gate SA0.
4) All inputs to an AND gate SA1 is equivalent to the output of the AND gate SA0.
5) Failures on both the inputs (one or more failures) and the output of a gate will propagate the gate output failure, masking the input faults.
6) Any gate output has, as an equivalent, a single gate input fault (not necessarily an input to that gate) or multiple input faults. However, any gate input fault does not necessarily have an equivalent gate output fault.

12.1.5 The Problem

The most common fault for current technology (such as DRL, DTL, RTL and TTL[1]) is the single, permanent, stuck-at fault where one of the following has occurred:

1) A gate output is stuck at logical "1" (SA1)
2) A gate output is stuck at logical "0" (SA0)
3) Any single gate input is stuck at logical "0"
4) Any single gate input is stuck at logical "1"

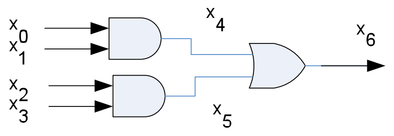
NOTE: The single fault assumption is not proper for the initial circuit checkout.

Component failures which alter or affect voltage levels, current levels, pulse widths, or other circuit timing, but which do not alter or affect the logical function realized by the circuit, will not be considered here, These qualitative failures are presumed to be detected during initial electrical parametric testing.

To be complete, a test set must be able to detect any single detectable fault. It should also include test for multiple faults, where such faults are not covered by equivalent single faults. Further, the test set should include tests for faults which become detectable when another undetectable fault occurs (this is a special type of multiple fault).

[1] 1979

The circuit of Figure 12-2 has seven input lines and a SA0 or SA1 fault may occur on any line. There are, therefore, fourteen (14) single faults possible for this case. There are, by computation, 84 possible double faults which may occur. All 84 double faults are "covered" by the fourteen single faults.



a. TNC Labeling for a two-level AND-OR circuit

Equation:

$$x_6 = x_4 + x_5 = x_0x_1 + x_2x_3$$

Number of possible single-fault locations: 7

Number of possible single-faults: $\binom{2}{1}\binom{7}{1} = 14$

Number of double fault combinations :

$$\binom{2}{1}\binom{2}{1}\binom{7}{2} = 84$$

Figure 12-2 Sample Circuit used in Comparing Methods

Further, of the fourteen single faults, not all are distinct. As an example, a test for x_0 SA0 also tests x_4 SA0 and x_6 SA0. This allows a test set to be derived which is smaller than the exhaustive test set (24 = 16 tests for this case) and often smaller than that produced with the "one-test-per-fault" approach.

If it is assumed that one or more permanent stuck-at faults (SA1 or SA0), or the equivalent, has occurred, then the problem is to construct a complete and minimal test set such that this fault is detected, provided that masking has not covered the effects.

12.2 The Test Sequence

The original paper on Boolean differences by this author, written as a class exercise in 1971, began the search to find a method of fault detection which would be producible by the Boolean Analyzer. IN the course of studying the fault detection problem, a number of papers, several of which have been used as references, were examined and the various methods presented catalogued by their approach to the problem.

With the discovery of certain properties of the edge structure of the Existence function, which allows the formation of links, and the first hypothesis for sequence construction, most of the examples from those papers were re-examined. In each case, both the original procedure and the Test Sequence procedure were performed and the results compared. For combinational circuits, the limit of this research, the results in all cases were favorable [to the Test Sequence].

12.2.1 Deriving the Existence Function

12.2.1.1 The Equations

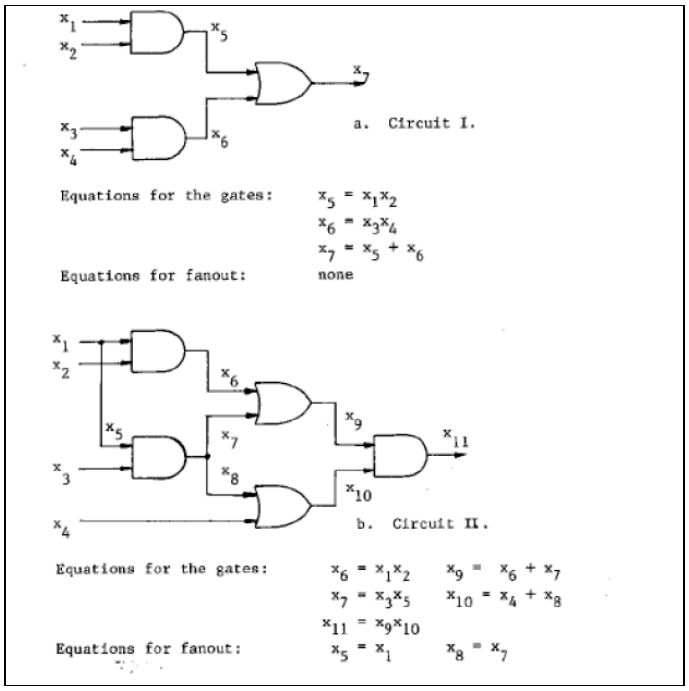
The first step in developing a Test Sequence is the derivation of the equations of a given circuit configuration. Equations are first derived for each of the logic gates, using the circuit primary inputs as known variables, and all intermediate and primary outputs as unknown variables.

A ***primary input*** is an input connected to an external source. A ***primary output*** is an output going to an external "sink" or connection. All other lines (interconnects) are to be considered as internal or intermediate lines and their labels are handled as intermediate variables. For the purposes of this paper, a logical gate will be constrained to be an SSI-level gate (AND, OR, NAND, NOR, or INV (a.k.a. NOT)).

As an example of this first type of equation derivation, there are three equations for the first circuit, Circuit 1, of Figure 12-3. They are:

$$x_6 = x_3x_4 \qquad x_5 = x_1x_2 \qquad x_7 = x_5 + x_6$$

Labeling for this case has begun with x_1.



x_1 x_2 x_5 x_3 x_4 x_6 x_7
a. Circuit I.
Equations for the gates: $x_5 = x_1x_2$ $x_6 = x_3x_4$ $x_7 = x_5 + x_6$
Equations for fanout: none
x_1 x_2 x_5 x_3 x_4 x_6 x_7 x_8 x_9 x_{10} x_{11}
b. Circuit II.
Equations for the gates: $x_6 = x_1x_2$ $x_9 = x_6 + x_7$ $x_7 = x_3x_5$ $x_{10} = x_4 + x_8$ $x_{11} = x_9x_{10}$
Equations for fanout: $x_5 = x_1$ $x_8 = x_7$

Figure 12-3 Examples of Labeling; Equations

For the case where there is a fan-out of primary input lines, the first line is labeled and treated as a known variable while the remaining fan-out lines are labeled and treated as intermediate variables. Equations must be added to the equation set which equate the first [fan-out] line with the remaining fan-out lines.

For example, the second circuit, Circuit 2, of Figure 12-3, has the two equations:

$$x_5 = x_1 \qquad x_8 = x_7$$

added to the listed equations for the input/output relationships of the circuit.

Internal fan-out and fan-in are handled in a similar manner. Additional examples are presented later to demonstrate the procedure.

12.2.1.2 Required Labeling

There is a labeling convention which has been used in the above equations and which has been given the name ***TNC (Terminal Numbering Convention)*** in previous papers. Under this convention, the primary input lines are labeled with the lowest variables (x_0 , x_1 are the two most common starting labels). Each gate input and output receives a distinct [unique] label with the convention that, for any gate, the indices of the labels on the inputs are each lower than the index (indices) of the output(s). Any intermediate variables created by fan-out are indexed to comply with this convention. The primary outputs (one or more) receive the highest indices for their variables.

Proper labeling has been shown in Figure 12-3.

12.2.1.3 Equation Reformation

After constructing all of the equations for a circuit, these equations are solved to produce the Existence Function of the circuit.

The initial form of the equation is:

$$F_i(x_0, \ldots, x_{n-1}, x_n, \ldots, x_p) = G_i(x_0, \ldots, x_{n-1}, x_n, \ldots, x_p)$$

where: $i = 1, \ldots, k$ (for k equations)

$\{x_i \mid i = 0, 1, \ldots, n-1\}$ the set of n known variables (in this paper, these are all of the circuit primary inputs)

$\{x_i \mid i = n, \ldots, p\}$ the set of m unknown variables (in this paper, these are all of the circuit primary outputs, internal variables, and fanout of primary inputs)

$p = n - 1 + m$ (when the starting index is zero)

The formalism is usually discarded and the form noted as simply:

F = G

These equations express the validity requirements which must be satisfied for the function to be true, that is, for the circuit output to be logical "1". Therefore:

$$(F = G) \iff (y = 1)$$

where y denotes the output.

Equations in the form (F = G) may be rewritten as:

$$\overline{F}G + F\overline{G} = 0$$

which expresses the validity requirements for the complement function. Therefore:

$$(\overline{F}G + F\overline{G} = 0) \iff (y = 0)$$

The terms of the rewritten equations are either in the form $\underline{F}G$ or $F\underline{G}$ and are referred to as the terms of Y, where Y = $\underline{y.}$ (underscore for negation)

12.2.1.4 Generating the Existence Function

A system of equations in the form $(\overline{F}G + F\overline{G} = 0)$ are solved by the Boolean Analyzer when it is operated in the binary mode. The processing will result in the canceling of all points in the logical space of the system of equations which are covered by at least one of the Terms of Y. This actually "cancels" the points of the Existence Function of Y, leaving its compliment function, the Existence Function of y, in the memory.

For convenience, the Existence Function can be given the form of the Discriminant of the system of equations. This form uses all known variables as the independent (horizontal) index, and all known variables as the dependant (vertical) index. [Referencing the maps.]

The existence Function contains information about all behavioral properties of the circuit. Fault testing problems are solvable by either:

1) Adding equations to the systems of equations prior to generating the existence functions;

2) Performing further processing (with software) on the Existence Function after it is generated.

To demonstrate the generation of an existence Function, Figure 12-4 details the procedure for Circuit 1 of Figure 12-3. The canceled points or squares represent the Existence Function of Y. The remaining points are the desired points of the Existence Function of y. The function is repeated at the top of Figure 12-5.

Repeating the equations of Circuit I, Figure 12.3:

$x_5 = x_1x_2$ $\quad$ $x_6 = x_3x_4$ $\quad$ $x_7 = x_5 + x_6$

These equations represent the relationships between the gate input and output lines. Each equation is in the form:

$$F = G$$

The equations are then rewritten using the relationship:

$$(F = G) \iff (\bar{F}G + F\bar{G} = 0)$$

into the following terms:

$\bar{F}G$: $\bar{x}_5x_1x_2$ $\quad$ $\bar{x}_6x_3x_4$ $\quad$ $\bar{x}_7x_5$ $\quad$ $\bar{x}_7x_6$

$F\bar{G}$: $x_5\bar{x}_1$ $\quad$ $x_6\bar{x}_3$ $\quad$ $x_5\bar{x}_2$ $\quad$ $x_6\bar{x}_4$ $\quad$ $x_7\bar{x}_6\bar{x}_5$

The Existence Function is formed by canceling all points in the logical space which are covered by at least one of these terms. For convenience, the circuit primary inputs are used as the known (horizontal) axis, the remaining variables, intermediate and outputs, are used as the unknown (vertical) axis.

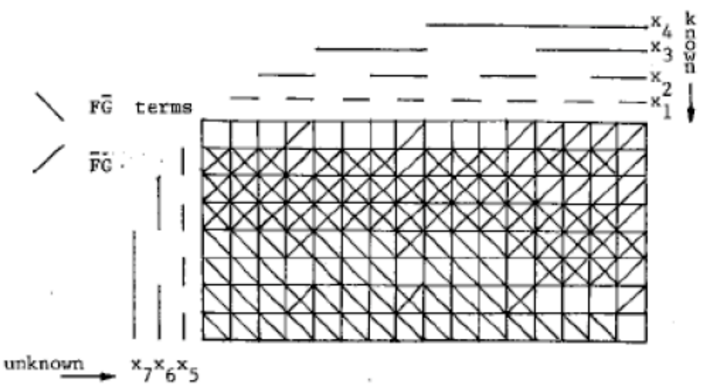
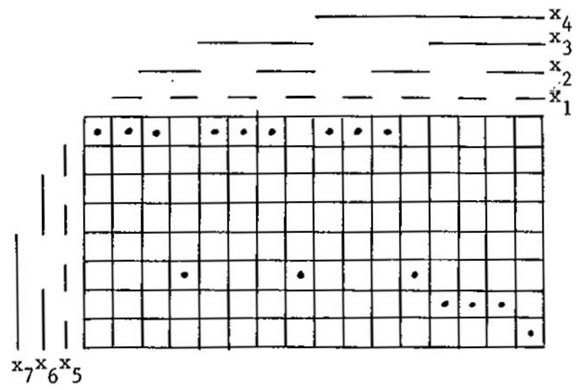
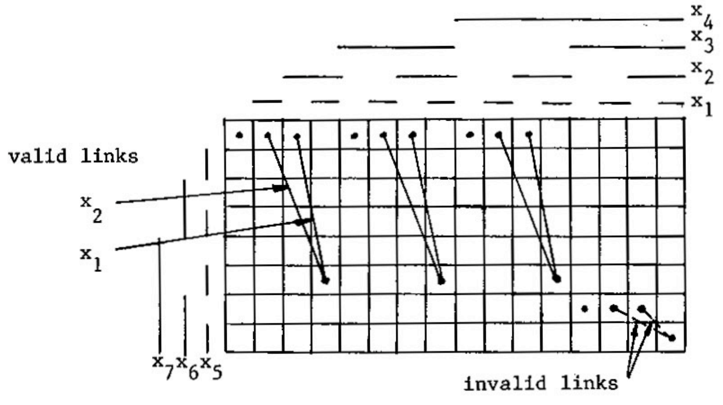


Figure 12-4 Existence Function Generation

Following the cancelation of all points covered by one or more terms, the remaining points in the logical space are the 'ONES' of the Existence Function for the circuit. For this case (Circuit I):





Partial interconnection of 'links'. Those shown are formed from observing a change in the output variable x_7 when either input variable x_1 or x_2 is changed, but not both.

Figure 12-5 Link Formation

12.2.2 Deriving the Test Sequence

12.2.2.1 Formation Rules

A Test Sequence is actually a sequence of test or input vectors which are represented by the indices of the known variables of selected points on the Existence Function. These points are the "one" points of the Existence Function, which are interconnected by ***links*** or ***bars***.

A ***link*** or ***bar*** is formed by connecting any two points in the logical space of the Existence Function of a system which satisfy both of the following conditions:

1) The two points are logical distance one apart in the space of the known (horizontal) axis. **Note**: Logical distance one is defined as the distance between two points in a logical space whose ***minterms*** differ by one and only one variable.
2) The two points are such that they differ in at least one observable output variable. **Note**: The ***observable outputs*** are the ***primary outputs.***

Figure 12-5 presented the Existence Function of Circuit 1 of Figure 12-3. The lower map shown in Figure 12-5 is the Existence Function redrawn with some of the possible links added, specifically, those links formed when the input variable x1 or the input variable x2 is used as the horizontal measure of logical distance.

Figure 12-4 shows how an Existence Function is generated. Using the equations of the system being mapped, the Existence Function is formed by canceling all points in the logical space which are covered by at least one of the terms from the equation (from F = G). For convenience, the circuit primary inputs are used as the ***known*** (horizontal axis) (x1, x2, x3, and x4), and the remaining variables, intermediate and outputs both, are used as the ***unknown*** (vertical) axis. (x7, x6, and x5)

Continuing into Figure 12-5, following the cancellation of all points covered by one or more terms, the remaining points in the logical space are the ***ones*** of the Existence Function for the circuit. Also in Figure 12-5, the lower map shows the partial interconnection of ***links***. Those shown are formed from observing a change in the output variable x7 when either input variable x1 or x2 is changed, but not both. (EXOR)

12.2.2.2 Chain Selections

When all of the possible ***bars*** or ***links*** have been formed, there will be one or more ***chains***, which are ***sets of interconnected links.***

The longest chain will produce the desired Test Sequence.

There are cases where there is no unique longest chain. These cases are;

1) When a circuit is redundant, there are two or more sequences of equal length, and only one is necessary for fault detection. (Either one.)

2) Certain functions in which the terms of the function share no common true or complimented variables.

For those cases requiring two or more disconnected chains, a ***connecting sequence*** is used to join them. A connecting sequence must be the minimal length required to join the two Test Sequences, while keeping its own point logical distance one apart. [Think of this a stepping from one to the other.] This later requirement is to reduce the hazards that could otherwise be introduced by testing.

The completed linking for the example circuit is given in Figure 12-6. The short chains are discarded and the resulting Test Sequence is 5-7-6-14-10-11-9-13-5.

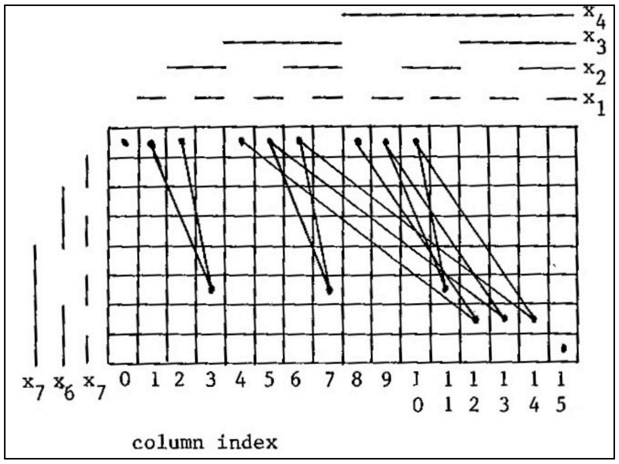


Figure 12-6 The Test Sequence

At first, it was believed that it was sufficient to pick out a closed subset of points along the chain such that each type of link (each input variable variation) was included. By examining the results of other test set generation methods, and by a close evaluation of the results produced by these methods and by the Test Sequence method, it has been determined that the entire chain is necessary.

The use of the entire chain of points ensures complete testing of both single and multiple faults. This includes those multiple faults which do not have single fault equivalences. The nature of the Test Sequence is such that each variable is tested for its ability to change value from *'1'* to *'0'* and from *'0'* to *'1'*.

From Figure 12-6: with all the valid links in place, the links connecting points in columns 1-3-2 and those connecting 4-12-8 are not chained to a sufficient length. The remaining chain connects 5-7-6-14-10-11-9-13-5.

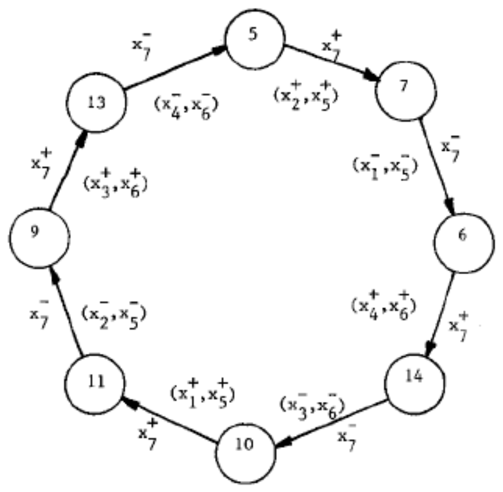
Examination of the sequence of index values shows that, when applied to Circuit 1, each of the input variables will be tested independently for a change from 0 to 1 and again for a change in value from 1 to 0.

Each of the intermediate variables is also tested in as thorough a manner, and it is obvious from the rules of link formation that the output variable (or variables) is (are) tested.

12.2.2.3 Advantages of the Test Sequence

There are several desirable advantages offered by the Test Sequence.

1) The test Sequence may be produced by the Boolean Analyzer, a hardware unit with a speed advantage over software approaches to test-set generation. [1979]
2) An automated tester using the Test Sequence would not require continual resetting (set to known state) between tests and, therefore, throughput with the approach will be higher.
3) The requirement that there be no more than logical distance one between tests reduces the possibility of hazard introduction due to multiple test lead level changes.
4) The Test Sequence formation rules require that, upon the application of any test after the first, at least one observable output will change its logic level. This makes the detection of a failure logically straightforward.
5) The Test Sequence exercises every circuit variable through two-way logic level changes; that is, from ‘1’ to ‘0’ and from ‘0’ to ‘1’. This ensures the detection of ***temporarily correct*** variables (those variables which change value and then become stuck at that value). As an aide, a diagram referred to as the Diagnostic Continuity diagram has been developed which represents graphically the variables which should change levels when the input changes from test to test. (See Figure 12-7.)



The Diagnostic Continuity diagram for Circuit I. The observable output is x_7 and is written separatly from the input and intermediate variables, shown in parenthesis. The nodes are labeled with the input vector.

Figure 12-7 Diagnostic Continuity Diagram

6) The Test Sequence is complete in its coverage of all ***detectable single faults***. For those examples studied, the Test Sequence also covered the multiple faults which were not equivalent to signal faults.
7) The Test Sequence is ***closed***, that is, it returns to the initial test. This reduces the resetting needed between circuits and may also have an advantage when intermittent fault detection is attempted. [Again, in terms of testing in 1979.]

8) There is also the possibility that the natural order of the Test Sequence will be advantageous in the detection of ***bridging faults***. However, this topic was ***not researched*** with the Test Sequence by this author.

From the above advantages, it is concluded that coupling an automated tester of parallel input classification to the Test Sequence procedure will provide an economical and feasible solution to the fault detection problem for combinational circuits.

12.2.2.4 Possible Extension to Sequential Circuits

As an aside to the main line of research, a brief examination of sequential circuits has been made.

For sequential circuits, primary input variables are the known variables; primary output variables, intermediate variables, and feedback variables are the unknown variables. Formation of the linking for these cases requires a different approach, and this is an area of future research. It is believed that the Test Sequence approach can be extended to sequential circuits.

Note: It was used to develop a successful minimal sequence for functional testing for a cross-bar switch. See ***Logic Design for Array-Based Circuits***.

12.3 Examples

This section presents a few of the combinational circuits which were used in the research. The examples are ordered on the basis of their size.

12.3.1 Elementary Gates

The first example is a set of elementary gates, shown with their Existence Functions and Test Sequences in Figure 12-8. In accordance with their definitions by truth table, the OR and NOR gats are seen to have complementary Existence Functions, and they may be tested using the same Test Sequence. The same can be said of the AND and NAND gates. The Test Sequences on both cases contain the identical test specified by Eldred, and by others, for testing these individual gates, as was expected.

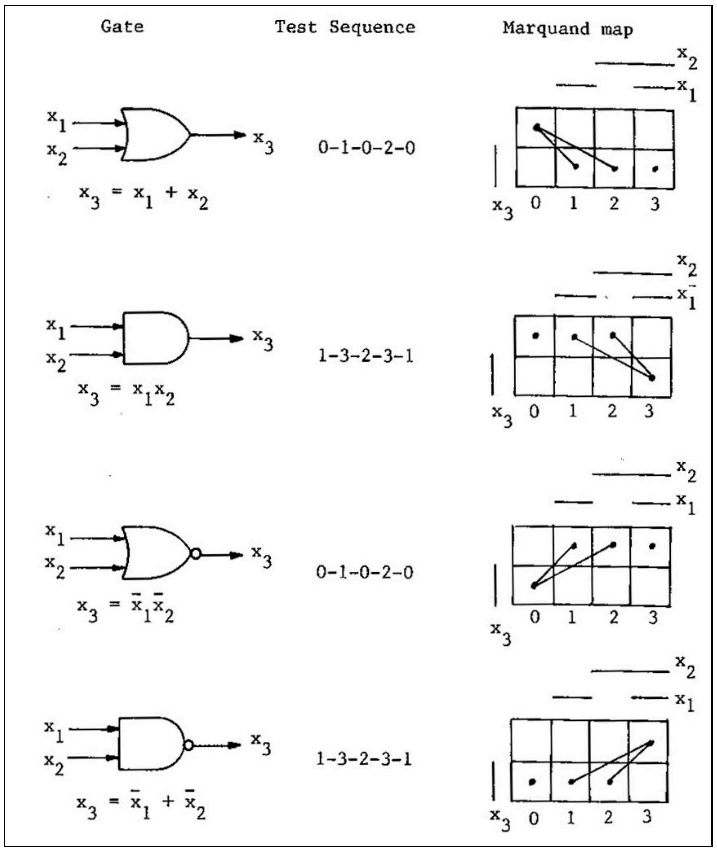


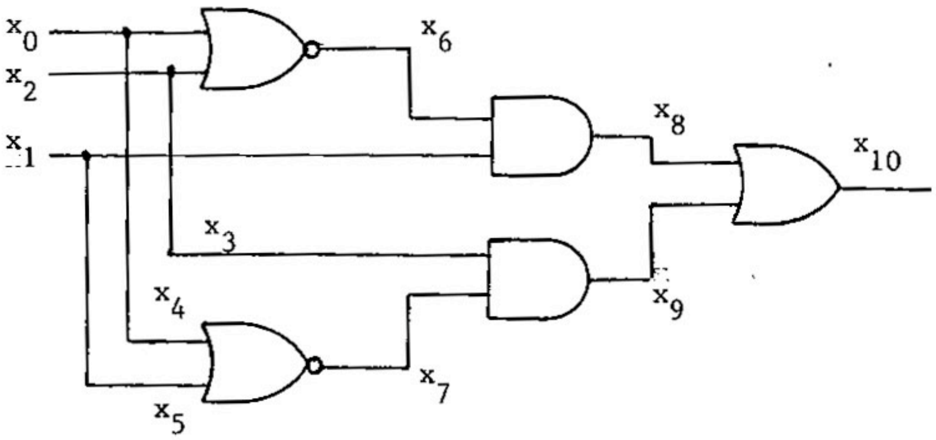
Figure 12-8 Test Sequences for Elementary Gates

12.3.2 Test Sequence vs. Boolean Difference

An example from the paper by Marinos is given in Figure 12-9. This example was analyzed in detail by Marinos using the Partial Boolean Difference approach, and the minimized test set that he derived is given on the second page of the figure. Both of the system equations for the circuit, which has three fanout lines, and the Terms of Y, where Y = $\underline{x}_{10}$ = $\underline{f}$ (underscore for negation) are presented.

Due to the size of the full Existence Function (2^{11} = 2,048 points), a scheme for computing the Test Sequence from an ordered listing of its “1” points has been used. The points are tabulated on the second page with an “*” to denote points chosen for the Test Sequence (for this case, there is only one chain).

The test set is formed by the tests in the Test Sequence and that found by Marinos are seen to be identical.



$$x_{10} = f = \bar{x}_2 x_1 \bar{x}_0 + x_2 \bar{x}_1 \bar{x}_0$$

System equations:

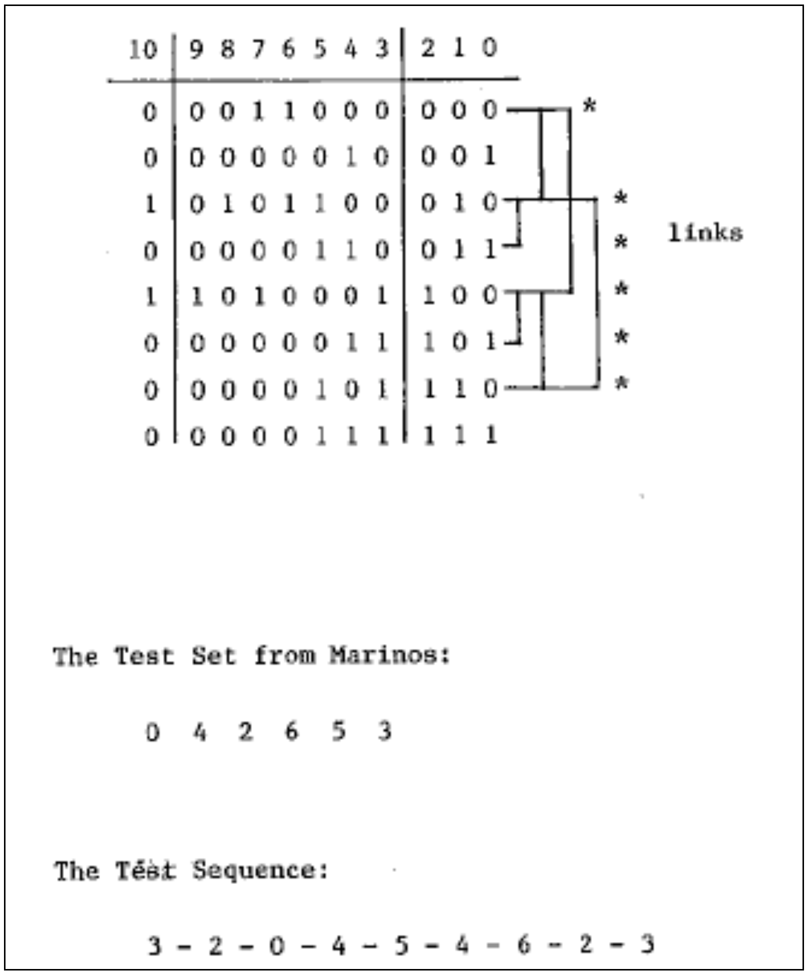
$x_6 = \bar{x}_0 \bar{x}_2$	$x_7 = \bar{x}_4 \bar{x}_5$	$x_8 = x_6 x_1$	$x_3 = x_2$
$x_9 = x_3 x_7$	$x_{10} = x_8 + x_9$	$x_4 = x_0$	$x_5 = x_1$

The Terms of Y:

$\bar{x}_6 \bar{x}_0 \bar{x}_2$	$x_6 x_0$	$x_6 x_2$	$\bar{x}_7 \bar{x}_4 \bar{x}_5$	$x_7 x_4$	$x_7 x_5$
$\bar{x}_8 x_6 x_1$	$x_8 \bar{x}_6$	$x_8 \bar{x}_1$	$\bar{x}_9 x_3 x_7$	$x_9 \bar{x}_3$	$x_9 \bar{x}_7$
$\bar{x}_{10} x_8$	$\bar{x}_{10} x_9$	$\bar{x}_3 x_2$	$x_{10} \bar{x}_8 \bar{x}_9$	$x_3 \bar{x}_2$	$\bar{x}_4 x_0$

The Existence Function is so large that it is not shown here. Instead a table of one points is given and the Test sequence derived from this in the same manner it would have be derived from the Existence Function.

Figure 12-9 An Example from Marinos (see references)



10	9	8	7	6	5	4	3	2	1	0	
0	0	0	1	1	0	0	0	0	0	0	*
0	0	0	0	0	0	1	0	0	0	1	
1	0	1	0	1	1	0	0	0	1	0	*
0	0	0	0	0	1	1	0	0	1	1	*
1	1	0	1	0	0	0	1	1	0	0	*
0	0	0	0	0	0	1	1	1	0	1	*
0	0	0	0	0	1	0	1	1	1	0	*
0	0	0	0	0	1	1	1	1	1	1	

The Test Set from Marinos:

0 4 2 6 5 3

The Test Sequence:

3 - 2 - 0 - 4 - 5 - 4 - 6 - 2 - 3

Figure 12-9 Con't

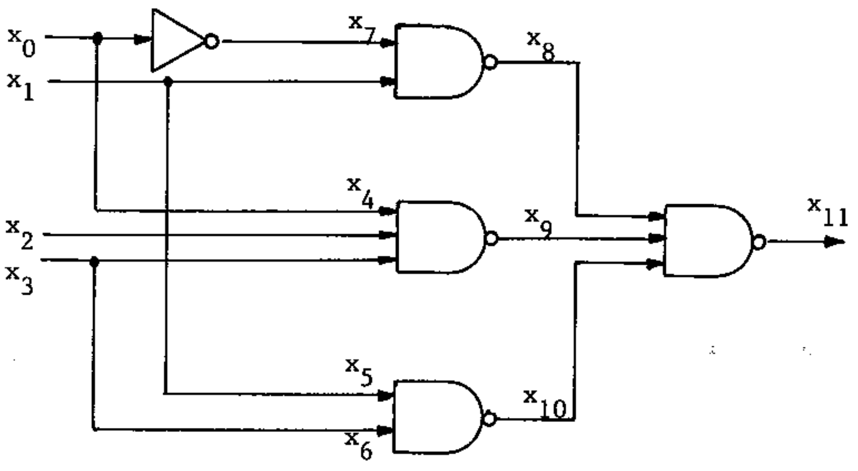
12.3.3 A Diagnostic Table

A fan-out example is shown in Figure 12.10 with its equations and, again, due to the size of the Existence Function (2^{12} = 4,096 points), only the '1' points are shown. The links have again been added to the value table.

As a means of evaluating the Y=Test Sequence, a Diagnostic Table, similar to a fault table, was designed and is presented in Figure 12.11. The links are grouped as row labels according to the input variable which alters value, and the indices of all circuit variables are used as column headings.

- A '+' indicates that a variable changes its logical value from '0' to '1';
- A '-' means that the reverse occurs;
- A blank (ƀ or ' ') means that no change has occurred (the links are shown for one direction only). This is a tabular version of the Diagnostic Continuity diagram.
- The Test Sequence points are shown by '#'
- The Test Set points generated by authors Bearnson and Carroll are shown by '*'

One variable of the link pair appearing in the Test Set is sufficient for the '*' to appear on that link. As can be seen, the methods produced equivalent tests. The variation is between tests 7 and 3, which, when all variable are scanned for these two input configurations, are seen to produce identical internal and output values.



System equations:

$x_7 = \bar{x}_0$ $\quad x_4 = x_0$ $\quad x_5 = x_1$ $\quad x_6 = x_3$

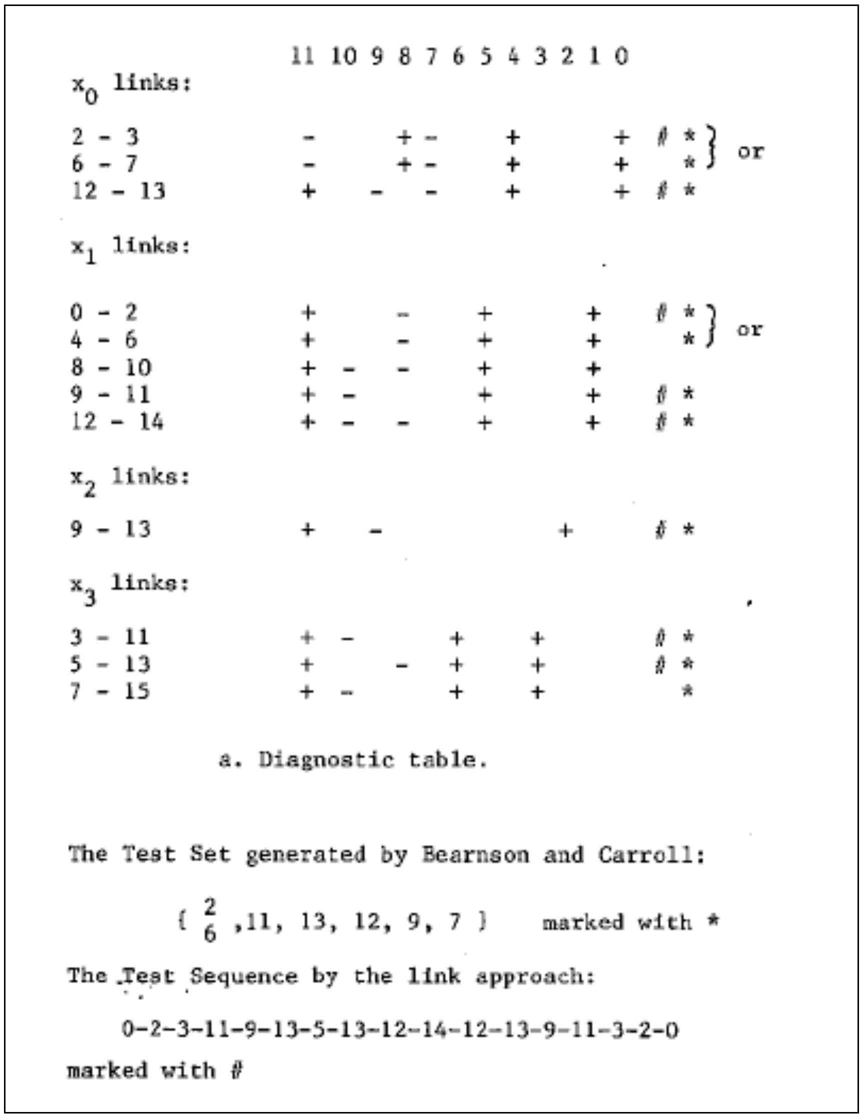
$x_8 = \bar{x}_7 + \bar{x}_1$ $\quad x_9 = \bar{x}_4 + \bar{x}_2 + \bar{x}_3$ $\quad x_{10} = \bar{x}_5 + \bar{x}_6$

$x_{11} = \bar{x}_8 + \bar{x}_9 + \bar{x}_{10}$

11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	0	0	0	0	0	0	0
0	1	1	1	0	0	0	1	0	0	0	1
1	1	1	0	1	0	1	0	0	0	1	0
0	1	1	1	0	0	1	1	0	0	1	1
0	1	1	1	1	0	0	0	0	1	0	0
0	1	1	1	0	0	0	1	0	1	0	1
1	1	1	0	1	0	1	0	0	1	1	0
0	1	1	1	0	0	1	1	0	1	1	1
0	1	1	1	1	1	0	0	1	0	0	0
0	1	1	1	0	1	0	1	1	0	0	1
1	0	1	0	1	1	1	0	1	0	1	0
1	0	1	1	0	1	1	1	1	0	1	1
0	1	1	1	1	1	0	0	1	1	0	0
1	1	0	1	0	1	0	1	1	1	0	1
1	0	1	0	1	1	1	0	1	1	1	0
1	0	0	1	0	1	1	1	1	1	1	1

links

Figure 12-10 An Example for Bearnson and Carroll



	11	10	9	8	7	6	5	4	3	2	1	0			
x_0 links:															
2 - 3	-			+	-			+				+	#	*	} or
6 - 7	-			+	-			+				+		*	
12 - 13	+		-		-			+				+	#	*	
x_1 links:															
0 - 2	+			-			+				+		#	*	} or
4 - 6	+			-			+				+			*	
8 - 10	+	-		-			+				+				
9 - 11	+	-					+				+		#	*	
12 - 14	+	-		-			+				+		#	*	
x_2 links:															
9 - 13	+		-							+			#	*	
x_3 links:															
3 - 11	+	-				+			+				#	*	
5 - 13	+			-		+			+				#	*	
7 - 15	+	-				+			+					*	

a. Diagnostic table.

The Test Set generated by Bearnson and Carroll:

$\{ \begin{smallmatrix}2\\6\end{smallmatrix}, 11, 13, 12, 9, 7 \}$ marked with *

The Test Sequence by the link approach:

0-2-3-11-9-13-5-13-12-14-12-13-9-11-3-2-0

marked with #

Figure 12-11 Tabular Sequence Generation

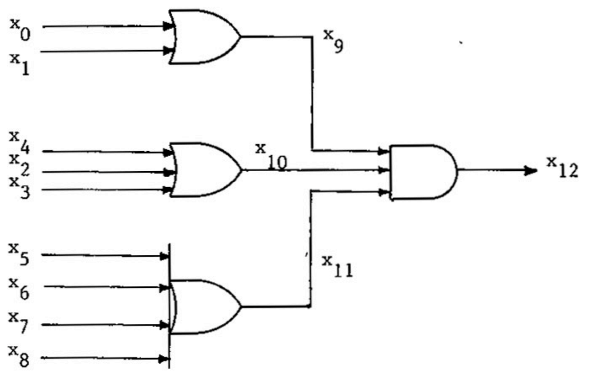
12.3.4 Equivalent Circuits: Test Sequence vs. Kohavi's Maps

An example circuit is sh own in Figure 12-12 and in Figure 12-13b. Figure 12-12 shows the labeling used in Kohavi's paper and also TVC labeling. The equations are given and the Existence Function '1' points are tabled along with the decimal index of the input variables. Both the Test Set derived by Kohavi and the Test Sequence are given. Note the use by the Test Sequence of two of the three tests given as equivalents in the Test Set, 0 and 4, ad the addition of 2 ($\underline{D}\underline{C}B\underline{A}$) as a test. (Underscore as negation)

BOOLE, an APL program which emulates the Boolean Analyzer generation of the Existence Function, has been used as a "check". Figure 12-14 presents the output generated by BOOLE when the input terms listed are used. Note that APL uses $\underline{E}$ (underscore for negation) and uses letters of the alphabet for the line labels rather than the xi (TNC labeling was not use din the program due to memory size limitation). The Test Sequence produced by this version is the same as before. (Figure 12-12)

The circuit is repeated with two others in Figure 12-13. Circuits a. and b. are equivalent when their Marquand maps are compared; Circuit c. is a complementary circuit.

The BOOLE output for the analysis of Circuit c. is shown in Figure 12-15. Note that all three circuits have the same Test Sequence. There and other examples support the idea proposed by Akers and others that equivalent circuits, i.e., various implementations of a function, can be tested by the same test set. For the circuits shown, it is hypothesized that complementary equivalents, i.e., the various implementations of the complement of a function, can also be tested by this Test Set.



Equations:

$x_4 = x_1 \quad x_8 = \bar{x}_3 \quad x_6 = \bar{x}_1 \quad x_5 = \bar{x}_0 \quad x_7 = \bar{x}_2$

$x_9 = x_0 + x_1 \quad x_{10} = x_2 + \bar{x}_3 + x_4 \quad x_{11} = x_5 + x_6 + x_7 + x_8$

$x_{12} = x_9 x_{10} x_{11}$

12	11	10	9	8	7	6	5	4	$\bar{3}$	2	1	0	index
0	1	0	0	0	1	1	1	0	0	0	0	0	8
0	1	0	1	0	1	1	0	0	0	0	0	1	9
1	1	1	1	0	1	0	1	1	0	0	1	0	10
1	1	1	1	0	1	0	0	1	0	0	1	1	11
0	1	1	0	0	0	1	1	0	0	1	0	0	12
1	1	1	1	0	0	1	0	0	0	1	0	1	13
1	1	1	1	0	0	0	1	1	0	1	1	0	14
0	0	1	1	0	0	0	0	1	0	1	1	1	15
0	1	1	0	1	1	1	1	0	1	0	0	0	0
1	1	1	1	1	1	1	0	0	1	0	0	1	1
1	1	1	1	1	1	0	1	1	1	0	1	0	2
1	1	1	1	1	1	0	0	1	1	0	1	1	3
0	1	1	0	1	0	1	1	0	1	1	0	0	4
1	1	1	1	1	0	1	0	0	1	1	0	1	5
1	1	1	1	1	0	0	1	1	1	1	1	0	6
1	1	1	1	1	0	0	0	1	1	1	1	1	7

Test sequence: 2-0-1-9-11-15-7-15-14-12-13-15-13-9-1-0-2

Test set: $\left\{ 14,\ 13,\ 11,\ 7,\ 1,\ 9,\ 15,\ \begin{matrix} 0 \\ 4 \\ 12 \end{matrix} \right\}$

Figure 12-12 An Example from Kohavi and Kohavi (see references)

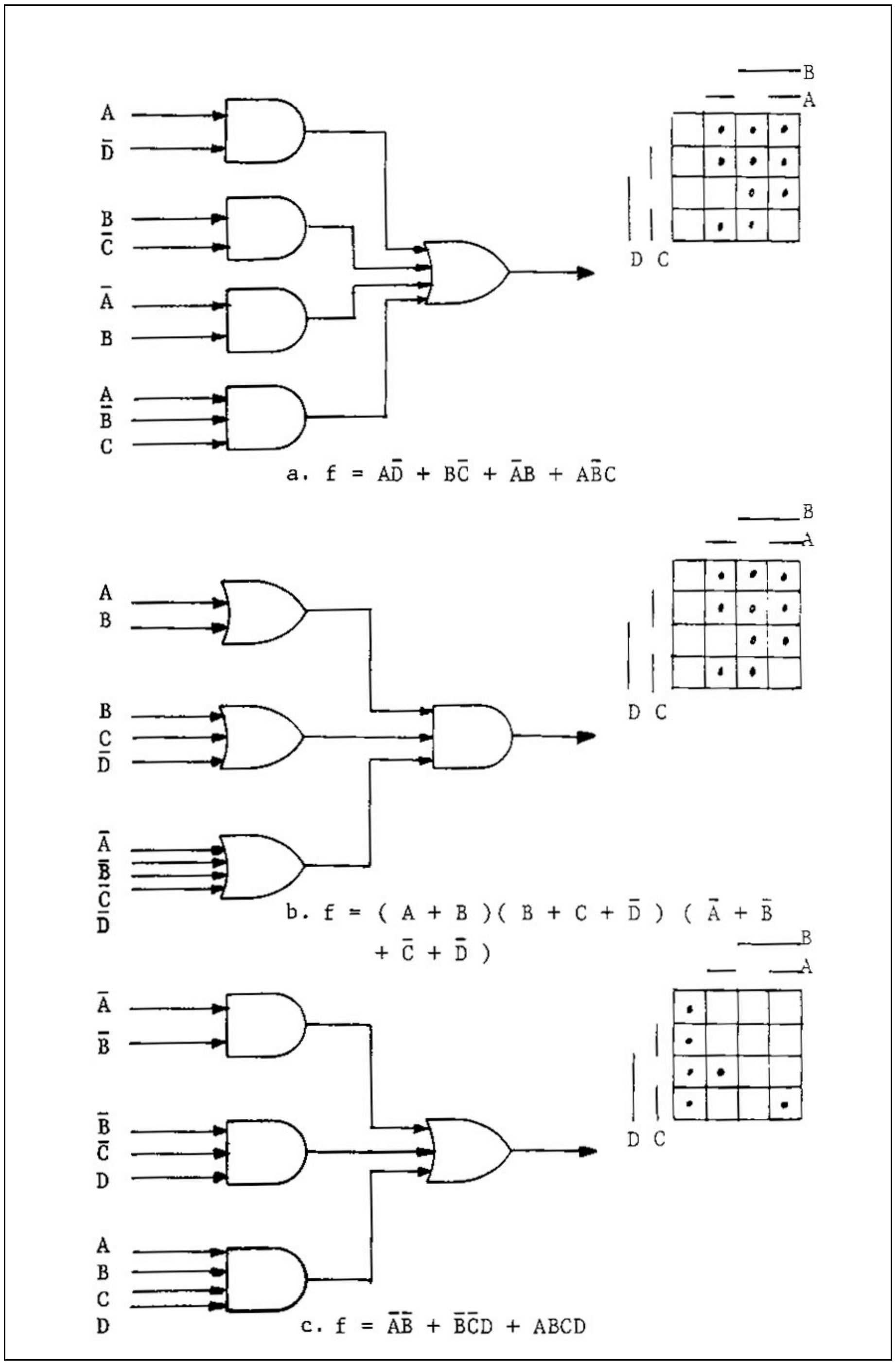
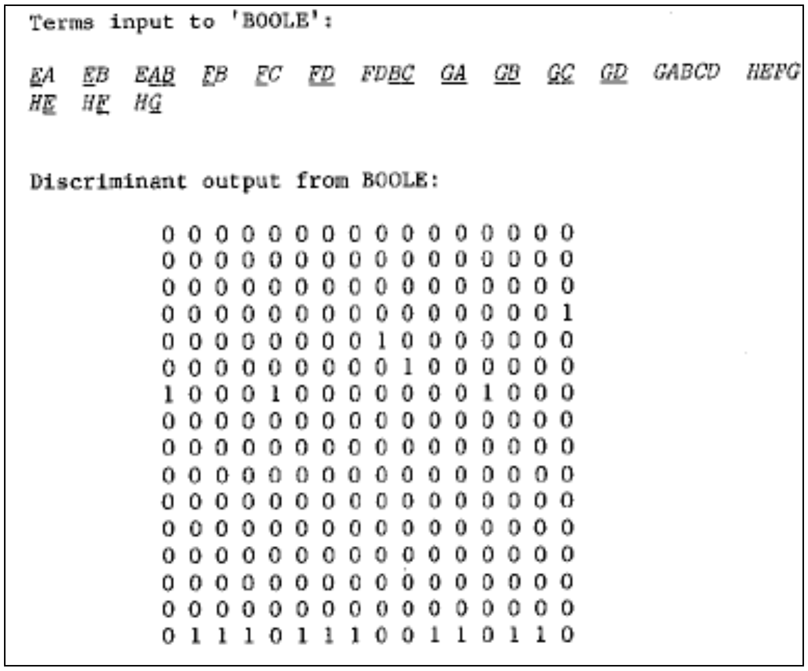
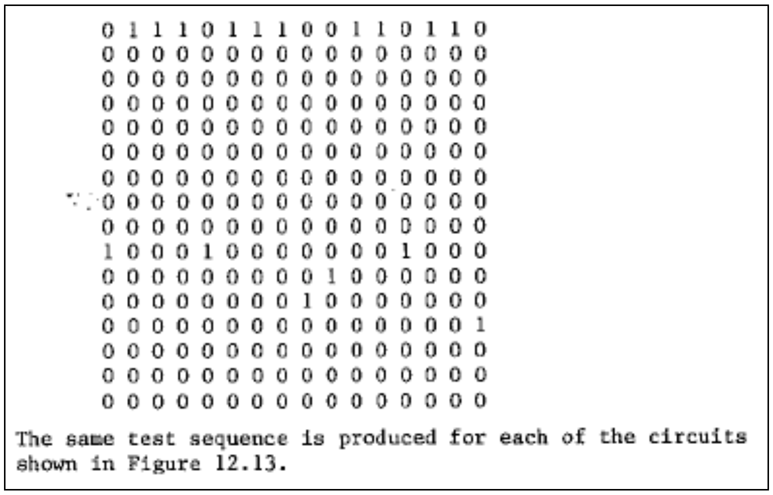

A
$\bar{D}$
B
$\bar{C}$
$\bar{A}$
B
A
$\bar{B}$
C
B
A
D C
a. $f = A\bar{D} + B\bar{C} + \bar{A}B + A\bar{B}C$
A
B
B
C
$\bar{D}$
$\bar{A}$
$\bar{B}$
$\bar{C}$
$\bar{D}$
B
A
D C
b. $f = (A + B)(B + C + \bar{D})(\bar{A} + \bar{B} + \bar{C} + \bar{D})$
$\bar{A}$
$\bar{B}$
$\bar{B}$
$\bar{C}$
D
A
B
C
D
B
A
D C
c. $f = \bar{A}\bar{B} + \bar{B}\bar{C}D + ABCD$

Figure 12-13 Three Example Circuits from Kohavi and Kohavi (see references)



```
Terms input to 'BOOLE':

EA   EB   EAB   FB   FC   FD   FDBC   GA   GB   GC   GD   GABCD   HEFG
HE   HF   HG


Discriminant output from BOOLE:

          0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
          0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
          0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
          0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
          0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
          0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
          1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0
          0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
          0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
          0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
          0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
          0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
          0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
          0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
          0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
          0 1 1 1 0 1 1 1 0 0 1 1 0 1 1 0
```

Figure 12-14 BOOLE Output for Figure 12-12



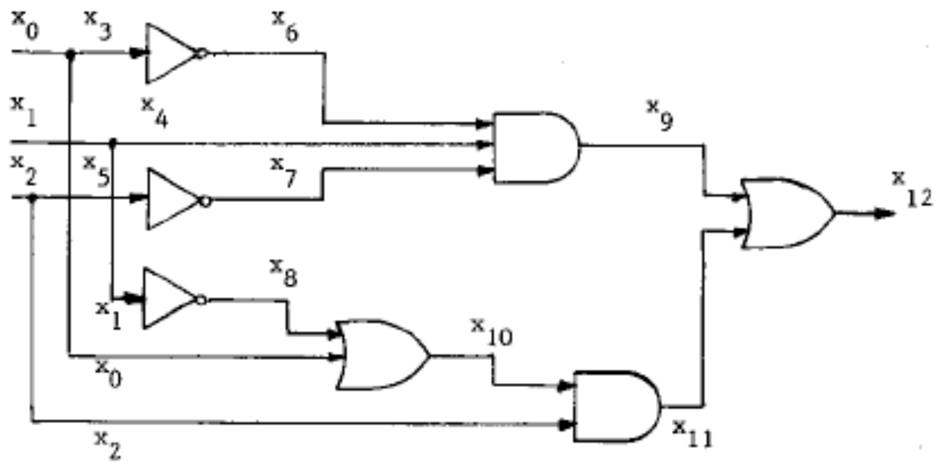
```
       0 1 1 1 0 1 1 1 0 0 1 1 0 1 1 0
       0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
       0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
       0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
       0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
       0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
       0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
       0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
       0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
       1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0
       0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
       0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
       0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
       0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
       0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
       0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

The same test sequence is produced for each of the circuits
shown in Figure 12.13.
```

Figure 12-15 BOOLE Output for Figure 12-13.c

12.3.5 Multiple Faults

An example from the paper by Yau and Tang on multiple faults appears in Figure 12-16. The Test Set they generated for multiple fault detection for this circuit is seen to contain the same points as appear in the Test Sequence. The Diagnostic Continuity Diagram for this circuit is given in Figure 12-17.

The classic approach to Test Set minimization of the fault Test Set minimization uses a Fault Table and treats the minimization of the fault test set as a coverage problem. (Note that the latest algorithm for the Boolean Analyzer is the Coverage Algorithm.) The fault table for the example is given in Figure 12-18. It should be noted as well that the Test Set thus generated omitted 7 as a necessary test.



$x_3 = x_0 \qquad x_4 = x_1 \qquad x_5 = x_2 \qquad x_6 = \bar{x}_3$

$x_7 = \bar{x}_5 \qquad x_8 = \bar{x}_1 \qquad x_9 = x_4 x_6 x_7$

$x_{10} = x_0 + x_8 \qquad x_{11} = x_2 x_{10} \qquad x_{12} = x_9 + x_{11}$

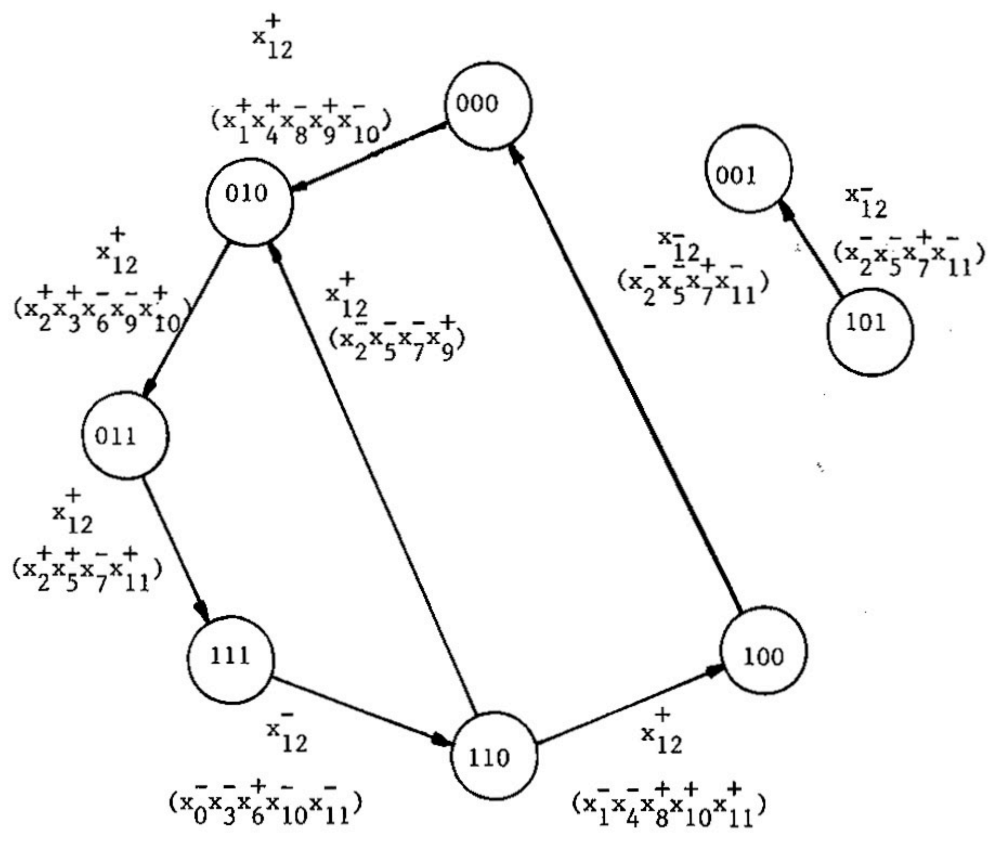
12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	1	1	1	0	0	0	0	0	0
0	0	1	0	1	1	0	0	0	1	0	0	1
1	0	0	1	0	1	1	0	1	0	0	1	0
0	0	1	0	0	1	0	0	1	1	0	1	1
1	1	1	0	1	0	1	1	0	0	1	0	0
1	1	1	0	1	0	0	1	0	1	1	0	1
0	0	0	0	0	0	1	1	1	0	1	1	0
1	1	1	0	0	0	0	1	1	1	1	1	1

links

Test sequence: 0-2-3-7-6-2-6-4-0

Test set for multiple fault detection:{ 0, 4, 2, 6, 3, 7 }

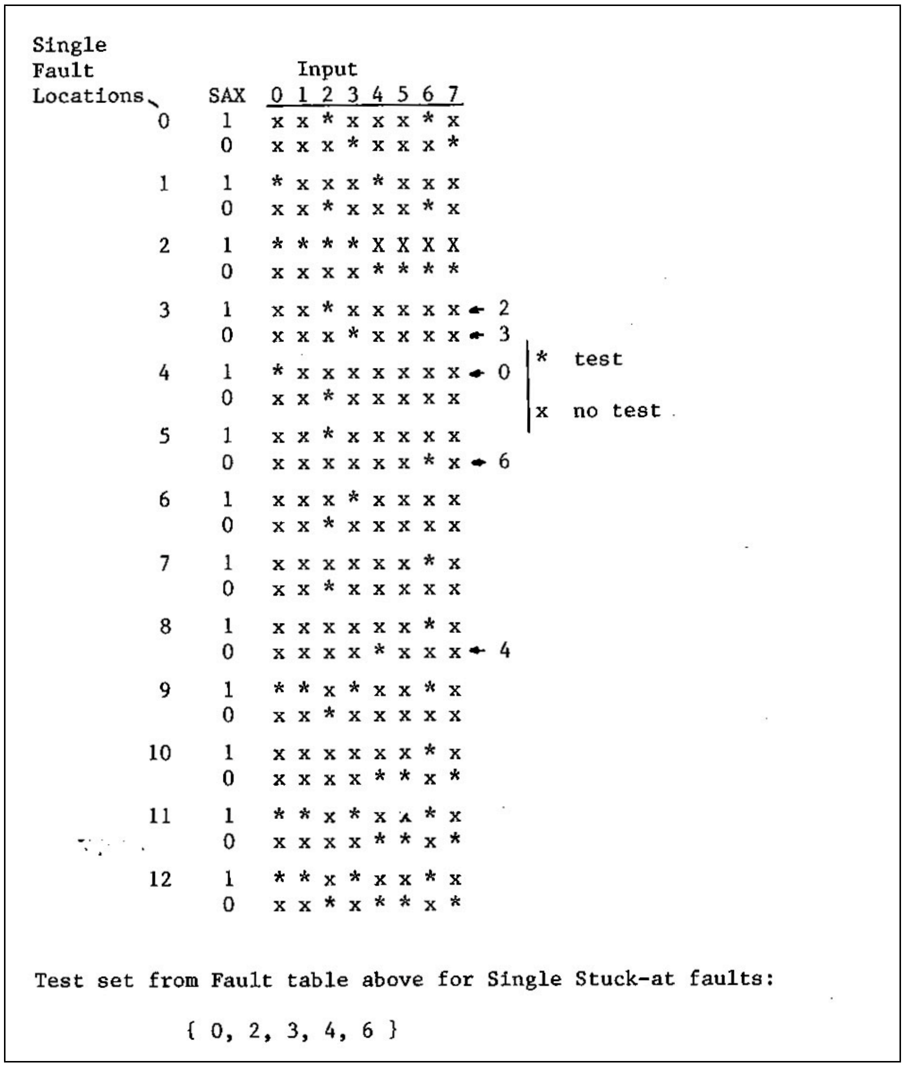
Figure 12-16 Multiple Fault Testing Example from Yau and Tang (see references)


000
010
011
111
110
100
001
101
x_{12}^{+}
$(x_1^{+}x_4^{+}x_8^{-}x_9^{+}x_{10}^{-})$
x_{12}^{+}
$(x_2^{+}x_3^{+}x_6^{-}x_9^{-}x_{10}^{+})$
x_{12}^{+}
$(x_2^{+}x_5^{+}x_7^{-}x_{11}^{+})$
x_{12}^{-}
$(x_0^{-}x_3^{-}x_6^{+}x_{10}^{-}x_{11}^{-})$
x_{12}^{+}
$(x_1^{-}x_4^{-}x_8^{+}x_{10}^{+}x_{11}^{+})$
x_{12}^{+}
$(x_2^{-}x_5^{-}x_7^{-}x_9^{+})$
x_{12}^{-}
$(x_2^{-}x_5^{-}x_7^{+}x_{11}^{-})$
x_{12}^{-}
$(x_2^{-}x_5^{-}x_7^{+}x_{11}^{-})$

Test sequence: 0-2-3-7-6-2-6-4-0

The links are shown in only one direction for simplicity.

Figure 12-17 Diagnostic Continuity Diagram for Figure 12-16



Single Fault Locations	SAX	Input 0	1	2	3	4	5	6	7	
0	1	x	x	*	x	x	x	*	x	
	0	x	x	x	*	x	x	x	*	
1	1	*	x	x	x	*	x	x	x	
	0	x	x	*	x	x	x	*	x	
2	1	*	*	*	*	X	X	X	X	
	0	x	x	x	x	*	*	*	*	
3	1	x	x	*	x	x	x	x	x	← 2
	0	x	x	x	*	x	x	x	x	← 3
4	1	*	x	x	x	x	x	x	x	← 0
	0	x	x	*	x	x	x	x	x	
5	1	x	x	*	x	x	x	x	x	
	0	x	x	x	x	x	x	*	x	← 6
6	1	x	x	x	*	x	x	x	x	
	0	x	x	*	x	x	x	x	x	
7	1	x	x	x	x	x	x	*	x	
	0	x	x	*	x	x	x	x	x	
8	1	x	x	x	x	x	x	*	x	
	0	x	x	x	x	*	x	x	x	← 4
9	1	*	*	x	*	x	x	*	x	
	0	x	x	*	x	x	x	x	x	
10	1	x	x	x	x	x	x	*	x	
	0	x	x	x	x	*	*	x	*	
11	1	*	*	x	*	x	x	*	x	
	0	x	x	x	x	*	*	x	*	
12	1	*	*	x	*	x	x	*	x	
	0	x	x	*	x	*	*	x	*	

* test

x no test .

Test set from Fault table above for Single Stuck-at faults:

{ 0, 2, 3, 4, 6 }

Figure 12-18 Fault Table Test Set Generation for Figure 12-16

12.4 Summary

Many other methods for test set generation exist in the literature. Most of these become unmanageable for large circuits.

The main objective of present research has been to find efficient programmable algorithms. Of the existing methods which were studies, Roth's D-Algorithm program set appears to be the most complete.

While the Test Sequence presented [herein] is programmable, it is also possible to produce the results using the Boolean Analyzer. The method is, therefore, a hardware solution to the fault detection problem.

The test sequence generation method described is intended for use with any multilevel, combinational circuit. It will also perform stable state test generation for sequential circuits; further research is needed to define a complete procedure for sequential circuits.

There has been sufficient indication from the results of the examples which have been studied to hypothesize that the generation of the Test Sequence does not require knowledge of the internal functional logic of a circuit. [Blackbox.]

The number of variables which may be handled is presently limited to the number of variables which the Analyzer is designed to process, i.e., twenty-two.

For circuits which fit the size restriction, a complete, minimal test sequence is produced.

References

Svoboda, Antonin. "Class Notes in Engineering 125A," Computer Science Department, School of Engineering and Applied Science, University of California, Los Angeles, 1966 and 1972.

________________. "Proposal for Research in Automated Design of Switching Circuits," UCLA-P-1482-N, May 1967.

________________. "Ordering of Implicants," IEEE Trans., EC-16(2):100-105, Feb. 1967.

________________. "Synthesis of Logical Systems of a Given Activity," IEEE Trans., EC-12(12):904-910, Dec. 1963.

________________. "Boolean Analyzer," Information Processing 68. North Holland, Amsterdam, 1969, pp. 824-830.

________________. "Parallel Processing in Boolean Algebra," IEEE Trans., C-22(9):848-851, Sept. 1973.

________________. "Boolean Analyzer Solutions of Fundamental Logic Design Problems," 1974 (unpublished).

________________. "Logical Systems and Spaces," Paper Distributed at the Symposium on Computers, Prague, Czechoslovakia, 1964.

________________. "Some Applications of Contact Grids," Proceedings, International Symposium on the Theory of Switching, Harvard U. Press, Cambridge, Mass., 1959, pp. 293-305.

________________. "Logical Instruments for Teaching Logical Design," IEEE Trans. on Educ., E-12(4):262-273, Dec. 1969.

________________. "The Concept of Term Exclusiveness and Its Effect on the Theory of Boolean Functions," JACM, 22(3):425-440, July 1975.

DeVries, Ronald C., and Antonin Svoboda. "Multiple-Output Optimization with Mosaics of Boolean Functions," IEEE Trans., C-24(8):777-785, Aug. 1975.

Marquand, Allan. "On Logical Diagrams for n Terms," Philosophical Magazine, V. 12, pp. 266-270, 1881(year).

Karnough, M. "The Map Method for Synthesis of Combinational Logic Circuits," AIEE., pp. 593-599, Nov. 1953.

Bearnson, L. W. and C. C. Carroll. "On the Design of Minimal Length Fault Tests for Combinational Circuits," IEEE Trans., C-20(11):1353-1356, Nov. 1971.

Bouricius, W. G., et al. "Algorithms for Detection of Faults on Logic Circuits," Digest of Papers of the 1971 International Symposium on Fault Tolerant Computing, Pasadena, CA, March 1971 (IEEE pub # 71-C-6-C), pp. 5-8.

Kohavi, Zvi, and DeWayne Spires. "Designing Sets of Fault Detection Tests for Combinational Logic Circuits," IEEE Trans., C-20(12):1463-1469, Dec. 1971.

Kohavi, Igal and Zvi Kohavi. "Detection of Multiple Faults in Combinational Logic Networks," IEEE Trans., C-21(6):556-568, June 1972.

Marin, Miguel A. "Investigation of the Field of Problems for the Boolean Analyzer," Computer Science Department, University of California, Los Angeles, UCLA-ENG-6828, June 1968 (PhD Thesis).

White, D. E. "Fault Detection Through Parallel Processing in Boolean Algebra," UCLA-ENG-7504, March 1975 (PhD Thesis).

__________. "Test Sequence Alternative to Fault Detection," Seminar notes presented at Cal. State Polytechnic Institute, October 1974.

White, D. E., and Antonin Svoboda. "Fault Detection in Combinational Circuits: The Test Sequence," Proceedings, The Eigth Asilomar Conference on Circuits, Systems and Computers, Pacific Grove, CA., Dec. 1974. Pages UKN.

McCluskey, E. J. "Minimization of Boolean Functions," Bell Syst. Tech. J., 35:1417-1444, 1956.

Marinos, Peter N. "Derivation of Minimal Complete Sets of Test Input Sequences Using Boolean Differences," IEEE Trans., C-20(1):25-32, Jan. 1971.

Sellers, F. F., and M. Y. Hsiao, and L. W. Bearnson. "Analyzing Errors with the Boolean Difference," IEEE Trans., C-17(4):676-683, July 1968, Correction, C-18(4):381, April 1969.

Yau, Stephen S., and Yu-Shan Tang. "An Efficient Algorithm for Generating Complete Test Sets for Combinational Logic Circuits," IEEE Trans., C-20(11):1245-1251, Nov. 1971.

Armstrong, D. B. "On Finding a Nearly Minimal Set of Fault Detection Tests for Combinational Logic Nets," IEEE Trans., EC-15(2):66-73, Feb. 1966.

Hsiao, M. Y., and Dennis K. Chia. "Boolean Difference for Fault Detection in Asynchrnous Sequential Machines," IEEE Trans., C-20(11):1356-1361, Nov. 1971.

Friedman, Arthur D., and P. R. Menon, Franklin F. Kho, ed. Fault Detection in Digital Circuits. Computer Applications in Electrical Engineering Series, Prentice-Hall, Englewood Cliffs, New Jersey, 1971.

Friedman, Arthur D. "Fault Detection in Redundant Circuits," IEEE Trans., C-16(2):99-100, Feb. 1967.

________________. "Diagnosis of Short Circuit Faults in Combinational Circuits," IEEE Trans., C-23(7):746-752, July 1974.

Avizienis, Algirard. "Fault Tolerant Computers: Theory and Design of Ultra Reliable (Fault Tolerant) Computers: Protective Redundancy, Diagnosis, Self Repair Techniques," Class Notes, UCLA Seminar ENG 298/ENG 867.7 , March 1971.

Howell, T. H. Design Criteria for Error Detecting and Error Correcting Codes in Memory Subsystems, (February 1972 PhD Thesis) to be published by Garland.

Basham, Gary R. "New Error Correcting Technique for Solid State Memories Saves Hardware," Computer Design, pp. 110-113, Oct. 1976.

Berger, Isreal, and Zvi Kohavi. "Fault Detection in Fan-out Free Combinational Networks," IEEE Trans., C-22(10):908-914, October 1973.

White, D. E., et al. "Introduction to Designing with the Am 2900 Family [bit-slice microprogrammed control devices]," Class Notes for AMD, Inc. Seminar ED2900A, June 1978.

Muroga, Saburo, and Hung Chi Lai. "Minimization of Logic Networks Under a Generalized Cost Function," IEEE Trans., C-25(9):893-907, Sept. 1976.