



Scholars' Mine

[Masters Theses](#)

[Student Theses and Dissertations](#)

1969

Fault diagnosis of sequential circuits

Virgil Willis Hughes

Follow this and additional works at: https://scholarsmine.mst.edu/masters_theses

 Part of the [Electrical and Computer Engineering Commons](#)

Department:

Recommended Citation

Hughes, Virgil Willis, "Fault diagnosis of sequential circuits" (1969). *Masters Theses*. 5357.
https://scholarsmine.mst.edu/masters_theses/5357

This thesis is brought to you by Scholars' Mine, a service of the Missouri S&T Library and Learning Resources. This work is protected by U. S. Copyright Law. Unauthorized use including reproduction for redistribution requires the permission of the copyright holder. For more information, please contact scholarsmine@mst.edu.

FAULT DIAGNOSIS OF SEQUENTIAL CIRCUITS

BY

VIRGIL WILLIS HUGHES, JR., 1945-

A

THESIS

submitted to the faculty of

171231

THE UNIVERSITY OF MISSOURI-ROLLA

in partial fulfillment of the requirements for the

Degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

Rolla, Missouri

1969

T 2272
c. I
31 pages

Approved by

James H. Daley

(advisor)

Lee J. Bain

Lee J. Bain

ABSTRACT

Compared to the problem of combinational network (CN) diagnosis, that of sequential network (SN) diagnosis has been an extremely difficult one. Present techniques of SN diagnosis are difficult to apply, and generally lead to lengthy test schedules or additional logic. This paper presents a new approach to the SN diagnostic problem which results in a substantially simpler technique than those found in the literature. The approach is to modify the SN so that it can be diagnosed from a combinatoric point of view. This is accomplished by the addition of outputs for testing purposes to certain lines in the circuit--no additional logic is required. The applicability of the technique is dependent on the density of stable states associated with the circuit, but attempts at finding a practical flow table whose circuit is undiagnosable by the method have been unsuccessful. Although test sequences were not the major concern of the investigation, the approach has resulted in almost minimal test sequences.

ACKNOWLEDGEMENT

The author is indebted to his advisor, Dr. James H. Tracey, for his encouragement, guidance, and time.

TABLE OF CONTENTS

	Page
ABSTRACT	2
ACKNOWLEDGEMENT	3
LIST OF FIGURES	5
I. INTRODUCTION	6
II. REVIEW OF CN TESTING	9
III. EVALUATION OF BASIC APPROACHES	12
IV. THE ALGORITHM	15
V. EXAMPLES	22
VI. CONCLUSIONS AND SUMMARY	29
BIBLIOGRAPHY	30
VITA	31

LIST OF FIGURES

Figure		Page
1	Fault table for NAND gate	9
2	SN and associated transition table	12
3	Modification of SN with additional primary input	13
4	SN for illustrating algorithm	15
5	Fault table for Figure 4	16
6	Example (1) for illustrating the algorithm	23
7	Stable state conditions for Figure 6	24
8	Fault table for Figure 7	25
9	Example (2)	28
10	Results of Step (1) for Figure 9	28

I. INTRODUCTION

Classical methods of diagnosis divide networks into two entities(1): 1) combinational networks(CN)--circuits whose outputs are determined completely by their present inputs, and 2) sequential networks(SN)--circuits whose outputs depend on previous and present inputs. Of the two, the SN has presented the more formidable diagnostic problem, and in comparison with the state of the art of CN diagnosis, that of SN diagnosis leaves much to be desired. Diagnostic procedures for SN's are in general difficult to derive, or the techniques lead to lengthy test schedules. Since in the computer industry high reliability and simple maintenance is stressed, a worthy investigation is the identification of simple diagnostic procedures which might be applicable to many of the practical sequential circuits.

It seems that the basic approach to the SN problem has been the design of checking experiments or test schedules which would determine from observation of input-output behavior whether the SN is operating correctly. When introduced by Moore(2), this approach resulted in little more than forcing the SN through its entire state diagram. Utilizing Moore's experiments, and assuming that a failure transforms a good machine into a different machine, Seshu and Freeman(3) has simulated the good machine and the machine resulting from each possible fault. A partitioning procedure is derived so that several faulty machines are tested at once. The concept of the distinguishing sequence was introduced by

Hennie(4). The distinguishing sequence permits a unique determination of the various states of the machine by inspecting the machine's response to the sequence. This concept has led to relatively good results for machines which possess distinguishing sequences. The distinguishing sequence approach is further developed by Kime(5). Finally, Kohavi and Lavallee(6) presents a method for designing SN's so that they will possess special distinguishing sequences which lead to short fault detection experiments.

In the above development, the SN has been considered essentially as a black-box, and the concept of testing for a given fault has not existed. Emphasis has been on the derivation and minimization of checking sequences. Although impressive strides have been made in the development from the work of Moore to that of Kohavi and Lavallee, the reduction in the length of test sequences as been at the expense of additional logic. The main emphasis of this investigation has not been centered on test sequences. Rather, the intention has been to derive substantially simpler techniques of SN diagnosis than those represented in the literature, while keeping any additional logic to a minimum. In this initial investigation, the main concern has been with the asynchronous SN whose memory characteristics result from feedback lines. The underlying philosophy has been to modify such a SN, already existing in the form of gate diagrams, so that it can be diagnosed from a combinatoric point of view. The approach, which has consequently resulted in very satisfactory test

schedules, was motivated by the relative ease of application and success of CN fault detection techniques.

In the next section elements of CN testing are reviewed. First, however, some terminology is defined. The following definitions by Roth(7) will be adopted:

failure--Any transformation of hardware that changes the logical character of the function realized by the hardware. In this paper only singly occurring stuck-at-one(s-a-1) and stuck-at-zero(s-a-0) line failures will be considered.

primary input--In a logical circuit, a line that is not fed by any other line in the circuit.

primary output--In a logical circuit, a line whose signal output is accessible to the exterior of the circuit.

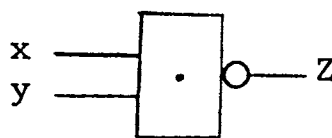
test for a failure--A pattern or set of patterns on primary inputs such that the value of the signal on some primary output will differ according to the presence or absence of that failure.

In addition, fault diagnoses will imply fault detection only. Signals are restricted to the level type. Other definitions will be given as the discussion progresses.

II. REVIEW OF CN TESTING

The fault table of Figure 1 represents a fault catalog of all s-a-1 and s-a-0 line failures of the NAND gate. x and y are the inputs; Z is the normal output. In each column of the subscripted variable columns, the output is given for the associated line with the failure designated by the subscript. For example, under x_1 the outputs are given for the various inputs when primary input x is s-a-1.

The circled entries are possible tests for the faults associated with the columns the circled entries are in. It is seen that for a given circled entry the output differs from the normal output according to the presence or absence of the associated fault. In fact, the input combinations $xy = (01, 10, 11)$ completely diagnose the line failures of the NAND gate. It is to be noted that not all of the possible input combinations are requisite for complete diagnoses. Similar analyses on the other conventional logic gates and on CN's indicate that this is true in general.



x	y	Z		Z_1	Z_0	x_0	x_1	y_0	y_1
0	0	1		1	0	1	1	1	1
0	1	1		1	0	1	0	1	1
1	0	1		1	0	1	1	1	0
1	1	0		1	0	1	0	1	0

Figure 1. Fault table for NAND gate

Sophisticated methods have been developed for deriving test schedules for CN's. One method which has been receiving attention is the D-algorithm(7). Based on the notation and calculus of D-cubes, the algorithm guarantees the computation of an existing test for a failure if the circuit is constructed from AND, NAND, OR, and NOR gates. The method involves intersecting primitive D-cubes of the circuit until a D-cube chain which extends from the primary inputs to the outputs is developed. For a given fault on a line the resulting chain is not unique, and another sequence of intersections may result in another representative D-cube chain for the fault. A procedure developed by Kautz(8) is based on the fault table concept. Kautz presents methods for simplifying the fault table to obtain minimal test schedules for the network. The advantage of the fault table is that all possible tests for a given fault are shown simultaneously. Because of this feature, the fault table is particularly useful in explaining the method of this paper, and will be used in favor of any modified version of the D-algorithm. Several other methods are in existence, each method having its specialized cases for which it is practically applicable. Basically the results of the various methods are the same; similar test schedules are generally derived.

The presence of a feedback line in a circuit greatly increases the complexity of the diagnostic problem, in that the response of the circuit to a given input set is not unique. This fact has been the bottleneck of the state of the art of

SN diagnosis.

Now, if the feedback lines could be intentionally locked at a constant binary value, the SN would appear essentially as a CN. This would encourage an attempt to apply CN diagnostic techniques to the network. In the next section the feasibility of various approaches to this possibility is discussed.

III. EVALUATION OF BASIC APPROACHES

The transitions of a SN can be described by an excitation matrix called the transition table, shown in Figure 2. Y , the next state variable, is mapped as a function of y , the present state variable, and x_1x_2 , the primary inputs. Circled entries are stable states. Z is assumed to be a function of the present state variable, as a consequence of the lumped circuit delay. The delay is not a physical element, and the symbol will not be shown on subsequent figures. Immediately it is seen that the response to a given input set is not unique, for the final output of the response depends on the initial output associated with the set.

Suppose another primary input, say x_3 , is associated with the circuit so that when $x_3 = 0$, the fault-free circuit behaves normally, but when $x_3 = 1$, all transitions are such that when the circuit stabilizes $y = Y$ for all x_1x_2 . In

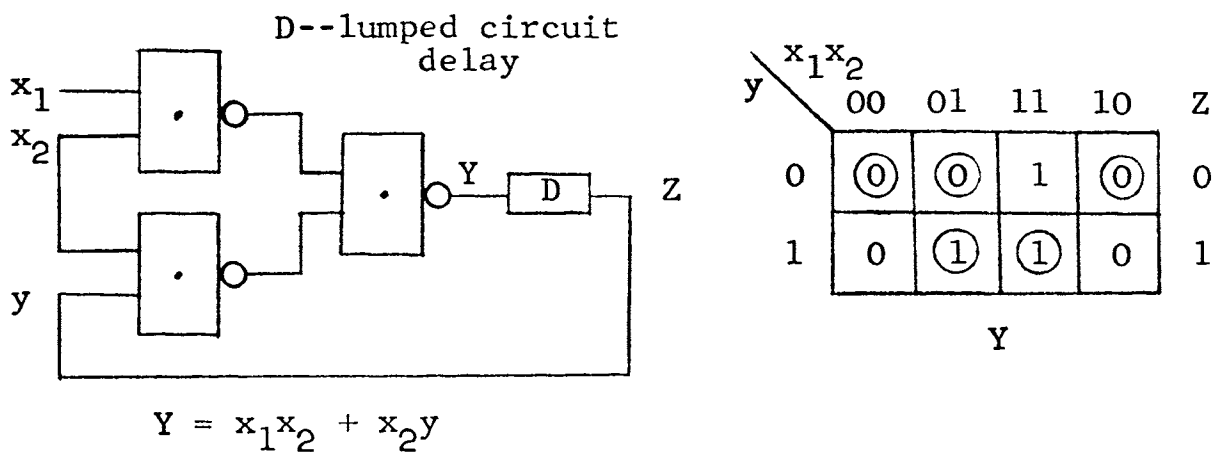


Figure 2. SN and associated transition table

Figure 3, when the circuit stabilizes to $y = Y = 0$, it is essentially locked with the feedback line at 0, and with the restriction that $y = 0$, the circuit appears as a CN. Similarly, the circuit may be locked at $y = 1$, or at either $y = 1$ or 0, depending on the values of x_1 and x_2 .

The major problem evolving from this approach is the diagnosis of extra logic resulting from the addition of x_3 . Although the states of the circuit under $x_3 = 1$ are chosen after consideration of the test requirements of the original circuit, the test requirements for the additional logic cannot be predetermined, and consequently the logic may not be diagnosable. Suppose that under $x_3 = 1$ the test mode is affixed such that all states are stable. Since all possible combinations of x_1x_2y are available as locked states, there is no problem in applying tests which require that $x_3 = 1$. Unfortunately, a test for a fault might require an unstable state under $x_3 = 0$. For example, if a test in the circuit

		$x_3 = 0$			
		x_1x_2 00	01	11	10
y	0	0	0	1	0
	1	0	1	1	0

		$x_3 = 1$			
		x_1x_2 00	01	11	10
y	0	0	0	0	0
	1	0	0	0	0

Figure 3. Modification of SN with additional primary input

associated with Figure 3 required the combination $x_1x_2x_3y = 1100$, the fault would avoid detection.

Now suppose another input, x_4 , is associated with the circuit, so that the combinations $x_3x_4 = 00, 01, 10, 11$ are available for test modes. Under one of these, say $x_3x_4 = 00$, the circuit behaves normally. x_3 and x_4 both assume values of 0 and 1 under test modes $x_3x_4 = 01, 10, 11$. As before, however, some faults may depend partially on $x_3x_4 = 00$ for detection. Summarizing, it seems that for this approach to be successful a fault detection test must not depend on the values of the extra inputs while the circuit is in the normal mode of operation.

It was noted earlier that in the case of CN diagnosis the set of required test patterns is usually a small subset of the totality of input patterns. Since many SN's have several stable states, some possible test patterns are inherent without modifications to the transition table. If optimum use were made of the stable states, most required test patterns might be available as a subset of the stable states. This is indeed the case for many circuits, and the next section presents an algorithm resulting from this approach to the SN diagnosis problem.

IV. THE ALGORITHM

The circuit of Figure 2 is given again in Figure 4, and will be used to elucidate the algorithm. Under stable state conditions, the circuit will be considered as a CN, with y as a primary input and $Z = Y$. For the present the delay is disregarded.

The five stable state conditions of the example are listed under columns x_1 , x_2 , y , and Z_c of Figure 5. Under Z_f are the values of the output for each line s-a-0 and s-a-1, the particular fault being designated by the subscript. The suggested technique for deriving the entries under Z_f is writing the output function as a function of the input variables and appropriate internal lines. For example, the entries for a_0 and a_1 are found from the equation

$$Z = \bar{a} + x_2 y.$$

It is seen that, according to the definition of a test

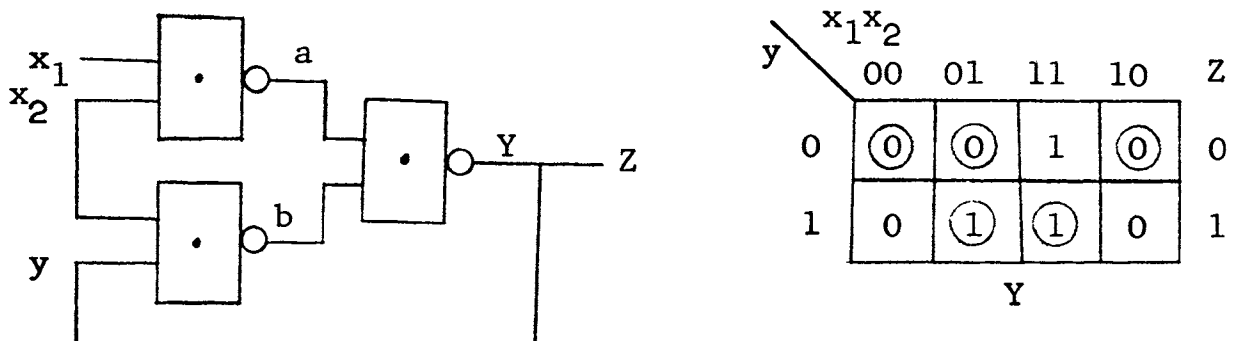


Figure 4. SN for illustrating algorithm

Z_F

x_1	x_2	y	z_c		$(x_1)_0$	$(x_1)_1$	$(x_2)_0$	$(x_2)_1$	y_0	y_1	a_0
0	0	0	0		0	0	0	0	0	0	①
0	1	0	0		0	①	0	0	0	①	①
1	0	0	0		0	0	0	①	0	0	①
0	1	1	1		1	1	①	1	①	1	1
1	1	1	1		1	1	①	1	1	1	1

Z_F

a_1	b_0	b_1	y_0	y_1
0	①	0	0	①
0	①	0	0	①
0	①	0	0	①
1	1	①	①	1
1	1	1	①	1

Figure 5. Fault table for Figure 4

for a failure, tests (the circled entries) presently exist for all faults except $(x_1)_0$ and a_1 .

Attention is now given to the equation

$$a = \bar{x}_1 + \bar{x}_2.$$

Under stable state conditions both a and x_1 assume values of 0 and 1. Furthermore, when x_1 is s-a-0 and the pattern $x_1x_2y = 111$ is applied, the value on a differs according to the presence or absence of the fault. If a is made an output, x_1 and a can be tested simultaneously. That is, if a testing output, Z_t , is added to the circuit at line a , the previously undetectable faults $(x_1)_0$ and a_1 can now be detected.

The general statement of the above results is the following theorem:

Theorem: Any circuit in which each line assumes values of both 1 and 0 under stable state conditions can be diagnosed for line failures, provided testing outputs can be added to those lines for which the primary outputs do not indicate the presence or absence of a given fault.

Clearly, if a fault on a line exists, a test for that fault must stimulate the value on the line to its complement. If under stable state conditions the primary outputs are independent of the fault, testing outputs are required to detect the fault.

The theorem is a sufficient, but not necessary condition for diagnosis. It is not difficult to imagine that a fault might cause the circuit to stabilize in a normally unstable state. This could lead to the detection of the fault, even though both binary values did not appear on every line

in the circuit under stable state conditions. If the conditions of the theorem are met, however, detection of all possible line faults is guaranteed.

A line is any input or output connection associated with the discrete AND, OR, NAND, NOR gates comprising the circuit. This does not preclude the diagnoses of more complex modules by the method, but the extension requires further research. When the conditions are met the method is also applicable to redundant circuits.

Because of their association with the same gate, x_1 and a can be tested simultaneously by utilizing the one testing output. Generally, testing outputs might have been placed at both lines.

Several assumptions have been made which are not stated explicitly. These are the following:

- (1) The circuit is operating in the fundamental mode (1); the inputs are level, and are never changed unless the circuit is in a stable condition;
- (2) The machine is strongly connected; all stable states are sequentially connected; and, as stated earlier,
- (3) Faults occur singly;
- (4) Each line in the circuit must assume both binary values under stable state conditions.

In conventional sequential circuit analysis the output is generally a function of the present state variables. In the algorithm outputs are considered as functions of the next

state variables. This is because for stable states the next state values equal the present state values. If a fault exists and a test for that fault is applied, the circuit may transit to the state indicated by the fault table, or another incorrect state, or the circuit may oscillate. Nevertheless, the fault will have been detected.

With the modifications resulting from the addition of the testing outputs, the circuit is diagnosed by proceeding from one testing stable state to another in any possible manner. Although it has been emphasized that test sequences are not the major concern of this paper, it is instructive to show a possible test sequence for the example. Assume the initial conditions are $(x_1x_2, Z) = (00, 0)$. Then a possible sequence for diagnosing the circuit is the following: $(00, 0)$, $(01, 0)$, $(00, 0)$, $(10, 0)$, $(11, 1)$, $(01, 1)$.

The algorithm is applicable to circuits with normal or nonnormal modes of transition, the basic requirement being a relatively high density of stable states. The probability of complete diagnosis increases as the density of stable states increases, and the probability is conditional on what stable states are available. An attempt to find a flow table with more than a stable state density of 0.5 whose circuit is undiagnosable has been unsuccessful; thus, the number of undiagnosable circuits seems to be small.

In the next section some examples are presented to illustrate more thoroughly the manual use of the algorithm. First, to provide a systematic approach, the steps of the

algorithm are reiterated. It is assumed that a circuit and its transition table are given. Signal values refer to stable state values.

- (1) From the stable state conditions, construct a table of values on each line under normal operation. This can be done by writing equations for internal lines in terms of primary inputs.
- (2) If a 1 and 0 appear on each line in Step (1), construct the fault table, simulating s-a-0 and s-a-1 faults on each line and considering the circuit as combinational logic; that is, the present state variables are considered as primary inputs, and the outputs are considered as functions of the next state variables.
- (3) If not all faults are detected on the primary outputs, group the lines which are not fully diagnosed according to logic levels. Add testing outputs to all undiagnosed lines in the logic level group nearest the primary outputs.
- (4) Modify the fault table of Step (2) to include the testing outputs of Step (3) as primary outputs.
- (5) If not all faults are detected on the original primary outputs or the testing outputs, repeat Steps (3) and (4) by adding testing outputs to the logic level group nearest the last logic level group considered. Repeat Steps (3) and (4) until all faults are detected on the original primary outputs or the

testing outputs.

It is entirely possible to add the first set of testing outputs to the logic level group nearest the primary inputs. This in general would require more testing outputs, since any output has a probability of fault-wise covering all previous logic levels.

The circuit can now be diagnosed by attaining the testing stable states in any efficient manner. Because of this fact, test schedules can be reduced to a near minimal.

If in Step (1) a 1 and 0 do not appear on each line the circuit is not diagnosable by this method.

V. EXAMPLES

In this section the algorithm will be applied first to the circuit of Figure 6.

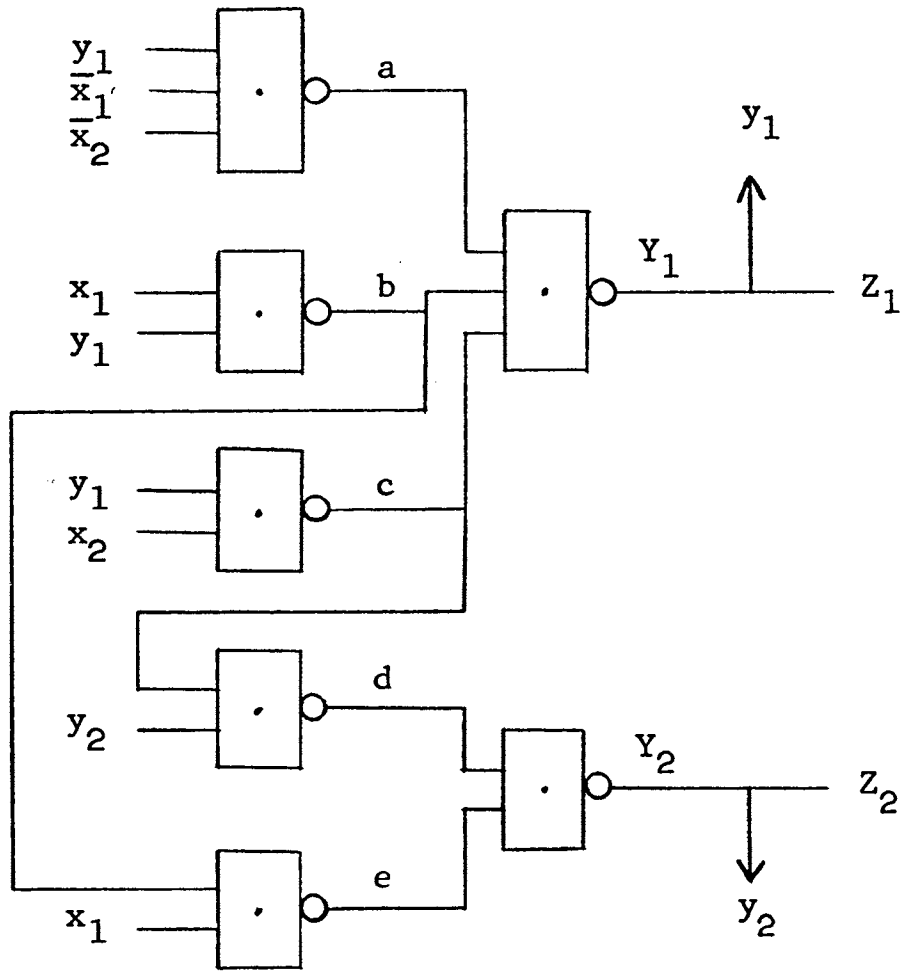
The stable state conditions are listed in Figure 7.

Since in Figure 7 a 1 and 0 appear on each line under stable state conditions, the conditions for proceeding to Step (2) are met. The results of Step (2) are given in Figure 8.

In the derivation of Figure 8, it is assumed that the pairs $x_1\bar{x}_1$ and $x_2\bar{x}_2$ are, fault-wise, four independent primary inputs. This is because the source of the complements is unknown. All other complemented variables are considered as true complements, since they are complemented by the gates, which are assumed to be fault free.

From Figure 8, it is seen that tests do not presently exist for $(x_1)_1$ and $(\bar{x}_1)_1$. To test for $(x_1)_1$ at line c, the pattern $y_1x_1 = 10$ is needed. This pattern does exist as a subset of one of the stable state conditions, and a testing output at c provides a means of testing for $(x_1)_1$. Similarly, for $(\bar{x}_1)_1$ the pattern $y_1\bar{x}_1\bar{x}_2 = 110$ is needed to detect the fault via line a. This pattern also exists in the set of stable state conditions, so a testing output is added at line a. If the desired patterns had not been in the cover of stable state conditions, the testing outputs would have been connected directly to lines x_1 and \bar{x}_1 .

With the testing outputs, the stable states actually required for diagnosing the circuit are $x_1x_2y_1y_2 = 0000$,



$$Y_1 = y_1 \bar{x}_1 \bar{x}_2 + x_1 y_1 + x_2 y_1 = \bar{a} + \bar{b} + \bar{c}$$

$$Y_2 = (\bar{y}_1 + \bar{x}_2) y_2 + (\bar{x}_1 + \bar{y}_1) x_1 = \bar{d} + \bar{e}$$

		$x_1 x_2$			
		00	01	11	10
$y_1 y_2$	00	00	00	01	01
	01	11	01	01	01
	11	11	10	10	11
	10	00	10	10	10
		$Y_1 Y_2$			

Figure 6. Example (1) for illustrating the algorithm

x_1	x_2	\bar{x}_1	\bar{x}_2	y_1	y_2		a	b	c	d	e		Z_1	Z_2
0	0	1	1	0	0		1	1	1	1	1		0	0
0	0	1	1	1	1		0	1	1	0	1		1	1
0	1	1	0	0	0		1	1	1	1	1		0	0
0	1	1	0	0	1		1	1	1	0	1		0	1
0	1	1	0	1	0		1	1	0	1	1		1	0
1	1	0	0	0	1		1	1	1	0	0		0	1
1	1	0	0	1	0		1	0	0	1	1		1	0
1	0	0	1	0	1		1	1	1	0	0		0	1
1	0	0	1	1	1		1	0	1	0	1		1	1
1	0	0	1	1	0		1	0	1	1	1		1	0

Figure 7. Stable state conditions for Figure 6

Primary Inputs						$(z_1)_F (z_2)_F$									
x_1	x_2	\bar{x}_1	\bar{x}_2	y_1	y_2	$(x_1)_0$	$(x_1)_1$	$(\bar{x}_1)_0$	$(\bar{x}_1)_1$	$(x_2)_0$	$(x_2)_1$	$(\bar{x}_2)_0$	$(\bar{x}_2)_1$	$(y_1)_0$	$(y_1)_1$
0	0	1	1	0	0	0 0	0 ①	0 0	0 0	0 0	0 0	0 0	0 0	0 0	① 0
0	0	1	1	1	1	1 1	1 1	⊙ 1	1 1	1 1	1 1	⊙ 0	1 1	⊙ 1	1 1
0	1	1	0	0	0	0 0	0 ①	0 0	0 0	0 0	0 0	0 0	0 0	0 0	① 0
0	1	1	0	0	1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	① 0
0	1	1	0	1	0	1 0	1 ①	1 0	1 0	⊙ 0	1 0	1 0	1 0	⊙ 0	1 0
1	1	0	0	0	1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	① 0
1	1	0	0	1	0	1 0	1 0	1 0	1 ①	1 0	1 0	1 0	1 0	⊙ 1	1 0
1	0	0	1	0	1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	① 1
1	0	0	1	1	1	⊙ 1	1 1	1 1	1 1	1 1	1 1	1 ⊙	1 1	⊙ 1	1 1
1	0	0	1	1	0	⊙ 0	1 0	1 0	1 ①	1 0	1 0	1 0	1 0	⊙ 1	1 0

Figure 8. Fault table for Figure 7

$(z_1)_F (z_2)_F$																		
$(y_2)_0$	$(y_2)_1$	a_0	a_1	b_0	b_1	c_0	c_1	d_0	d_1	e_0	e_1	$(Y_1)_0$	$(Y_1)_1$	$(Y_2)_0$	$(Y_2)_1$		$(z_1)_c$	$(z_2)_c$
0 0	0 ①	①0	00	①①	0①	①0	00	0①	00	0①	00	0 0	① 0	0 0	0 ①		0	0
1 ①	1 1	11	①①	①0	①0	①0	11	11	①0	11	11	① 1	1 1	1 ①	1 1		1	1
0 0	0 ①	①0	00	①①	0①	①0	00	0①	00	0①	00	0 0	① 0	0 0	0 ①		0	0
0 ①	0 1	①①	01	①0	00	①0	01	01	00	01	①①	0 1	① 1	0 ①	0 1		0	1
1 0	1 0	10	10	①①	①①	10	00	①①	10	①①	00	① 0	1 0	1 0	1 ①		1	0
0 1	0 1	①①	01	①0	01	①①	01	01	01	01	①①	0 1	① 1	0 ①	0 1		0	1
1 0	1 0	10	10	①①	①①	10	10	①①	10	①①	00	① 0	1 0	1 0	1 ①		1	0
0 1	0 1	①①	01	①0	01	①①	01	01	01	01	①①	0 1	1 ①	0 ①	0 1		0	1
1 ①	1 1	11	11	①0	①①	①0	11	11	①0	11	11	① 1	1 1	1 ①	1 1		1	1
1 0	1 ①	10	10	①①	①①	10	10	①①	10	①①	①①	① 0	1 0	1 0	1 ①		1	0

Figure 8. Fault table for Figure 7 (cont.)

0100, 0110, 1010. The circuit is diagnosed by attaining these states in any manner.

This section is closed with an example on which the algorithm is not applicable. Since attempts at finding such an example which is also practical have been unsuccessful, an example with only two stable states has been chosen.

The results of Step (1) for the circuit of Figure 9 are given in Figure 10. It is seen that some lines do not assume both binary values under the cover of stable states, and so the method is not applicable to this example.

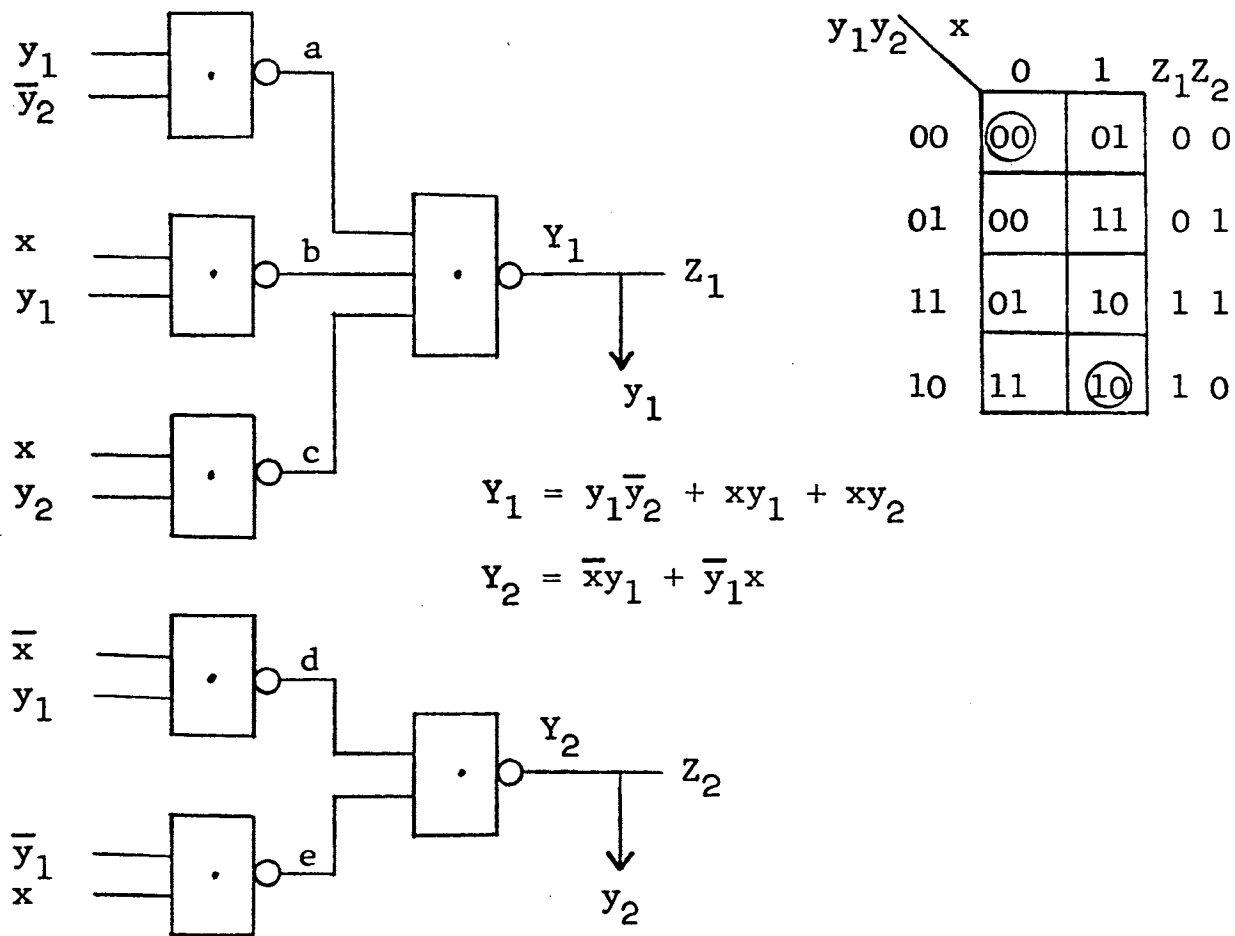


Figure 9. Example (2)

x	y ₁	y ₂	a	b	c	d	e	Z ₁	Z ₂
0	0	0	1	1	1	1	1	0	0
1	1	0	0	0	1	1	1	1	0

Figure 10. Results of Step (1) for Figure 9

VI. CONCLUSIONS AND SUMMARY

This paper has presented an algorithm for modifying a sequential circuit so that single faults can be detected using a combinatoric point of view. The effectiveness of the algorithm depends on the relative density of stable states.

The strongest restriction imposed on the SN is that a 1 and 0 must appear on each line under the cover of stable states. This restriction precludes some SN's, especially those with a light density of stable states, from being diagnosable by the method. The probability of diagnosis is conditional also upon what stable states are available, but the number of circuits which cannot be modified for diagnosis by the algorithm seems to be small.

The method now appears to have several advantages. The testing outputs are relatively cheap to implement, and the modification does not decrease the speed of the circuit. Since diagnostic tests depend only on certain stable states, the distance of transitions between the testing states can be nearly minimized; that is, the transitions can be stimulated in any efficient manner. In this way, the total test schedules can be reduced to a near minimal. Since a knowledge of the circuit implementation is required at the outset, it seems reasonable to expect that the algorithm might be extendable to include fault location.

BIBLIOGRAPHY

1. McCLUSKEY, E. J. (1965) Introduction to the theory of switching circuits. New York, McGraw-Hill, p. 180-218.
2. MOORE, E. F. (1956) Gedanken-experiments on sequential machines, in Automata studies. Princeton University Press, Princeton, New Jersey, p. 125-153.
3. SESHU, S. and FREEMAN, D. (1962) The diagnosis of asynchronous sequential switching systems. IRETEC, vol. EC-11, no. 4, p. 459-465.
4. HENNIE, F. C. (1964) Fault detecting experiments for sequential circuits. Proc. of the Fifth Annual Switching Theory and Logical Design Symposium, S-164, p. 95-110.
5. KIME, C. R. (1966) An organization for checking experiments on sequential networks. IEEEETEC (Short Notes), vol. EC-15, no. 1, p. 113-115
6. KOHAVI, Z. and LAVALLEE, P. (1967) Design of sequential machines with fault detection capabilities. IEEEETEC, vol. EC-16, no. 4, p. 473-484
7. ROTH, J. P. (1966) Design of automata failures: a calculus and a method. IBM Journal 10, p. 278-291.
8. KAUTZ, W. H. (1968) Fault testing and diagnosis in combinational digital circuits. IEEEETC, vol. C-17, no. 4, p. 352-366.

VITA

Virgil Willis Hughes, Jr. was born 7 February 1945 in Ironton, Missouri. He received his primary and secondary education in Leadwood, Missouri. He received a Bachelor of Science Degree in Physics from the University of Missouri-Rolla in 1968 January. Since that time he has been enrolled in the Graduate School of the University of Missouri-Rolla, and has held a research assistantship in the Electrical Engineering Department.

1968