REPORT R-381 MAY, 1968

CSL COORDINATED SCIENCE LABORATORY

OPTIMAL TEST SETS FOR THE DIAGNOSIS MULTIPLE OUTPUT COMBINATIONAL NETS

RICHARD D. ISENHART

UNIVERSITY OF ILLINOIS - URBANA, ILLINOIS

This work was supported in part by the Joint Services Electronics Program (U.S. Army, U.S. Navy, and U.S. Air Force) under contract DAAB-07-67-C-0199; and in part by NSF GK-1663.

OPTIMAL TEST SETS FOR THE DIAGNOSIS OF MULTIPLE OUTPUT COMBINATIONAL NETS

ABSTRACT

Given a set of tests and the faults which each of them partition, it is desirable to have a systematic method for selecting a subset of these tests which is minimum in number and which will isolate all distinguishable faults. Presented in this paper is a tabular method which selects an optimum set of diagnostic tests from the fault table of multiple output combinational nets. The procedure first removes undetectable faults and combines indistinguishable faults. Then it systematically removes redundant tests and tabulates most of the essential tests which must be used to isolate certain faults. From this reduced table the test combinations which isolate each fault are derived. A search of these test combinations produces the remaining essential tests which must be used for isolating certain faults. The essential tests are removed from the test combinations and are tabulated separately. Then the reduced test combinations are tabulated and Quine-McClusky minimization is used to select a minimum set of tests which diagnoses the remaining faults.

ACKNOWLEDGMENT

I wish to express my deep appreciation to Professor Gernot Metze whose patience, encouragement, and guidance will long be remembered.

I also wish to thank the members of the Switching Systems Group of the Coordinated Science Laboratory of the University of Illinois, in particular Mr. Anthony Wojcik, for the assistance I received. The typing of the manuscript by Miss Rosemary Swartz and Mrs. Sherry Kallembach is greatly appreciated.

TABLE OF CONTENTS

1. INTRODUCTION

With the increasing size and complexity of digital computers and with their expanding applications in such fields as telephone switching, time-sharing, and aero-space guidance has come a need for automated repair techniques.

The importance of reliability and serviceability of a system cannot be overlooked. Two basic methods are used to increase the reliability of digital computers. The first is the use of redundancy which extends the up-time of the computer. Redundancy is used in most real-time applications such as telephone switching systems and in spacecraft guidance where whole systems are duplicated. In general, redundancy is the addition of extra hardware to mask certain isolated faults.* It has been argued that because of this increase in hardware the availability of the system will abruptly decrease. This effect needs further investigation.

The second approach increases the availability of the system by increasing its serviceability. Automated test routines which detect and locate faults (diagnosis) are run at electronic speeds to reduce the system's down-time.

---

*An extensive bibliography can be found in an article by Robert A. Short, "The Attainment of Reliable Digital Systems Through the Use of Redundancy - A Survey" pp. 2-17, Computer Groups News, Volume 2, Number 2, March 1968.

If a fault has been masked by a redundant path then it cannot be detected by a diagnostic test. Much work needs to be done on the application of diagnosis to redundant circuitry. One possible approach would be to disable the redundant paths before running the diagnostic tests.

In diagnosis a set of tests is applied to the computer in the form of inputs and the outputs recorded and compared to the previously computed correct outputs of the good machine. If they differ a fault has been detected. How they differ is an indication of the location of the failure.

In this paper a method is presented which reduces the number of diagnostic tests needed to completely diagnose a combinational machine.

Before going deeper into this method, a few definitions and assumptions will be given.

DEFINITIONS:

Fault: For the purpose of diagnosis a fault shall consist of any physical defect in the hardware which causes a change in the logical character of the circuit.

Test: A test consists of the application of any signal or vector to the inputs of a logic net and the comparison of the output of the net with the known correct output. If the actual output and the correct output are different, then a fault has been detected.

Fault Signature: A fault signature is the response or output of a faulty machine to a fixed set of tests.

Fault Detection: Fault Detection, sometimes called "checkout" or "confidence test" on the system level, is the use of a test or set of tests to determine whether or not a logic net is operating properly; i.e., if the hardware performs the function it was designed to implement.

Diagnosis: Diagnosis or fault location is the isolation of faults. Fault diagnosis is an extension of fault detection.

Stuck-at-Zero (sa-0) and Stuck-at-One (sa-1): Abbreviated sa-0 and sa-1, the failure which causes a logic line to have a permanent logical value of zero or one respectively is called a stuck-at-zero or stuck-at-one fault.

Fault Dictionary: A fault dictionary is a lexicographical listing of fault signatures which correlates each fault signature with its corresponding faulty machine or machines.[10]

Simulation or Logic Simulation: Logic simulation is the process of generating the detailed logic net's structure in a general purpose computer. Using logic simulation one can compute the output of the good or faulty machine for any input vector.

For the remainder of this paper the following assumptions will be made:

1) The class of failures is finite and known,

2) The faults are well-behaved or non-transient and are logical in character,

3) Only single faults are present (this condition will be examined further in section 4),

4) The test procedure is combinational, i.e., the set of tests applied is fixed and unordered.

In general for fault diagnosis the first step is to derive a set of tests using any of several existing algorithms.[5-9] Then the good and faulty machines can be simulated on a computer, the test set applied to each machine, and the resulting outputs computed.[5,6]

Another method of obtaining the outputs under fault conditions is to use the actual machine and have faults physically inserted. Then the outputs can be recorded as the tests are applied.[11] Using either method, logic simulation or physical fault insertion, a table, called a fault table, can be constructed. In a fault table each test corresponds to a row and each fault corresponds to a column. For every test a mark (normally a "one") is placed under those faults which give an output different from a good machine.[1]

Many of the tests derived above are redundant and can be removed from the diagnostic test set without any loss of information. Using the fault table discussed above, optimum[1-3] and near-optimum[3] procedures have been developed for reducing the number of tests in the test set. The optimum procedures are restricted to machines

which have a smaller set of faults because the procedure must be exhaustive.

The methods above are developed around a fault table which contains only information as to which set of faults are detected by each test. When a test is applied to the machine several sets of faults often have different outputs or fault signatures.[3] Thus the test will not only detect all of these failures but will partition (or isolate) these sets of faults from each other. The purpose of this paper is to present a method for reducing the set of diagnostic tests using this added information.

Section 2 of this report discusses the details of the construction of a fault table which contains this added information and gives an example of the procedure to follow. The reader who is familiar with fault tables as presented by Kautz,[1] Chang,[2] Powell,[3] or Poage[4] should skip section 2 and go to section 3 which gives a detailed statement of the procedure. Section 4 discusses the restrictions and applications of this procedure and contains some concluding remarks and suggestions for further investigation.

2. A FAULT TABLE FOR MULTIPLE OUTPUT NETS AND AN EXAMPLE OF THE TEST REDUCTION PROCEDURE

In this section the detailed construction of fault tables for multiple output combinational circuits will be presented. This will be followed by an example of a matrix procedure which selects an optimum set of diagnostic tests. The reader who is familiar with fault tables can omit this section.

2.1 Fault Table

It is assumed that a set of tests has been derived and that the output vectors for the good machine and the faulty machines are known. From this information a fault table is prepared in which each fault is a column and each test is a block of consecutive rows. Each test is divided into a block of rows so that the sets of faults with different outputs can be tabulated separately. It is this point which causes this method to differ from most other methods of test reduction.

In each row of each test, 1's are placed in those columns whose faults are detected by that test and have the same fault signature. For a given test, no column in the block of rows corresponding to this test will have more than one 1 in it. This results from the assumption that all faults are well-behaved; therefore, for a fixed input, a fault can have only one fault signature.

Two more columns are added to the fault table for later reference. The first is a list of the input vectors for each test, and the second

lists the output vectors for each row. After test reduction these inputs and fault signatures are used to compile the fault dictionary.

2.2 Example

As an introduction to this matrix procedure and to illustrate the approach, a simple example will be given. Consider the full-adder in Figure 1.

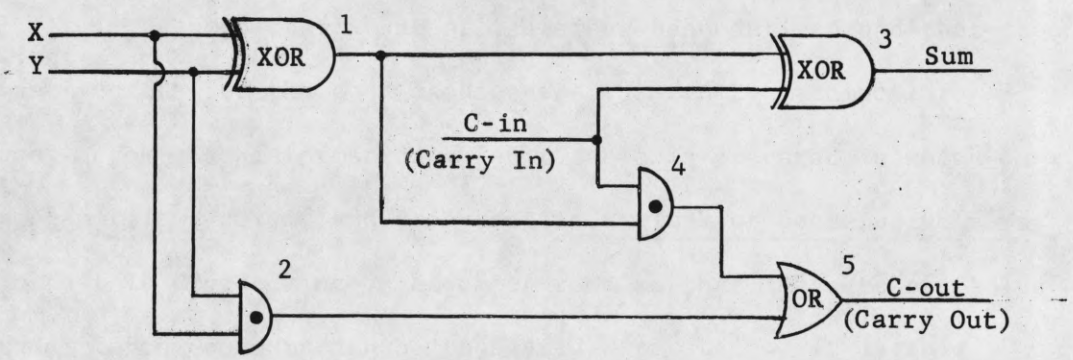


Figure 1. Full-Adder

It is desired to obtain an optimum set of tests which will diagnose this circuit for the set of single faults where the outputs of blocks 1 thru 5 are either stuck-at-one or stuck-at-zero. The outputs (Sum and C-out) resulting from the given faults were derived for all input combinations. These are tabulated in Table 1. The fault table (Table 2) is obtained directly from Table 1.

	Inputs			Correct Outputs		Output Resulting from Given Fault (Sum, C-out)									
Test #	X	Y	C-in	Sum	C-out	A	B	C	D	E	F	G	H	I	J
0	0	0	0	0	0	10	01	10	01	01					
1	0	0	1	1	0	01	11		11	11			00		
2	0	1	0	1	0		11		11	11	00		00		
3	0	1	1	0	1			11			10			00	00
4	1	0	0	1	0		11		11	11	00		00		
5	1	0	1	0	1			11			10			00	00
6	1	1	0	0	1	11		11				00			00
7	1	1	1	1	1	01						10	01		10

Where:

A = Line 1 SA - 1
B = " 2 "
C = " 3 "
D = " 4 "
E = " 5 "
F = Line 1 SA - 0
G = " 2 "
H = " 3 "
I = " 4 "
J = " 5 "

Table 1. Outputs Under Fault Conditions

Test # (Input)	Faults										Fault Output	Correct Output
	A	B	C	D	E	F	G	H	I	J		
0 (000)	1		1								1 0	0 0
		1		1	1						0 1	0 0
1 (001)								1			0 0	1 0
	1										0 1	1 0
		1		1	1						1 1	1 0
2 (010)						1		1			0 0	1 0
		1		1	1						1 1	1 0
3 (011)									1	1	0 0	0 1
						1					1 0	0 1
			1								1 1	0 1
4 (100)						1		1			0 0	1 0
		1		1	1						1 1	1 0
5 (101)									1	1	0 0	0 1
						1					1 0	0 1
			1								1 1	0 1
6 (110)							1			1	0 0	0 1
	1		1								1 1	0 1
7 (111)	1							1			0 1	1 1
							1			1	1 0	1 1

Table 2. Fault Table

This fault table contains all the information needed to diagnose the circuit for the assumed set of faults. For example, if test 3 (input = 011) is performed and the output is the correct output, (01), then it is known that the faults C, F, I and J have not occurred. If the output is (00) then we know that either the fault I or the fault J has occurred, if the output is (01) then fault F has occurred, and if the output is (11) then fault C has occurred.

Let us assume that test 3 is performed and the output is (00); another test has to be performed to distinguish between the faults I and J. Any test which gives different outputs for I and J can be used. Such tests are characterized in the table as having a row which has a 1 in either column I or J, but not both. In this example, both test 6 and 7 fulfill this requirement; and therefore when either is combined with test 3, the faults I and J will each be diagnosed.

To find the minimum number of tests which will isolate every fault, each fault will be examined individually to determine which test combinations partition it.

These test combinations will be tabulated and the smallest subset of them which isolates every fault will then be selected.

The most time consuming part of this procedure is the derivation of each fault's test combinations. Because of this, the procedure will incorporate several table reduction techniques which will reduce the fault table before performing the search for test combinations.

The first step in reducing the fault table is to remove any faults which are undetectable. In the example considered, there are no undetectable faults since every column has at least one 1 in it.

The next reduction in the table combines indistinguishable faults. If two or more faults always have identical fault signatures, they are said to be indistinguishable. Such faults are characterized by identical columns. In Table 2, columns B, D, and E are identical; and the three faults can be combined into one column. (See Table 3 where B' replaces B, D and E.)

The next two reductions in the fault table are interrelated. The first is the elimination of redundant tests. A test is said to be redundant if the faults which it detects are detected in another test and are partitioned into identical groups by both tests. If the fault patterns of two tests are identical, then either test can be eliminated. In Table 3, it is seen that tests 2 and 4 and tests 3 and 5 are identical. Therefore tests 4 and 5 are redundant and can be removed from the table. (See Table 4.)

The next reduction in the table is a result of certain tests being essential. If any fault has a single 1 in its column then that fault can only be partitioned from the good machine by that test which contains the 1. This test is an essential test and must be included in the final diagnostic test set. Therefore, any additional information contained in this test is obtained "at no extra cost."

Test #	Faults A	B'	C	F	G	H	I	J
0	1		1					
		1						
1						1		
	1							
		1						
2				1		1		
		1						
3							1	1
				1				
			1					
4				1		1		
		1						
5							1	1
				1				
			1					
6					1			1
	1		1					
7	1					1		
					1			1

B' = B, D & E

Table 3. Fault Table with Undetectable Faults Removed and with Indistinguishable Faults Combined

Test #	Faults A	B'	C	F	G	H	I	J
0	1		1					
		1						
1						1		
	1							
		1						
2				1		1		
		1						
3							1	1
				1				
			1					
6					1			1
	1		1					
7	1					1		
					1			1

B' = B, D & E

Table 4. Fault Table with Redundant Tests 4 & 5 Removed

In Table 4, column I has a single 1 which occurs in a row of test 3. Therefore test 3 is the only test which will detect fault I and is therefore an essential test. Since in the second row of test 3 there is a single 1 which occurs in column F, test 3 isolates fault F and column F can be removed from the table along with the second row of test 3. In like manner C is isolated by the third row of test 3, and this row and column can be removed from the reduced fault table. The new reduced table is shown in Table 5.

At this point there can be new redundant tests in the reduced fault table; in general they can be removed. The details of the restrictions on the removal of redundant tests will be explained in section 3; at this point an approach which does not incorporate the removal of further redundant tests will be applied to this example.

All of the table reduction techniques employed above are not mandatory in the sense that the number of tests chosen for the final diagnostic test set is independent of these reductions. The following procedure for deriving the test combinations which isolate each fault individually can be applied to any fault table whether reduced or not. In either case, the application of a minimal covering procedure will always result in the same number of tests in the diagnostic test set. If no table reductions are made, then in general there will be a larger number of solutions all with the same minimal number of tests.

Returning to the example, all of the test combinations which isolate each fault will be derived directly from Table 5. Starting

B' = B, D & E

Test 3 is Essential
Output 10→F
Output 11→C

Test #	Faults A	B'	G	H	I	J
0	1					
		1				
1				1		
	1					
		1				
2				1		
		1				
3					1	1
6			1			1
	1					
7	1			1		
			1			1

Table 5. Fault Table with Those Columns Removed Which Are Isolated by an Essential Test

with fault A, a search is made for tests with 1's in column A. Test 0 contains a row with a single 1 in column A. Therefore test 0 will isolate A from all other faulty machines. Also test 1 and test 6 will each isolate A. Test 7 contains the only other 1 in column A, but in that row there is another 1 in column H. Test 7 alone does not isolate A and a search must be made for tests which will partition A from the remaining faults. Since only fault H remains unpartitioned from A, only its column need be searched for tests which detect it and partition it from A. Test 1 is such a test, but it is already included in the list and need not be considered. Test 2 isolates H from A and is not a previous member of the list, therefore test 7 and test 2 together will isolate A from all possible faulty machines and the good machine. This completes the table for fault A. The test combinations which isolate A are listed below:

Fault A:	0
	1
	6
	7, 2

In the final minimal covering procedure at least one of these test combinations must be used; and since the use of any one of them ensures that all other faults will be distinguishable from A, column A can be removed from the table (Table 6).

From Table 6, the test combinations which isolate fault B' (B' = B, D, and E) are derived using the same approach as before.

Test #	Faults B'	G	H	I	J
0	1				
1			1		
	1				
2			1		
	1				
3				1	1
6		1			1
7					
		1			1

Table 6. Fault Table with Column A Removed

These test combinations are listed below:

Fault B' :	0
	1
	2

Now column B' can be removed from the table and the test conditions for fault G derived. This process is continued until every fault has been considered. The results are listed below:

Fault	Test Combination
A	0
	1
	6
	7, 2
B	0
	1
	2
G	6, 3
	7, 3
H	1
	2
	7
I	3, 6
	3, 7
J	3
	6
	7

From this list a table is made in which the faults are again columns, but now the rows are the test combinations listed above. In each row a 1 is placed under those columns which are isolated by that test combination (Table 7).

The test combinations which contain more than one test can isolate more faults than are listed in Table 7. These additional faults are merely those faults isolated by the individual test (or set of tests) of that test combination. These additional faults are included in Table 8. The cost figure in Tables 7 and 8 is the number of tests in the test combination (excluding essential tests). Quine-McClusky minimization or any other prime implicant covering procedure is used to find the set of test combinations which contains the least number of tests necessary to isolate every fault.

From Table 8 it is seen that five minimum covers exist. They are (1,6), (2,6), (7,0), (7,1) or (7,2). All have a cost of two and will diagnose all distinguishable faults with two tests, in addition to the previously determined essential tests. In this example only test 3 is essential. Notice that of the minimal covers the combination (1,6) tests a particular fault twice, i.e., it has overlap on two 1's in a column (column A). This also occurs with column H when tests 7 and 2 are used. If this added checking power is desired, then the test combination 1, 3, and 6 would be desirable. Table 9 lists the tests and the fault signatures that would result from each fault. This information can be obtained directly from Table 2.

The procedure presented in this example will be formalized in the next section.

Test Combination	Faults A	B'	G	H	I	Cost = No. of Tests
0	1	1				1
1	1	1		1		1
2		1		1		1
6	1		1		1	1
7			1	1	1	1
7,2	1					2

Table 7. Test Combinations

Test Combination	Faults A	B'	G	H	I	Cost
0	1	1				1
1	1	1		1		1
2		1		1		1
6	1		1		1	1
7			1	1	1	1
7,2	1	1	1	1	1	2

Table 8. Test Combinations

Table of Fault Signatures	Outputs (Sum, C-out) Test 1 (001)	Test 3 (011)	Test 6 (110)
Good Machine	1 0	0 1	0 1
Faulty Machines: A	0 1	0 1	1 1
B'	1 1	0 1	0 1
C	1 0	1 1	1 1
F	1 0	1 0	0 1
G	1 0	0 1	0 0
H	0 0	0 1	0 1
I	1 0	0 0	0 1
J	1 0	0 0	0 0

Table 9. Fault Signatures

Presented in this section is an algorithm for the selection of an optimum set of diagnostic tests. It is assumed that a fault table has been prepared in which each fault is a column and each test is a block of rows. Each row contains 1's under those faults detected by the same output vector.

3.1 Undetectable and Indistinguishable Faults

To begin with, much useful information can be obtained from the fault table by inspection. Faults which are undetectable or indistinguishable can be readily determined and tabulated separately. An undetectable fault is a fault whose column contains no 1's. A quick search of the fault table for any blank columns produces all such faults.

If two or more faults have identical columns then they are indistinguishable since there does not exist a combination of tests that will partition them.

Indistinguishable faults can be automatically located by forming the EXCLUSIVE-OR of every pair of columns (row by row) and testing each result for all zeros. Possible short cuts to this procedure exist. For example, when a pair of faults is found to be indistinguishable, one of the columns can be removed from the table thereby reducing the number of iterations in the procedure.

Another short cut would be to first form the row by row EXCLUSIVE-OR of only those pairs of columns which have 1's in the first row of the first test and test each result for all zeros. This is then repeated for all columns which have 1's in the second row of the first test, again checking for a result of all zeros after each ring-sum. This process is continued thru all remaining rows of the first test. Then the entire procedure is repeated for the second test with the exception that those columns which have previously been examined are excluded. This process can again be repeated for the next test and so on, until all columns have been examined. If desired the process can be terminated after any row. Then, using only those columns which have not been previously examined, the EXCLUSIVE-OR of every pair of columns must be formed and each tested for all zeros. Either strategy will greatly reduce the number of iterations of the procedure at the cost of only a small amount of memory.

The knowledge of which faults are indistinguishable or undetectable is extremely useful in test generation methods as will be explained later in section 4. The importance of combining indistinguishable faults at this point in the procedure will be examined later in this section.

3.2 Essential Tests

Still another reduction in the fault table can be made by inspection. Just as undetectable faults are characterized by columns with no 1's in them, certain essential tests are characterized by columns containing single 1's.

An essential test is a test which if removed from the fault table would cause two or more faults to become indistinguishable from each other or cause one or more faults to become undetectable.

Consider any column which contains a single 1. If the test which contains that 1 were removed from the table the corresponding fault would then become undetectable, therefore such a test is essential. Note that only a certain class of essential tests are characterized by columns containing single 1's, i.e., those which if removed would result in additional undetectable faults. The other class of essential tests is characterized by two columns being identical in all but one test. If that test were removed from the table the two columns would then become identical and thus indistinguishable. Therefore the test in which the two columns differ is an essential test.

Note that this concept of essential tests is different from the concept of essential rows in a prime implicant cover, in that the purpose here is not to find a minimal "cover" for the fault table, but to find first the test combinations for partitioning

columns of the fault table and then find the minimal cover of these tests.

Since an essential test must be included in the diagnostic test set, all of the information about the faults which it partitions is obtained at "no extra cost". If an essential test contains a row with a single 1 in it, the corresponding faults are isolated by that row. Thus, that column and row can be removed from the fault table. It is obvious that this reduction cannot affect the detectability or distinguishability of faults.

3.3 Redundant Tests

Another important reduction in the fault table is the elimination of redundant tests.

Let $a_1, a_2, \ldots, a_m$ be the row vectors of a test A. Test A is said to be redundant if there exists a test B with row vectors $b_1, b_2, \ldots, b_n$ $(n \geq m)$, such that for every a_i in A there exists a set of b_j's in B such that $a_i = Ub_j$ where Ub_j is defined to be the union of the row vectors in the set b_j.

Note that by definition an essential test cannot be redundant. In a reduced fault table a previously determined essential test may "appear" to be redundant. This can occur if a column reduction results in the removal of the fault which "required" that test to be essential.

If a test A is redundant, then it can be removed from the fault table without any loss of information since every set of faults which is partitioned by test A is partitioned by test B. If $m = n$, then test A is identical to test B and either A or B, but not both, can be removed from the fault table. (The remaining test might now become essential.) If $m < n$, then A, and not B, must be removed because B has more rows and thus contains more information than test A.

The most direct approach for finding redundant tests is to compare every pair of tests row by row. If test A is not essential and if every row in test A either has an identical row in test B or is exactly equal to the union of a group of rows in B, then test A is redundant and can be removed from the table. This procedure can easily be programmed on a digital computer and short cuts used to reduce the number of rows considered.

It is obvious that the removal of a redundant test cannot cause an undetectable fault to become detectable or cause a detectable fault to become undetectable. Also, if a set of faults is distinguishable (indistinguishable) before a redundant test is removed from the table, then they will be distinguishable (indistinguishable) after the test is removed, because for each row removed from the table, there exists another row to preserve each column's distinctness. It could also be shown that the removal of undetectable and indistinguishable faults cannot affect the redundancy of tests. Therefore,

redundant tests can be removed before or after undetectable and indistinguishable faults are removed. Whichever reduction results in the most simplification in the fault table should be performed first. In circuitry which is assumed or known to have a large number of undetectable or indistinguishable faults, column reduction followed by row reduction would be the more economical approach.

When redundant tests are found and subsequently removed from the fault table, it is possible for new essential tests to be formed. If columns are removed from this table because of these essential tests, it is possible for new redundant tests to be formed. Again when these are removed from the table, new essential tests may be generated. This chain will terminate with either a complete diagnostic test set containing all essential tests, or a fault table free of redundant tests and essential tests. When testing for new redundant tests, the only tests which need be examined are those which had 1's in columns which were isolated by an essential test and consequently removed from the table. Of these tests only those which are not essential have to be examined since a redundant test cannot be essential by definition.

Note that the concept of redundancy can easily be extended to a group of m tests being redundant to a group of n tests where $m \geqslant n$. The algorithm presented here does not include a check for the redundancy of groups of tests because to examine all combinations of two or more tests can be quite a formidable task.

Returning to the example of section 2, and examining Table 5, the rows of test 1 are found to include the rows of both test 0 and test 2. Hence tests 0 and 2 are redundant and can be removed. The resulting table (Table 10) has new essential tests. Test 1 is now essential due to the single 1 in column B'. Columns A, B', and H can be removed from the table since they are isolated by rows of test 1. The resulting fault table is shown in Table 11. In this table it can be seen that tests 6 and 7 are identical and either of them can be removed from the table. Table 12 has test 7 removed. Now test 6 has become essential, and the solution for the remaining tests is trivial. Note that test 3, which is essential due to column I, cannot be removed because fault I is not distinguished from fault J by test 3 alone. The final optimum solution is tests 1, 3, and 6, which is identical to the solution given in section 2.

3.4 Search for Test Combinations

At this point all of the undetectable faults have been found and recorded; all of the indistinguishable faults have been found, tabulated and combined into one column under one fault heading; all of the redundant tests have been removed from the original fault table; and several essential tests have been found, and the faults

Test #	Faults A	B'	G	H	I	J
1				1		
	1					
		1				
3					1	1
6			1			1
	1					
7	1			1		
			1			1

Table 10. Reduced Fault Table

Test #	Faults G	I	J
3		1	1
6	1		1
7	1		1

Table 11. Reduced Fault Table

Test #	Faults G	I	J
3		1	1
6	1		1

Table 12. Reduced Fault Table

isolated by each have been recorded. What is now needed is a procedure which will systematically find all combinations of tests which will isolate each fault. These test combinations will then be tabulated, and the smallest subset of them which will isolate every fault will be chosen and used as the final product--an optimal diagnostic test set. Any procedure which derives all test combinations which isolate each fault must have two basic properties. First it must be exhaustive in order to ensure optimality. Second it must be able to examine a combination of tests to determine whether or not the addition of a particular test aids in the isolation of a given fault. This second property is obtained with the use of a "test intersection" procedure which will be explained later in this section. Optimality is ensured in the exhaustive approach presented below.

In this approach, each fault is individually examined. The fault table is systematically searched for test combinations which will isolate the first fault. Each test is checked in succession to see if it detects the fault, and each test which detects the fault is examined to see if it isolates the fault. If it does not, a search must be made for all possible test combinations which partition the first fault from the remaining faults. This process is repeated until all the test combinations which isolate the first fault have been found. The column corresponding to this

fault can now be removed from the fault table since the use of any one of these test combinations ensures that all other faults in the table will be distinguished from it. With this column removed it is possible for some tests to become redundant and thus removable from the table. Again new essential tests may be present after the redundant tests are removed. After all such reductions are made the entire procedure is repeated for each of the remaining faults, each time removing the column after its test combinations are tabulated and then removing any new redundant tests and faults isolated by essential tests.

The flow table of this procedure is presented in the appendix and should be self-explanatory. Basically the procedure searches columns in the fault table from top to bottom, generating a decision tree of test combinations as it proceeds. On each branch of the tree is placed the number of the last test chosen. To generate the test combination used to reach any node of the tree, one has only to trace the tree back to the origin. At each node there is placed an r-tuple, where r is the total number of columns in the reduced table. This vector contains 1's in those positions which correspond to the faults which have not yet been partitioned from the first fault. This r-tuple is called the "test intersection" or just "intersection" (the notation NINT(n) used in the appendix corresponds to the "New INTersection").

The test intersection for each node is formed by starting with an r-tuple of all 1's and then setting to zero those positions corresponding to faults which have been partitioned from the first column. A short algorithm is used to compute this new intersection for each node.

Each row of the test to be combined with the last intersection is examined for a 1 in the first column, and those rows which contain 0's in the first column are inverted. Each row is then ANDed with the previous test intersection. The result is the "new intersection" (NINT(n)).

This test intersection contains information as to which columns should be searched for tests which further partition the first fault. To ensure that the procedure is exhaustive every column which has a 1 in the intersection must be searched. When a test is found and combined with the intersection at the previous node, the result is first checked to determine if it gives any new information. If it does, then it is checked to see if it isolates the first fault. If the test does not partition any new columns from the first column it is discarded and the search continued. If it does give new information and if it isolates the first column then a test combination for that fault has been found. The tree is traced back to its origin, outputing the tests stored at each branch. If the test gives added information but does not isolate the first

fault then this new intersection is stored at the new node of the tree and the search resumed.

The above search is continued until every branch terminates in a test combination which isolates the first fault.

Since all indistinguishable faults have been previously removed, it is impossible for a path to include every test and still not isolate the fault. If it were to happen that the search used every test and still had an intersection which contained multiple 1's then those columns which contained the 1's would be indistinguishable and should be combined. If in programming this procedure it is found that the initial removal of indistinguishable faults "costs" more than computing the intersection of every test for every group of indistinguishable faults, then their removal should be postponed and the search routine allowed to find them. This will not be more practical when the fault table has a large number of tests and a small number of faults.

The search for test combinations for each fault is greatly reduced by placing all of the essential tests at the top of the fault table after each table reduction. Since the search routine starts at the top of the table the essential tests will be used first. When the first test combination is found it is checked to see if it contains all essential tests. If it does then this fault is isolated "free" and the search for more test combinations for that fault can be terminated.

The flow table presented in the appendix which illustrates the above search routine is not intended to be the "best" approach to the implementation of the procedure but is included merely as an aid in understanding the mechanics of the search.

Note that if the undetectable faults were not previously removed from the table, then they would be found by the search routine.

If the essential tests were not removed from the fault table and the test combinations for a fault which required some essential tests were generated, then the essential tests would be those tests which are present in every test combination. Also, if the redundant tests were not removed, then the table of test combinations would be proportionally larger and there could result more optimal solutions of the cover to choose from. Because of the removal of redundant tests the procedure cannot generate all possible optimal solutions, but it does guarantee that at least one optimal solution will be found.

3.5 Package Level Diagnosis

The procedure presented above is easily extendable to package level diagnosis. Only two modifications to the strategy must be made. First a test which if removed from the table would cause two faults to become indistinguishable is essential only if the two

faults are not in the same package. The second modification is just as trivial as the first. Just as before, when performing the search for test intersection each and every fault must be considered individually. Now, when diagnosing to the package, the faults internal to the package need not be distinguished from each other. Therefore when examining an individual fault all of the columns which correspond to faults internal to the first fault's package can be set to 0. This can be added to the flow table by initially setting NINT(0) to all 1's and then setting to 0 the positions which correspond to faults inside the first fault's package.

3.6 Fault Detection

For fault detection the search routine is not used. The essential tests are now only those tests which, if removed from the table, would result in one or more faults becoming undetectable. The fault table can be condensed by forming the OR of each test's rows. At this point the application of any prime implicant covering procedure will result in the optimal set of tests which detect the given faults. This procedure is identical to that of Kautz[1].

3.7 Upper Bound on Search Routine

To determine an upper bound on the total number of test intersections that must be formed, the maximum number of intersections necessary to find all the test combinations for each individual fault will be derived. This will then be summed over all faults.

It is conceivable that a fault table exists where all combinations of tests must be considered. Consider such a fault table with "n" tests. At the first level of the intersection tree there are at most n intersections, one for each test. The first branch of these n corresponds to the first test, the second branch to the second test, and so on. The first branch or test must be intersected with tests 2, 3, ..., n. Therefore branch 1 fans out into n - 1 branches. (See Figure 2.) Since the intersection of test 1 and test 2 has been included under the first branch, the second branch or test in the first level need only be intersected with those tests below it, i.e., tests 3, 4, 5, ..., n. There are n - 2 such intersections. The third branch in the first level is intersected with tests 4, 5, 6, ..., n. Therefore the third branch has a fan-out of n - 3. Constructing the second level of the intersection tree in this manner, it can be seen that the n branches in the first level have fan-outs of n-1, n-2, n-3, ..., 2, 1, 0 respectively. In the construction of any level of the intersection tree, each branch (or test) need only be intersected with those tests below it. Therefore, the n - i branches (from a node with

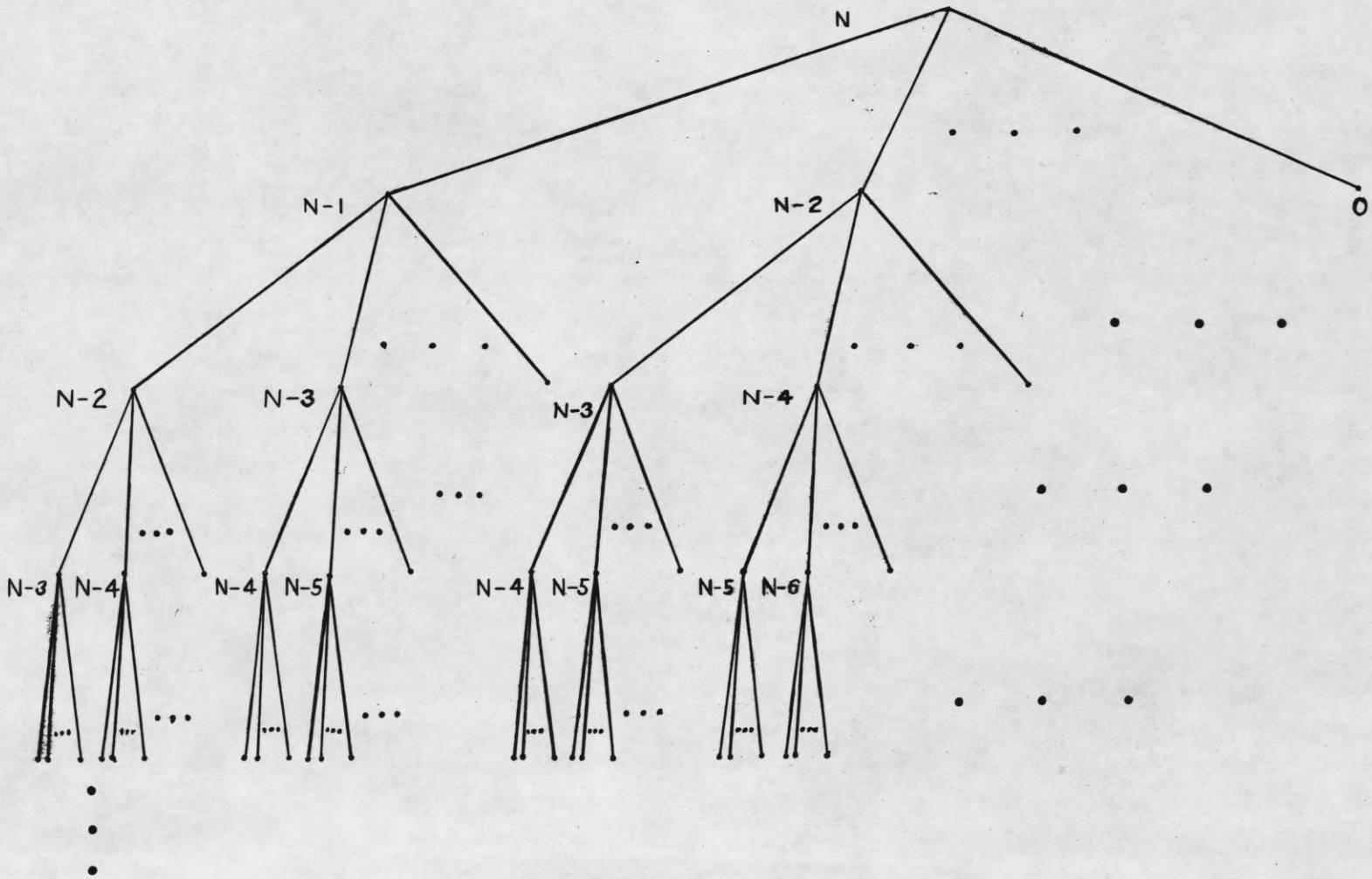


Figure 2. Intersection Tree

fan-out n - i where i < n) will have fan-outs of n - (i + 1), n - (i + 2), n - (i + 3), ..., 2, 1, 0 respectively. This process terminates after n levels.

Summing all of the branches in the n levels of the intersection tree one obtains the maximum number of test intersections which must be made in the "worst case" analysis for each fault. The summation is:

$$1(n) + 1(n - 1) + 2(n - 2) + 4(n - 3) + 8(n - 4) + \ldots$$

$$+ 2^{n-i}(n - (n-i + 1)) + \ldots + 2^{n-3}(n - (n-2)) +$$

$$+ 2^{n-2}(n - (n-1)) + 2^{n-1}(n \not\rightarrow^{0} n).$$

$$= n + \sum_{i=1}^{n} 2^{n-i} (n - (n-i + 1))$$

$$= n + \sum_{i=1}^{n} 2^{n-i} (i - 1)$$

It could easily be shown that this summation is equal to $2^n - 1$.

If there are k faults and n tests and if <u>no table reductions are made</u>, then the upper bound on the number of test intersections is $k(2^n - 1)$. For large n this approximates $k2^n$.

If table reductions are made then the upper bound becomes $\sum_{j=1}^{k} (2^{n_j} - 1)$ where n_j is the number of tests in the reduced fault table when the test combinations for the j^{th} fault are derived.

For any practical circuit it is unlikely that the number of iterations would ever approach this figure. However, the actual number can still be quite high, and it is for this reason that the table reduction techniques are employed.

4. CONCLUSIONS

The procedure presented here selects an optimum set of diagnostic tests from the fault table of a multiple output combinational net. Tests may be selected for diagnosis of individual faults, or for diagnosis to the package level. As with other test selection procedures which ensure optimality, large nets with numerous faults and many tests cannot be treated economically. The algorithm proposed here employs a search procedure which, in comparison to other optimal test selection procedures,[3] requires considerably less storage but more computation time. In fact, computation time tends to increase as 2^n, where n is the number of tests.

This optimal procedure, although not suited for large systems, could find use in the study of such problems as the following:

a) the diagnosability of circuits, in particular the identification of undetectable faults and indistinguishable faults,

b) the selection of test points to ensure diagnosability,

c) the effects of multiple faults,

d) the effects of hardware redundancy,

e) the influence of different packaging schemes on the partitioning of faults, and

f) the evaluation of proposed near-optimal selection procedures.

The procedure described here can also be used in conjunction with a test generation algorithm to construct, rather than select, a set of diagnostic tests by deriving the fault table row by row. The procedure described in this report can be used to evaluate tests

supplied by the test generation algorithm and to guide the generation of new tests.

Suggested topics for further research include obvious improvements to the procedure, aimed primarily at reducing the computation time, investigation of alternatives to the search routine, such as the Branch-and-Bound algorithm,[13] and the development of suitable heuristics to be used instead of the exhaustive search, particularly for test selection procedures which are not necessarily optimal.

REFERENCES

1) KAUTZ, W. H., "Fault Diagnosis in Combinational Digital Circuits," Digest of the First Annual IEEE Computer Conference, pp. 2-5, September 1967.

2) CHANG, H. Y., "An Algorithm for Selecting an Optimum Set of Diagnostic Tests," IEEE Transactions on Electronic Computers, Vol. EC - 14, No. 5, pp. 705-711, October 1965.

3) POWELL, T. J., "A Procedure for Ranking Diagnostic Test Inputs," Report R-354, May 1967, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois.

4) POAGE, J. F., "Derivation of Optimal Tests to Detect Faults in Combinational Circuits," Mathematical Theory of Automata, Polytechnic Press, Brooklyn, N. Y., pp. 483-528, 1963.

5) SESHU, S., "On an Improved Diagnosis Program," Report R-207, May 1964, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois; IEEE Transactions on Electronic Computers, (Short Notes), Vol. EC - 14, No. 1, pp. 76-79, February 1965.

6) SESHU, S. and FREEMAN, D. N., "The Diagnosis of Asynchronous Sequential Switching Systems," IRE Transactions on Electronic Computers, Vol. EC - 11, No. 4, pp. 459-465, August 1962.

7) ARMSTRONG, D. B., "On Finding a Nearly Minimal Set of Fault Detection Tests for Combinational Logic Nets," IEEE Transactions of Electronic Computers, Vol. EC - 15, No. 1, pp. 66-73, February 1966.

8) BOUKNIGHT, W. J., "On the Generation of Diagnostic Test Procedures," Report R-292, May 1966, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois.

9) ROTH, J. P., "Diagnosis of Automatic Failures: A Calculus and a Method," IBM Journal of Research and Development, Vol. 10, No. 4, pp. 278-291, July 1966.

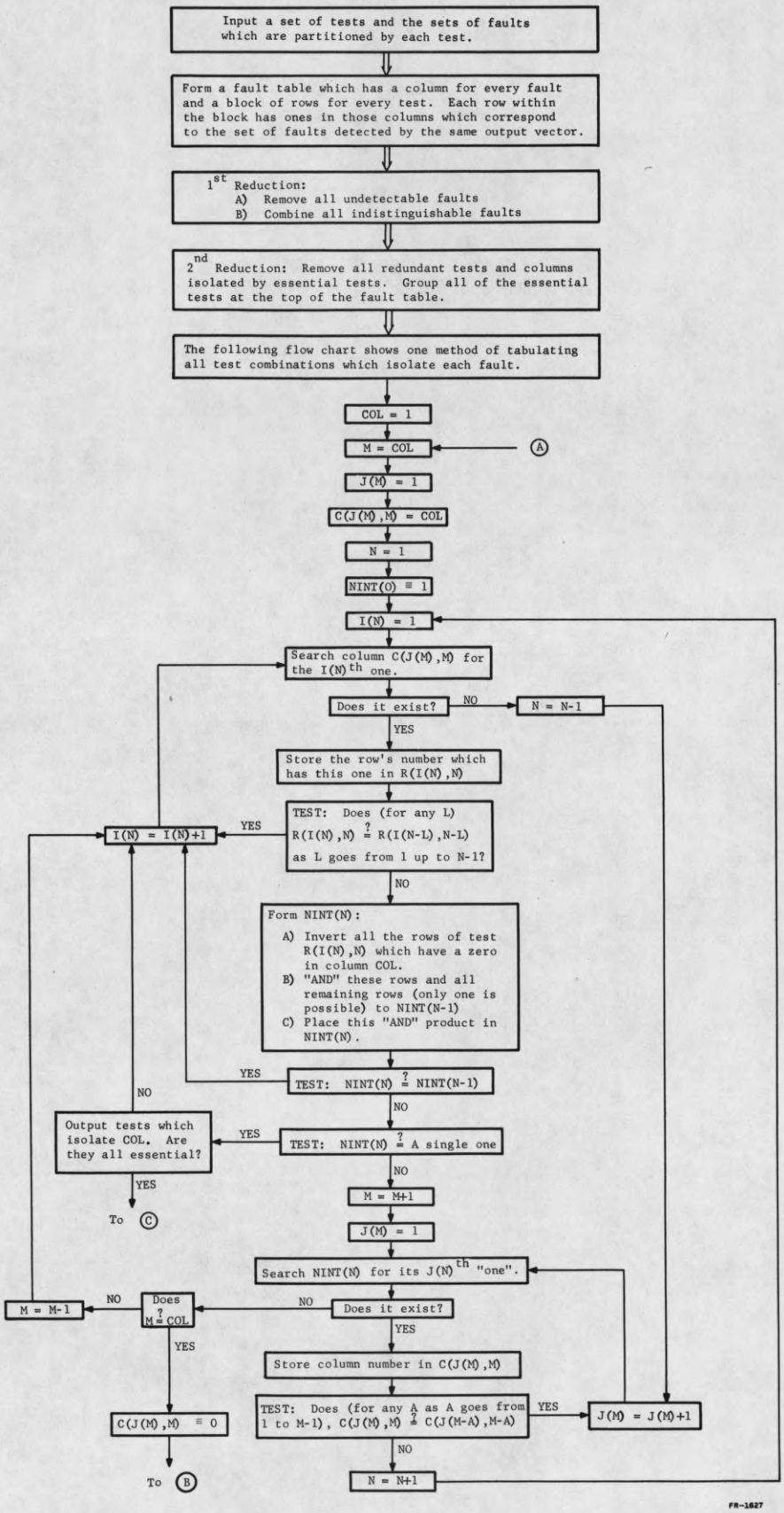
10) CHANG, H. Y. and THOMIS, W., "Methods of Interpreting Diagnostic Data for Locating Faults in Digital Machines," Bell System Technical Journal, Vol. 46, pp. 287-317, February 1967.

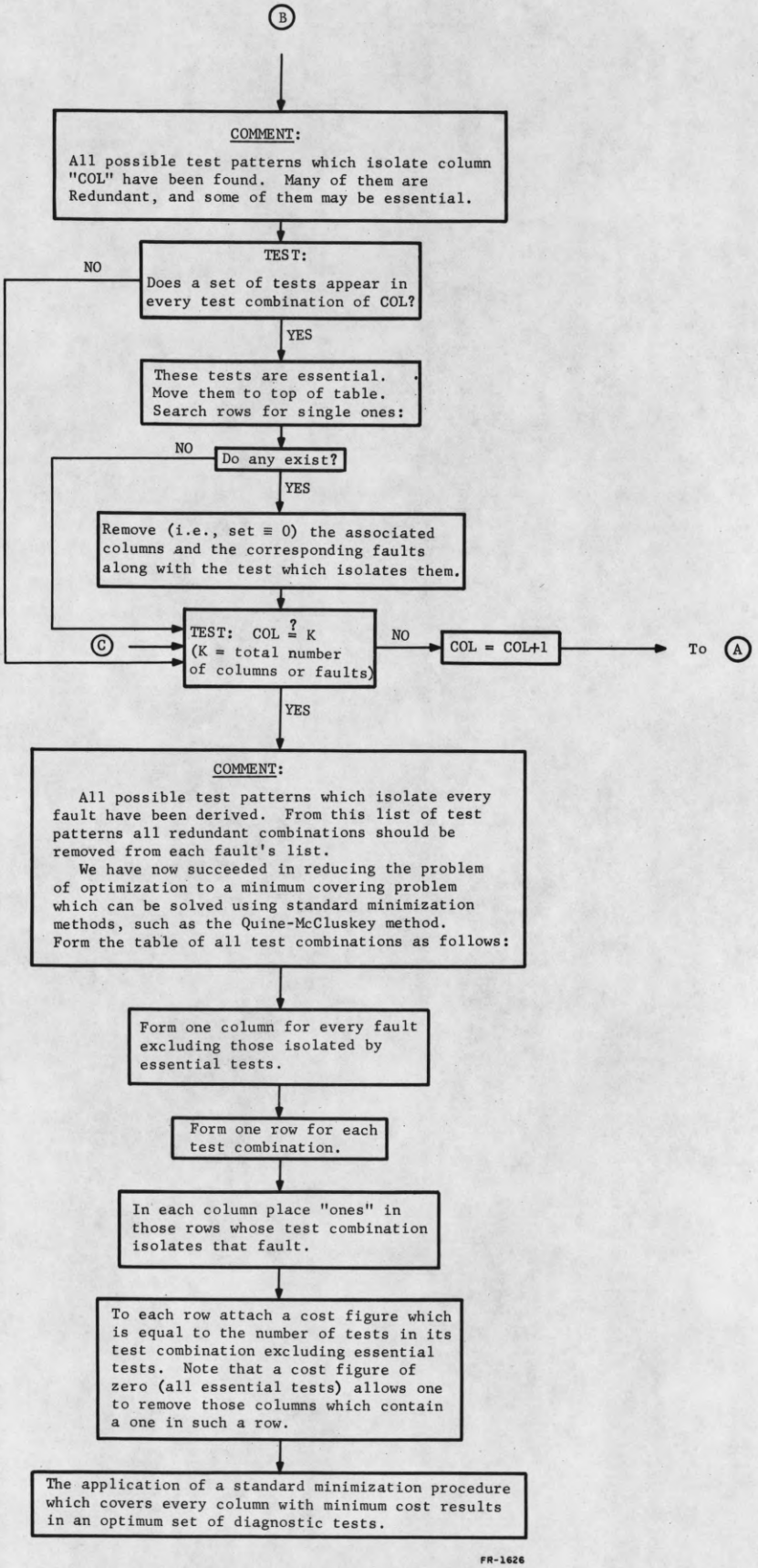
11) DOYLE, R. H., MEYER, R. A., and TUOMENOKSA, L. S., "No. 1 ESS Maintenance Plan," Bell System Technical Journal, Vol. 43, pp. 1961-2020, September 1964.

12) LITTLE, J. D. C., MURTH, K. G., SWEENEY, D. W., and CAROLINE, K., "An Algorithm for the Travelling Salesman Problem," Operations Research, Vol. 11, pp. 972-989, 1963.

APPENDIX

Flow Table


Input a set of tests and the sets of faults which are partitioned by each test.
Form a fault table which has a column for every fault and a block of rows for every test. Each row within the block has ones in those columns which correspond to the set of faults detected by the same output vector.
1st Reduction:
A) Remove all undetectable faults
B) Combine all indistinguishable faults
2nd Reduction: Remove all redundant tests and columns isolated by essential tests. Group all of the essential tests at the top of the fault table.
The following flow chart shows one method of tabulating all test combinations which isolate each fault.
COL = 1
M = COL
A
J(M) = 1
C(J(M),M) = COL
N = 1
NINT(O) ≡ 1
I(N) = 1
Search column C(J(M),M) for the I(N)th one.
Does it exist?
NO
N = N-1
YES
Store the row's number which has this one in R(I(N),N)
TEST: Does (for any L) R(I(N),N) ? R(I(N-L),N-L) as L goes from 1 up to N-1?
YES
I(N) = I(N)+1
NO
Form NINT(N):
A) Invert all the rows of test R(I(N),N) which have a zero in column COL.
B) "AND" these rows and all remaining rows (only one is possible) to NINT(N-1)
C) Place this "AND" product in NINT(N).
TEST: NINT(N) ? NINT(N-1)
YES
NO
TEST: NINT(N) ? A single one
YES
NO
Output tests which isolate COL. Are they all essential?
YES
NO
To C
M = M+1
J(M) = 1
Search NINT(N) for its J(N)th "one".
Does it exist?
NO
YES
Does M ? COL
NO
M = M-1
YES
Store column number in C(J(M),M)
TEST: Does (for any A as A goes from 1 to M-1), C(J(M),M) ? C(J(M-A),M-A)
YES
J(M) = J(M)+1
NO
C(J(M),M) ≡ 0
To B
N = N+1
FR-1627


B
COMMENT:
All possible test patterns which isolate column "COL" have been found. Many of them are Redundant, and some of them may be essential.
TEST:
Does a set of tests appear in every test combination of COL?
NO
YES
These tests are essential. Move them to top of table. Search rows for single ones:
Do any exist?
NO
YES
Remove (i.e., set ≡ 0) the associated columns and the corresponding faults along with the test which isolates them.
C
TEST: COL ? K
(K = total number of columns or faults)
NO
COL = COL+1
To A
YES
COMMENT:
All possible test patterns which isolate every fault have been derived. From this list of test patterns all redundant combinations should be removed from each fault's list.
We have now succeeded in reducing the problem of optimization to a minimum covering problem which can be solved using standard minimization methods, such as the Quine-McCluskey method. Form the table of all test combinations as follows:
Form one column for every fault excluding those isolated by essential tests.
Form one row for each test combination.
In each column place "ones" in those rows whose test combination isolates that fault.
To each row attach a cost figure which is equal to the number of tests in its test combination excluding essential tests. Note that a cost figure of zero (all essential tests) allows one to remove those columns which contain a one in such a row.
The application of a standard minimization procedure which covers every column with minimum cost results in an optimum set of diagnostic tests.
FR-1626

DISTRIBUTION LIST AS OF APRIL 1, 1967

1 Dr. Edward M. Reilley
Asst. Director (Research)
Ofc. of Defense Res. & Engrg.
Department of Defense
Washington, D. C. 20301

1 Office of Deputy Director
(Research and Information Rm. 3D1037)
Department of Defense
The Pentagon
Washington, D. C. 20301

1 Director
Advanced Research Projects Agency
Department of Defense
Washington, D. C. 20301

1 Director for Materials Sciences
Advanced Research Projects Agency
Department of Defense
Washington, D. C. 20301

1 Headquarters
Defense Communications Agency (333)
The Pentagon
Washington, D. C. 20305

50 Defense Documentation Center
Attn: TISIA
Cameron Station, Bldg. 5
Alexandria, Virginia 22314

1 Director
National Security Agency
Attn: TDL
Fort George G. Meade, Maryland 20755

1 Weapons Systems Evaluation Group
Attn: Col. Daniel W. McElwee
Department of Defense
Washington, D. C. 20305

1 National Security Agency
Attn: R4-James Tippet
Office of Research
Fort George G. Meade, Maryland 20755

1 Central Intelligence Agency
Attn: OCR/DD Publications
Washington, D. C. 20505

1 Colonel Kee
AFRSTE
Hqs. USAF
Room 1D-429, The Pentagon
Washington, D. C. 20330

1 Colonel A. Swan
Aerospace Medical Division
Brooks Air Force Base, Texas 78235

1 AUL3T-9663
Maxwell AFB, Alabama 36112

1 AFFTC (FTBPP-2)
Technical Library
Edwards AFB, California 93523

1 Space Systems Division
Air Force Systems Command
Los Angeles Air Force Station
Los Angeles, California 90045
Attn: SSSD

1 Major Charles Waespy
Technical Division
Deputy for Technology
Space Systems Division, AFSC
Los Angeles, California 90045

1 SSD(SSTRT/Lt. Starbuck)
AFUPO
Los Angeles, California 90045

1 Det. #6, OAR (LOOAR)
Air Force Unit Post Office
Los Angeles, California 90045

1 Systems Engineering Group (RTD)
Technical Information Reference Branch
Attn: SEPIR
Directorate of Engineering Standards
& Technical Information
Wright-Patterson AFB, Ohio 45433

1 ARL (ARIY)
Wright-Patterson AFB, Ohio 45433

1 Dr. H. V. Noble
Air Force Avionics Laboratory
Wright-Patterson AFB, Ohio 45433

1 Mr. Peter Murray
Air Force Avionics Laboratory
Wright-Patterson AFB, Ohio 45433

1 AFAL (AVTE/R.D. Larson)
Wright-Patterson AFB, Ohio 45433

2 Commanding General
Attn: STEWS-WS-VT
White Sands Missile Range,
New Mexico 88002

1 RADC (EMLAL-I)
Griffiss AFB, New York 13442
Attn: Documents Library

1 Academy Library (DFSLB)
U. S. Air Force Academy
Colorado Springs, Colorado 80912

1 Lt. Col. Bernard S. Morgan
Frank J. Seiler Research Laboratory
U. S. Air Force Academy
Colorado Springs, Colorado 80912

1 APGC (PGBPS-12)
Elgin AFB, Florida 32542

1 Commanding Officer
Human Engineering Laboratories
Aberdeen Proving Ground, Maryland 21005

1 Director
U. S. Army Engineer Geodesy, Intelligence
and Mapping
Research and Development Agency
Fort Belvior, Virginia 22060

1 Commandant
U. S. Army Command and General Staff College
Attn: Secretary
Fort Leavenworth, Kansas 66270

1 Dr. H. Robl
Deputy Chief Scientist
U. S. Army Research Office (Durham)
Box CM, Duke Station
Durham, North Carolina 27706

1 Commanding Officer
U. S. Army Research Office (Durham)
Attn: CRD-AA-IP (Richard O. Ulsh)
Box CM, Duke Station
Durham, North Carolina 27706

1 Librarian
U. S. Army Military Academy
West Point, New York 10996

1 The Walter Reed Institute of Research
Walter Reed Medical Center
Washington, D. C. 20012

1 Commanding Officer
U. S. Army Electronics R&D Activity
Fort Huachuca, Arizona 85163

1 Commanding Officer
U. S. Army Engineer R&D Laboratory
Attn: STINFO Branch
Fort Belvoir, Virginia 22060

1 Commanding Officer
U. S. Army Electronics R&D Activity
White Sands Missile Range, New Mexico 88002

1 Dr. S. Benedict Levin, Director
Institute for Exploratory Research
U. S. Army Electronics Command
Fort Monmouth, New Jersey 07703

1 Director
Institute for Exploratory Research
U. S. Army Electronics Command
Attn: Mr. Robert O. Parker, Executive
Secretary, JSTAC (AMSEL-XL-D)
Fort Monmouth, New Jersey 07703

1 Commanding General
U. S. Army Electronics Command
Fort Monmouth, New Jersey 07703

Attn: AMSEL-SC
RD-D
RD-G
RD-GF
RD-MAT
XL-D
XL-E
XL-C
XL-S
HL-D
HL-CT-R
HL-CT-P
HL-CT-L
HL-CT-O
HL-CT-I
HL-CT-A
NL-D
NL-A
NL-P
NL-R
NL-S
KL-D
KL-E
KL-S
KL-T
VL-D
WL-D

1 Chief of Naval Research
Department of the Navy
Washington, D. C. 20360
Attn: Code 427

3 Chief of Naval Research
Department of the Navy
Washington, D. C. 20360
Attn: Code 437

2 Naval Electronics Systems Command
ELEX 03
Falls Church, Virginia 22046

1 Naval Ship Systems Command
SHIP 031
Washington, D. C. 20360

1 Naval Ship Systems Command
SHIP 035
Washington, D. C. 20360

2 Naval Ordnance Systems Command
ORD 32
Washington, D. C. 20360

2 Naval Air Systems Command
AIR 03
Washington, D. C. 20360

2 Commanding Officer
Office of Naval Research Branch Office
Box 39, Navy No. 100 F.P.O.
New York, New York 09510

1 AFETR Technical Library
(ETV, MU-135)
Patrick AFB, Florida 32925

1 AFETR (ETLLG-I)
STINFO Officer (For Library)
Patrick AFB, Florida 32925

1 Dr. L. M. Hollingsworth
AFCRL (CRN)
L. G. Hanscom Field
Bedford, Massachusetts 01731

1 AFCRL (CRMXLR)
AFCRL Research Library, Stop 29
L. G. Hanscom Field
Bedford, Massachusetts 01731

1 Colonel Robert E. Fontana
Department of Electrical Engineering
Air Force Institute of Technology
Wright-Patterson AFB, Ohio 45433

1 Colonel A. D. Blue
RTD (RTTL)
Bolling Air Force Base, D. C. 20332

1 Dr. I. R. Mirman
AFSC (SCT)
Andrews AFB, Maryland 20331

1 Colonel J. D. Warthman
AFSC (SCTR)
Andrews AFB, Maryland 20331

1 Lt. Col. J. L. Reeves
AFSC (SCBB)
Andrews AFB, Maryland 20331

2 ESD (ESTI)
L. G. Hanscom Field
Bedford, Massachusetts 01731

1 AEDC (ARO, INC)
Attn: Library/Documents
Arnold AFS, Tennessee 37389

2 European Office of Aerospace Research
Shell Building
47 Rue Cantersteen
Brussels, Belgium

5 Lt. Col. Robert B. Kalisch
Chief, Electronics Division
Directorate of Engineering Sciences
Air Force Office of Scientific Research
Arlington, Virginia 22209

1 U. S. Army Research Office
Attn: Physical Sciences Division
3045 Columbia Pike
Arlington, Virginia 22204

1 Research Plans Office
U. S. Army Research Office
3045 Columbia Pike
Arlington, Virginia 22204

1 Commanding General
U. S. Army Materiel Command
Attn: AMCRD-RS-DE-E
Washington, D. C. 20315

1 Commanding General
U. S. Army Strategic Communications Command
Washington, D. C. 20315

1 Commanding Officer
U. S. Army Materials Research Agency
Watertown Arsenal
Watertown, Massachusetts 02172

1 Commanding Officer
U. S. Army Ballistics Research Laboratory
Attn: V. W. Richards
Aberdeen Proving Ground
Aberdeen, Maryland 21005

1 Commandant
U. S. Army Air Defense School
Attn: Missile Sciences Division, C&S Dept.
P.O. Box 9390
Fort Bliss, Texas 79916

1 Redstone Scientific Information Center
Attn: Chief, Document Section
Redstone Arsenal, Alabama 35809

1 Commanding General
Frankford Arsenal
Attn: SMUFA-1310 (Dr. Sidney Ross)
Philadelphia, Pennsylvania 19137

1 U. S. Army Munitions Command
Attn: Technical Information Branch
Picatinney Arsenal
Dover, New Jersey 07801

1 Commanding Officer
Harry Diamond Laboratories
Attn: Dr. Berthold Altman (AMXDO-TI)
Connecticut Avenue and Van Ness Street, N.W.
Washington, D. C. 20438

1 Commanding Officer
U. S. Army Security Agency
Arlington Hall
Arlington, Virginia 22212

1 Commanding Officer
U. S. Army Limited War Laboratory
Attn: Technical Director
Aberdeen Proving Ground
Aberdeen, Maryland 21005

1 Commanding Officer
Office of Naval Research Branch Office
219 South Dearborn Street
Chicago, Illinois 60604

1 Commanding Officer
Office of Naval Research Branch Office
1030 East Green Street
Pasadena, California 91101

1 Commanding Officer
Office of Naval Research Branch Office
207 West 24th Street
New York, New York 10011

1 Commanding Officer
Office of Naval Research Branch Office
495 Summer Street
Boston, Massachusetts 02210

8 Director, Naval Research Laboratory
Technical Information Officer
Washington, D. C. 20390
Attn: Code 2000

1 Commander
Naval Air Development and Material Center
Johnsville, Pennsylvania 18974

2 Librarian
U. S. Naval Electronics Laboratory
San Diego, California 95152

1 Commanding Officer and Director
U. S. Naval Underwater Sound Laboratory
Fort Trumbull
New London, Connecticut 06840

1 Librarian
U. S. Navy Post Graduate School
Monterey, California 93940

1 Commander
U. S. Naval Air Missile Test Center
Point Mugu, California 95468

1 Director
U. S. Naval Observatory
Washington, D. C. 20390

2 Chief of Naval Operations
OP-07
Washington, D. C. 20350

1 Director, U. S. Naval Security Group
Attn: G43
3801 Nebraska Avenue
Washington, D. C. 20016

2 Commanding Officer
Naval Ordnance Laboratory
White Oak, Maryland 21162

1 Commanding Officer
Naval Ordnance Laboratory
Corona, California 91720

1 Commanding Officer
Naval Ordnance Test Station
China Lake, California 93555

1 Commanding Officer
Naval Avionics Facility
Indianapolis, Indiana 46218

1 Commanding Officer
Naval Training Device Center
Orlando, Florida 32813

1 U. S. Naval Weapons Laboratory
Dahlgren, Virginia 22448

1 Weapons Systems Test Division
Naval Air Test Center
Patuxtent River, Maryland 20670
Attn: Library

1 Head, Technical Division
U. S. Naval Counter Intelligence Support Center
Fairmont Building
4420 North Fairfax Drive
Arlington, Virginia 22203

1 Mr. Charles F. Yost
Special Asst. to the Director of Research
National Aeronautics and Space Administration
Washington, D. C. 20546

1 Dr. H. Harrison, Code RRE
Chief, Electrophysics Branch
National Aeronautics and Space Administration
Washington, D. C. 20546

1 Goddard Space Flight Center
National Aeronautics and Space Administration
Attn: Library C3/TDL
Green Belt, Maryland 20771

1 NASA Lewis Research Center
Attn: Library
21000 Brookpark Road
Cleveland, Ohio 44135

1 National Science Foundation
Attn: Dr. John R. Lehmann
Division of Engineering
1800 G Street, N.W.
Washington, D. C. 20550

1 U. S. Atomic Energy Commission
Division of Technical Information Extension
P. O. Box 62
Oak Ridge, Tennessee 37831

1 Los Alamos Scientific Laboratory
Attn: Reports Library
P. O. Box 1663
Los Alamos, New Mexico 87544

2 NASA Scientific & Technical Information Facility
Attn: Acquisitions Branch (S/AK/DL)
P. O. Box 33
College Park, Maryland 20740

1 Director
Research Laboratory of Electronics
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

1 Polytechnic Institute of Brooklyn
55 Johnson Street
Brooklyn, New York 11201
Attn: Mr. Jerome Fox
Research Coordinator

1 Director
Columbia Radiation Laboratory
Columbia University
538 West 120th Street
New York, New York 10027

1 Director
Coordinated Science Laboratory
University of Illinois
Urbana, Illinois 61801

1 Director
Stanford Electronics Laboratories
Stanford University
Stanford, California 94305

1 Director
Electronics Research Laboratory
University of California
Berkeley, California 94720

1 Director
Electronic Sciences Laboratory
University of Southern California
Los Angeles, California 90007

1 Professor A. A. Dougal, Director
Laboratories for Electronics and Related Sciences Research
University of Texas
Austin, Texas 78712

1 Division of Engineering and Applied Physics
210 Pierce Hall
Harvard University
Cambridge, Massachusetts 02138

1 Aerospace Corporation
P. O. Box 95085
Los Angeles, California 90045
Attn: Library Acquisitions Group

1 Professor Nicholas George
California Institute of Technology
Pasadena, California 91109

1 Aeronautics Library
Graduate Aeronautical Laboratories
California Institute of Technology
1201 East California Boulevard
Pasadena, California 91109

1 Director, USAF Project RAND
Via: Air Force Liaison Office
The RAND Corporation
1700 Main Street
Santa Monica, California 90406
Attn: Library

1 The Johns Hopkins University
Applied Physics Laboratory
8621 Georgia Avenue
Silver Spring, Maryland 20910
Attn: Boris W. Kuvshinoff
Document Librarian

1 Hunt Library
Carnegie Institute of Technology
Schenley Park
Pittsburgh, Pennsylvania 15213

1 Dr. Leo Young
Stanford Research Institute
Menlo Park, California 94025

1 Mr. Henry L. Bachmann
Assistant Chief Engineer
Wheeler Laboratories
122 Cuttermill Road
Great Neck, New York 11021

1 School of Engineering Sciences
Arizona State University
Tempe, Arizona 85281

1 University of California at Los Angeles
Department of Engineering
Los Angeles, California 90024

1 California Institute of Technology
Pasadena, California 91109
Attn: Documents Library

1 University of California
Santa Barbara, California 93106
Attn: Library

1 Carnegie Institute of Technology
Electrical Engineering Department
Pittsburgh, Pennsylvania 15213

1 University of Michigan
Electrical Engineering Department
Ann Arbor, Michigan 48104

1 New York University
College of Engineering
New York, New York 10019

1 Syracuse University
Department of Electrical Engineering
Syracuse, New York 13210

1 Yale University
Engineering Department
New Haven, Connecticut 06520

1 Airborne Instruments Laboratory
Deerpark, New York 11729

1 Bendix Pacific Division
11600 Sherman Way
North Hollywood, California 91605

1 General Electric Company
Research Laboratories
Schenectady, New York 12301

1 Lockheed Aircraft Corporation
P. O. Box 504
Sunnyvale, California 94088

1 Raytheon Company
Bedford, Massachusetts 01730
Attn: Librarian

1 Dr. G. J. Murphy
The Technological Institute
Northwestern University
Evanston, Illinois 60201

1 Dr. John C. Hancock, Director
Electronic Systems Research Laboratory
Purdue University
Lafayette, Indiana 47907

1 Director
Microwave Laboratory
Stanford University
Stanford, California 94305

1 Emil Schafer, Head
Electronics Properties Info Center
Hughes Aircraft Company
Culver City, California 90230

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*
University of Illinois
Coordianted Scienc Laboratory
Urbana, Illinois 61801

2a. REPORT SECURITY CLASSIFICATION
Unclassified

2b. GROUP

3. REPORT TITLE

OPTIMAL TEST SETS FOR THE DIAGNOSIS OF MULTIPLE OUTPUT COMBINATIONAL NETS

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

5. AUTHOR(S) *(First name, middle initial, last name)*

ISENHART, Richard D.

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
May 1968	46	12

8a. CONTRACT OR GRANT NO.
DAAB-07-67-C-0199; also in part NSF GK-1663

b. PROJECT NO.

c.

d.

9a. ORIGINATOR'S REPORT NUMBER(S)
R-381

9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

10. DISTRIBUTION STATEMENT

Distribution of this report is unlimited.

11. SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY
Joint Services Electronics Program
thru U.S. Army Electronics Command
Ft. Monmouth, New Jersey 07703

13. ABSTRACT

Given a set of tests and the faults which each of them partition, it is desirable to have a systematic method for selecting a subset of these tests which is minimum in number and which will isolate all distinguishable faults. Presented in this paper is a tabular method which selects an optimum set of diagnostic tests from the fault table of multiple output combinational nets. The procedure first removes undetectable faults and combines indistinguishable faults. Then it systematically removes redundant tests and tabulates most of the essential tests which must be used to isolate certain faults. From this reduced table the test combinations produces the remaining essential tests which must be used for isolating certain faults. The essential tests are removed from the test combinations and are tabulated separately. Then the reduced test combinations are tabulated and Quine-McClusky minimization is used to select a minimum set of tests which diagnoses the remaining faults.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6811

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Network Diagnosis						
Combinational Logic Networks						
Fault Tables						
Test Selection						
Test Minimization						
Optimal Test Sets						
Failure Location						
Package Level Diagnosis						