STARS

University of Central Florida
STARS

Retrospective Theses and Dissertations

Fall 1981

An Optimal Algorithm for Detecting Pattern Sensitive Faults in Semiconductor Random Access Memories

Richard I. Subrin
University of Central Florida

Part of the Engineering Commons
Find similar works at: https://stars.library.ucf.edu/rtd
University of Central Florida Libraries http://library.ucf.edu

This Masters Thesis (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Retrospective Theses and Dissertations by an authorized administrator of STARS. For more information, please contact STARS@ucf.edu.

STARS Citation
Subrin, Richard I., "An Optimal Algorithm for Detecting Pattern Sensitive Faults in Semiconductor Random Access Memories" (1981). *Retrospective Theses and Dissertations*. 596.
https://stars.library.ucf.edu/rtd/596

University of Central Florida
STARS
Showcase of Text, Archives, Research & Scholarship

AN OPTIMAL ALGORITHM FOR DETECTING PATTERN SENSITIVE FAULTS IN SEMICONDUCTOR RANDOM ACCESS MEMORIES

BY

RICHARD IRA SUBRIN
B.S.E.E. University of Illinois, June 1973

RESEARCH REPORT

Submitted in partial fulfillment of the requirements
for the Master of Science in Engineering
in the Graduate Studies Program of the College of Engineering
University of Central Florida
Orlando, Florida

Fall Term
1981

ABSTRACT

Random-access memory (RAM) testing to detect unrestricted pattern-sensitive faults (PSFs) is impractical due to the size of the memory checking sequence required. A formal model for restricted PSFs in RAMs called adjacent-pattern interference faults (APIFs) is presented. A test algorithm capable of detecting APIFs in RAMs requiring a minimum number of memory operations is then developed.

ACKNOWLEDGEMENTS

I would like to express my gratitude to Dr. Darrell G. Linton for his guidance and encouragement. He took the time to familiarize himself with my research topic and provided timely advice and assistance. I very much appreciated the opportunity to perform my research on a subject in which I have always been deeply interested.

I would also like to thank my dear wife, Wendy, without whose support and patience this would not have been possible.

TABLE OF CONTENTS

	PAGE
LIST OF TABLES	v
LIST OF FIGURES	vi
CHAPTER	
1. INTRODUCTION	1
2. RAM TESTING	4
3. THE FAULT MODEL	15
4. THE NEIGHBORHOOD MODEL	21
5. BOUNDING THE TEST PROBLEM	26
6. RAM CELL ASSIGNMENT	33
7. DETERMINING A MINIMUM SEQUENCE	40
8. SUMMARY AND CONCLUSIONS	72
9. SUGGESTIONS FOR FUTURE RESEARCH	74
DEFINITION OF TERMS	75
DEFINITION OF NOTATION	82
REFERENCES	83

LIST OF TABLES

TABLE		PAGE
1.	POSSIBLE COUPLING FAULT COMBINATIONS	18
2.	ADJACENT NEIGHBORHOOD PATTERNS (ANPs) FOR A FIVE-CELL NEIGHBORHOOD	29
3.	TABLE OF WRITE OPERATIONS REQUIRED TO GENERATE ALL THIRTY-TWO ANPs INVOLVING ZERO TO ONE TRANSITIONS	31
4.	MEMORY ARRAY CELL ASSIGNMENT	36
5.	SIXTY-FOUR ANPs CORRESPONDING TO PATHS ON THE STATE DIAGRAM OF A FIVE-CELL NEIGHBORHOOD	49
6.	ANP OPERATIONAL SEQUENCE TABLE	56
7.	ALGORITHM TO DETECT APIFs (DAPIF)	58
8.	ALGORITHM DAPIF MEMORY ACCESS TABULATION	59

LIST OF FIGURES

FIGURE		PAGE
1.	BLOCK DIAGRAM OF A SEMICONDUCTOR RAM	5
2.	MARCHING ONES AND ZEROS TEST	9
3.	GALLOPING ONES AND ZEROS TEST	11
4.	DEVELOPMENT OF FIVE-CELL NEIGHBORHOOD PATTERNS	37
5.	CELL ASSIGNMENT IN AN 8 X 8 MEMORY ARRAY	38
6.	DIGRAPH OF ALL POSSIBLE FIVE-CELL NEIGHBORHOOD PATTERNS	44
7.	NEIGHBORHOOD DIGRAPH DECOMPOSED INTO SIX EULERIAN SUBGRAPHS	47
8.	TWENTY-FOUR ARC-DISJOINT SEQUENCES COMPOSING THE NEIGHBORHOOD DIGRAPH	50
9.	ALGORITHM DAPIF APPLIED TO A 4 X 4 MEMORY CELL ARRAY	60

CHAPTER 1

INTRODUCTION

Rapid advances in semiconductor technology have led to a trend toward larger and denser memories. This is evidenced by the fact that 16K Random-Access Memories (RAMs) are in widespread commercial use and 64K RAM chips are now available from most semiconductor manufacturers. An indirect result of this growth is not only a higher incidence of failure but an increase in the complexity of the failure modes themselves. This fact coupled with the increase in the number of storage elements in the memory to test have resulted in escalating testing costs (Srini 1977).

At this same time memories are being designed into an increasing number of different types of electronic equipment. Although responsibility for memory testing first belongs to the device manufacturer, it is the end-product manufacturer and the purchaser who must ultimately deal with the problem of memory reliability (Rosenfield 1979). RAMs are inherently less reliable than other commonly used integrated circuits (ICs) and

account for a disproportionate number of failures in computers and other electronic systems. This is due to the large number of devices contained on each RAM chip coupled with the large number of RAM chips used in most applications. In one study by J. B. Clary and R. A. Sacane (1979), memory failures accounted for up to 94% of the total failures in a PDP-11/70 computer system. It is this problem of reliability combined with testing complexity which makes memory testing a subject of continuing interest.

This paper deals with a class of failures known as pattern sensitive failures (PSFs) which occur in the memory array portion of the RAM chip. PSFs are caused by device anomolies and parasitic effects which can make the memory sensitive to both data patterns and the sequence of memory accesses. Although this class of faults represents only one of many which can occur in a RAM chip, it poses the most time consuming and therefore the most costly testing problem.

There can be 2^N different patterns of data in a memory of N cells. This is further complicated when the sequence of memory accesses is taken into account. Hayes (1975) formalized this problem by defining a sequential machine with 2^N states and 3^N inputs. If PSFs are considered unrestricted, he calculated that a

checking sequence of length $(3N^2 + 2N)2^N$ would be required. This results in test algorithms which are computationally infeasible (Anderson 1972; Srini 1976).

In practice, therefore, restrictions must be placed on the number of test patterns used to perform pattern sensitivity testing. An algorithm for memory testing based on the assumption that PSFs occur only among adjacent cells will be developed. This restriction which will heretofore be referred to as adjacent-pattern interference faults (APIFs). It is justifiable since charge leakage, parasitic capacitance and other phenomena to which pattern sensitive faults are attributed are likely to affect the contents of the immediate neighbor cell(s) whether or not other, more distant cells in the memory are similarly affected (Srini 1977).

The memory testing algorithm developed is considered optimal for the restricted neighborhood model presented. In the sense that a minimal number of RAM read/write operations are required.

CHAPTER 2

RAM TESTING

Commercially available RAMs are comprised of a two-dimensional memory cell array, row and column address decoder, write driver, sense amplifier and I/0 port, as illustrated in Figure 1 (Thatte 1977). A memory cell array consists of n rows and m columns whereby a particular cell is accessed by addressing the row and column corresponding to the cell and activating the desired operation mode, either read or write.

RAM testing involves the application of selected test algorithms to detect or locate faults. These test algorithms are comprised of a sequence of write and read operations to the memory cell array and can be conceptually divided into three categories (Suk 1978; Thatte 1977):

1. Functional testing: the test must detect physical failures which cause the RAM to function incorrectly; e.g., faults in memory cells, address logic, sense amplifiers, write drivers, noise coupling between cells, etc.

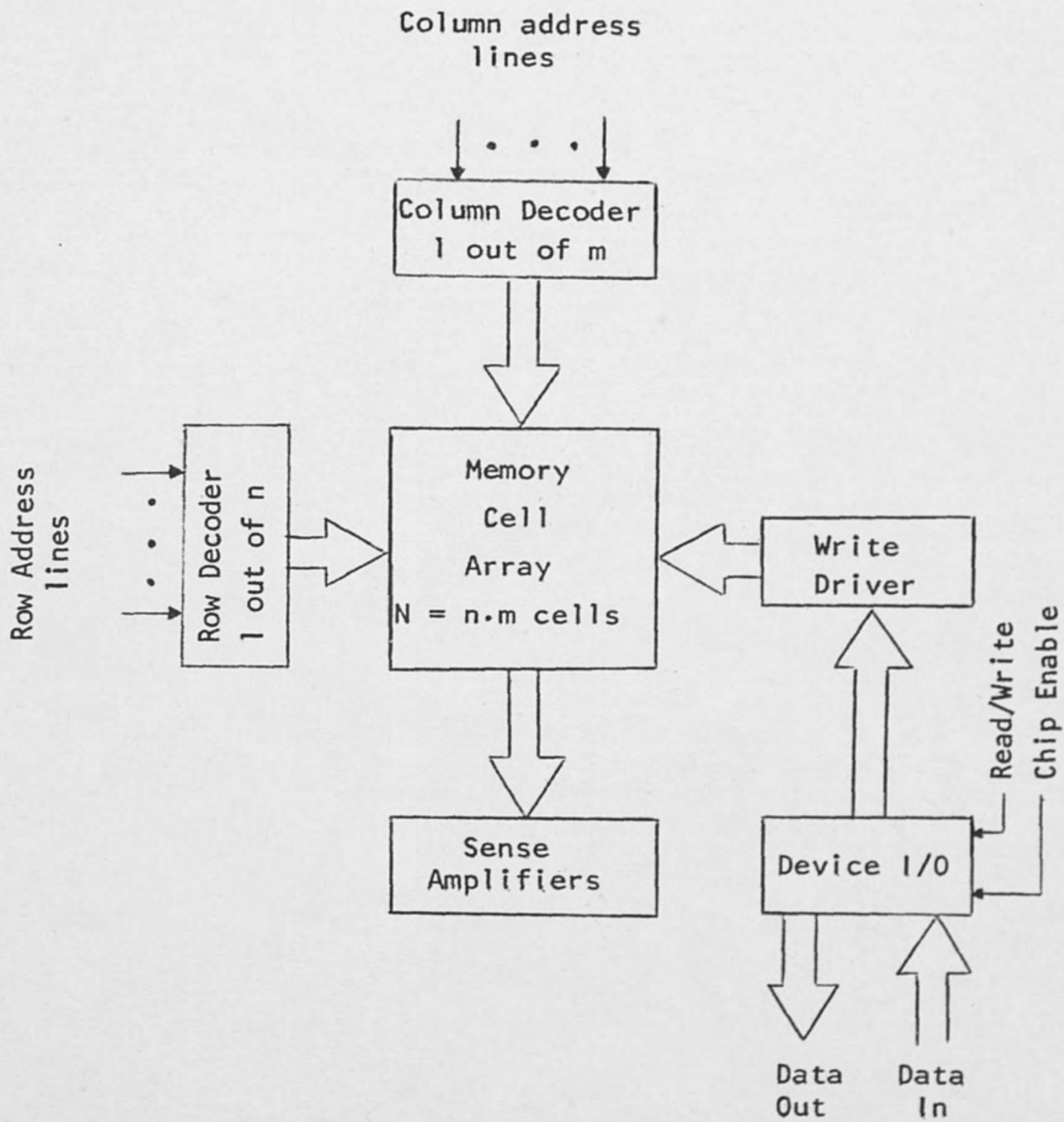


Fig. 1. Block Diagram of a Semiconductor RAM

2. Pattern sensitivity testing: even though a RAM has no physical failure, there could be device anomalies and parasitic effects which could make its dynamic behavior sensitive to data and/or patterns.

3. D.C. parameter testing: the D.C. parameters like power dissipation, fan out capabilities, noise margins and leakage currents must be checked. Since D.C. parameter testing is usually not a major RAM testing problem area, it will not be dealt with here.

Functional Testing

Functional tests for the memory cell array have been developed to detect the following types of faults:

1. One or more cells are stuck-at-zero or stuck-at-one (these faults are called cell stuck-at-faults). It should be emphasized that when a cell is stuck-at-x, then it will remain at x state, independent of reads and writes to any cell of the memory.

2. One or more cells fail to undergo a 0 to 1 and/or 1 to 0 transition, when the complement of the contents of the memory cell are written into the cell (these faults are called transition faults).

3. There exist two or more cells which are coupled. By this it is meant that a 0 to 1 or 1 to 0 transition in a cell changes the state of another cell in the memory, independent of the contents of other cells. This does not imply that if a transition in the state of cell C_i changes the state of cell C_j, then a transition in the state of cell C_j changes the state of cell C_i (these faults are called coupling faults) (Suk 1978).

Three of the most widely known and frequently used tests for semiconductor memories are the Marching Ones and Zeros, the Walking Ones and Zeros and the Galloping Ones and Zeros tests. These tests and variations of them are commonly used to test for interaction between pairs of cells in the memory.

The Marching Ones and Zeros is a basic test of memory to reasonably assure that it is functional (that is, the addressing is operational and each cell can be read and written in the zero/one state). The memory is first written to the all-zeros state. Then sequentially, starting at the first address, a zero is read and a one is written. This sequence is continued to the last location (i.e., until the memory is full of ones). Then, starting at the last location, a one is read and a zero is written. The address is reduced one location and the

sequence is repeated until the first location is reached. This overall sequence is then repeated with the data reversed.

As the memory is being scanned in the ascending direction, any effect on a location above will be detected when it is eventually read. If the effect is on a location below, it will not be detected until the memory is scanned in reverse. This by no means tests everything or all interactions, but does reasonably assure that the memory is working and that no defective elements are present. This is illustrated in Figure 2 using a 16-cell RAM array where the first cell, C_0, is in the upper left hand corner and the fourth cell, C_3, is in the upper right hand corner of the memory array.

The Walking Ones and Zeros test is much more extensive than the marching ones and zeros. Initially, all locations are written to a "background" pattern of all zeros and verified. Then starting at the first location, a "test cell" of one is written. All other locations in the memory are sequentially read to verify that they still contain the background pattern of all zeros. The "test cell" one is then read and written back to a zero. After this first iteration, it is known that writing a one in the first location does not affect any other location and reading in all other locations does not

1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

Write entire memory with data pattern, in this example, all "1".

0	0	0	0
0	1	1	1
1	1	1	1
1	1	1	1

In sequence read a word and write that word with the complement data (cell 4 has just been written and cell 5 is next to be read and checked).

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

All cells in the memory have been read in sequence and the data replaced with complement data ("0").

1	1	1	1
1	1	1	1
1	1	1	0
0	0	0	0

Repeat the sequence, each word in its turn, reading the memory and writing new data. (Cell 10 has just been written with new data and cell 11 is next to be read and checked.

Fig. 2. Marching Ones and Zeros Test

affect the first location. This sequence is then repeated for every location in the memory. At the completion of a walking one through a field of zeros, the patterns are reversed and a zero is walked through a field of ones. Overall, this test sequence results in $2(N^2 + 4N)$ operations where N is the number of locations in the memory.

The Galloping Ones and Zeros (GALPAT) is a test pattern that includes testing all possible address transitions. It uses the same data pattern sequence as walking ones and zeros. Initially, all locations are written to a background pattern of zeros. Then, starting with the first location (cell 0), a test cell of one is written followed by a read-sequence of read cell one, read cell zero (test cell), read cell two, read cell zero, etc., until every cell in the memory is read alternately with the test cell. The test cell is then moved to the second location and the sequence repeated, making the same alternating checks with respect to the second location. This process is repeated for all memory locations. The patterns are then reversed and the overall sequence is repeated (Colbourne, Coverley, and Behera 1974; Hnatek 1975). See Figure 3.

The GALPAT is the most extensive and complete of the three, however, it requires approximately $4N^2$ operations

1	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

Write a "1" in cell 0 of the memory. Verify rest of the memory is "0", however after reading each new location go back and read the "1" in cell 0.

0	1	0	0
0	0	0	0
0	0	0	0
0	0	0	0

Write cell 0 of the memory to 0. Write cell 1 (which is now the test cell) with a "1". Write "0" into the rest of the memory however following the writing of each location read the test cell (cell 1).

0	0	0	0
0	0	0	0
0	0	0	0
0	1	0	0

Repeat the test until each location has had its turn at being the test cell (the cell with the "1"). The example shows cell 13 as the test cell.

Fig. 3. Galloping Ones and Zeros Test

and therefore is not practical for large RAMs. For example, application of GALPAT to a 16K RAM with a 500 nanosecond access time requires over eight minutes. Numerous other test procedures for functional faults have been developed, however, the most comprehensive of these have been variations of GALPAT and require similarly long execution times.

Pattern Sensitive Faults

Pattern sensitive faults (PSFs) are caused by device anomolies such as dynamic timing parameters, current leakage and parasitic capacitance. As discussed earlier, it is impractical to attempt to detect unrestricted PSFs due to the length of the required test sequence. Therefore, the approach which will be used is to check for all possible dynamic patterns occurring within a restricted neighborhood of each memory cell of the RAM under test.

For the purpose of this paper, the neighborhood will be restricted to memory cells immediately adjacent to the cell under test and faults in this neighborhood will be referred to as Adjacent Pattern Interference Faults (APIFs). An APIF is one in which the content of a memory cell, say C_i, changes as a result of certain patterns of zeros, ones, zero-to-one transitions and

one-to-zero transitions occurring in those cells immediately adjacent to C_i (Patch 1980).

As is readily apparent, those tests developed for functional testing are ineffective in detecting PSFs. For this reason, test algorithms developed specifically for PSFs are necessary.

Unlike tests for functional faults, which tend to treat the memory as a "black box", tests for PSFs assume that the layout of the memory cell array is known. Knowledge about the RAM's internal architecture is necessary in order to restrict the neighborhood across which all interference is presumed to take place.

J. P. Hayes of the University of Southern California first developed a general memory failure model in 1975. It was based on checking experiments for sequential machines and resulted in a comprehensive but impractically large test sequence. In a more recent work, Hayes (1980) presented a single pattern sensitive fault model and developed an optimal test sequence based on non-overlapping (tiling) neighborhoods. In 1976, V. P. Srini at Virginia Polytechnic Institute presented a set of 32 heuristically determined memory assignments to detect APIFs in a nine-cell neighborhood. While concentrating on providing every combination of 0s and 1s in each row of the neighborhood, the sequence of the

memory operations was presumed not to be a factor. In the most recent and technically advanced work on PSFs to date, D. S. Suk (1978) developed test algorithms to detect and locate neighborhood PSFs. Their algorithms were based on a fault model which presumed that APIFs were such that a zero-to-one (one-to-zero) transition in one cell could cause an adjacent cell to change from one to zero or zero to one. It is the intent of this paper to assume an alternative fault model whereby a zero-to-one (one-to-zero) transition can only increase the number of ones (zeros) in the memory. Then utilizing a neighborhood model, an optimal test algorithm is derived. The justification for this is detailed in the next chapter.

CHAPTER 3

THE FAULT MODEL

Random access memories typically contain a two-dimensional memory cell array consisting of n rows and m columns where n and m are equal to an integral power of two. Although there are numerous bipolar and Metal-Oxide Semiconductor (MOS) memory cell designs, one basic principle is common to all RAMs. When a memory cell is to be accessed, the row and column select lines associated with the specified memory cell are activated. This memory cell resides at the intersection of the activated row and column and like all other cells in the memory, when functioning properly it is accessed uniquely. It is this operational principle which is violated by the APIF.

It is the intent of this paper to develop memory test algorithms to detect all Adjacent Pattern Interference Faults (APIFs). These faults exhibit a "coupling" between memory cells in the presence of certain patterns of ones and zeros in other, nearby cells. All coupling faults will be classified as "monotonic" write faults.

For example, the operation of writing a one into cell C_i is more likely to write an erroneous one into cell C_j than it is to write an erroneous zero into cell C_j. This concept was first presented by J. P. Hayes (1975) in his paper "Detection of Pattern-Sensitive Faults in Random-Access Memories". Hayes suggested that an effective method of restricting test length is to restrict the kinds of PSFs that can occur. He indicated that in practical situations, only monotonic interactions need be considered, however noting that such restrictions on PSFs tend to complicate the test generation process. The concept of monotonic write faults is not new, but rather provides the basis for many of the most commonly used memory tests such as the Marching Ones and Zeros and the Walking Ones and Zeros tests (Colbourne 1974; Huston 1973; Shah 1976).

In order to develop a fault model for monotonic coupling faults, the following definition is required:

Definition: A memory cell C_i is said to be <u>coupled</u> to another memory cell C_j, $i \neq j$, if and only after a 0 to 1 (1 to 0) transition in cell C_j changes the contents of cell C_i only if the contents of cell C_i is one (zero), regardless of its previous contents. In this case, C_i can be referred

to as the coupled cell and C_j can be referred to as the coupling cell (Suk 1978).

Coupling faults can occur due to parasitic capacitive coupling between cells or due to leakage current from one cell to another (Fischer 1974).

This can be caused by subthreshold current in the thick oxide separating one storage cell from another, interface charge trapping, ionic contamination and numerous other causes (Batdorf 1978; Green 1979; Hnatek 1976; Srini 1977).

The above definitions of coupling faults may occur in combinations as given in Table 1. Since multiple coupling faults can exist between cells C_i and C_j, the contents of the cells must be verified in such a way as to not enable the coupling faults to go undetected. This can occur when two coupling faults cancel each other out. For example, multiple coupling faults could exist as given in Figure 1 combination 10, with C_i coupled to C_j, the coupling cell, for both zero-to-one and one-to-zero transitions. If C_j is written back and forth from a one to a zero and back to a one, C_i will undergo a similar set of erroneous transitions. If the contents of C_i are not verified at the proper time, its contents could appear unchanged, thereby the fault going

TABLE 1

POSSIBLE COUPLING FAULT COMBINATIONS

(↑,0; 1,1) - C_j coupled to C_i for zero-to-one transitions.

(↓,1; 0,0) - C_j coupled to C_i for one-to-zero transitions.

(0,↑; 1,1) - C_i coupled to C_j for zero-to-one transitions.

(1,↓; 0,0) - C_i coupled to C_j for one-to-zero transitions.

The four coupling faults between two cells can occur in any of fifteen possible combinations as follows:

1 (↑,0; 1,1)
2 (↓,1; 0,0)
3 (0,↑; 1,1)
4 (1,↓; 0,0)
5 (↑,0; 1,1) and (↓,1; 0,0)
6 (↑,0; 1,1) and (0,↑; 1,1)
7 (↑,0; 1,1) and (1,↓; 0,0)
8 (↓,1; 0,0) and (0,↑; 1,1)
9 (↓,1; 0,0) and (1,↓; 0,0)
10 (0,↑; 1,1) and (1,↓; 0,0)
11 (↑,0; 1,1), (↓,1; 0,0) and (0,↑; 1,1)
12 (↑,0; 1,1), (↓,1; 0,0) and (1,↓; 0,0)
13 (↑,0; 1,1), (0,↑; 1,1) and (1,↓; 0,0)
14 (↓,1; 0,0), (0,↑; 1,1) and (1,↓; 0,0)
15 (↑,0; 1,1), (↓,1; 0,0), (0,↑; 1,1) and (1,↓; 0,0)

NOTE: The notation used above is (C_i, C_j; C_i, C_j) corresponding to the values of both cell C_i and cell C_j at the start of and after a memory write operation. The symbols ↑ and ↓ denote zero-to-one and one-to-zero transitions, respectively, as the write operation occurring on either cell C_i or cell C_j.

undetected. The memory test algorithm developed in this paper is capable of detecting coupling faults between cells C_i and C_j occurring in any of the fifteen possible combinations.

The following general definitions are necessary in order to develop memory testing algorithms.

Definition: Every memory cell in a RAM has the following three states associated with its contents:

a) Its internal state is the actual stored contents of the memory cell.

b) Its apparent state is the value resulting from a read of the memory cell.

c) Its expected state is the presumed contents of the memory cell after a write operation, assuming no errors have occurred.

Definition: A memory fault is said to have been detected when a difference between the expected state and the apparent state of one of the memory cells under test occurs. (Suk 1978).

Although the memory tests to be derived will be capable of detecting various faults which may occur throughout the RAM chip, they were specifically developed to detect coupling faults. Other failure modes may or may not be inadvertently detected as a result of coupling testing. This test problem therefore assumes integrity in the other portions of the RAM circuitry. For completeness, tests for APIFs should be used in combination with other complementary memory tests designed to detect faults in portions of the RAM outside of the memory array.

CHAPTER 4

THE NEIGHBORHOOD MODEL

The neighborhood model assumes we are interested in detecting adjacent pattern interference faults (APIFs). An APIF is one in which writing to one cell in the memory interferes with the contents of another cell, a cell other than the one being accessed. As was stated in Chapter 2, numerous tests have been devised to detect APIFs between cells, however, these algorithms require N^2 (such as Walking Ones and Zeros) memory accesses (where N is the number of cells in the memory). This is because these tests do not take advantage of our knowledge of memory operation and architecture and presume coupling can occur between any two cells in the memory. Recently a more reasonable approach to APIFs between cells was introduced by J. P. Hayes (1975) called the neighborhood model.

The neighborhood model is now widely accepted as a practical and effective bound on the extent of pattern interference between cells. It is based on the premise that the types of physical phenomenum which can cause APIFs to occur will predominantly affect cells which are

immediately adjacent to one another. The five-cell neighborhood model is made up of a single center cell surrounded by four cells, one each on the top, the bottom, the left and the right. It assumes that the only APIFs affecting the center cell occur as a result of writing to one of the four surrounding cells. The center cell, therefore, is not vulnerable to APIFs from non-neighborhood cells, or at least if such an APIF exists, the extent of the fault is such that it will also occur within the neighborhood.

The five-cell neighborhood model has been used predominantly since larger neighborhoods are significantly more complicated (Hayes 1975; Nair 1978; Nickel 1980). Some investigation into APIFs in a nine-cell neighborhood has been performed by V. P. Srini (1976), however, the memory patterns used were not derived analytically and limited fault coverage is achieved.

The following definitions are required to develop a formal neighborhood model:

Definition: Let the memory under test (MUT) have a two-dimensional array organization with dimensions n cells by m cells, where n, $m \geq 3$. Let a specified cell in the MUT be designated the _center cell_. The center cell and the four cells adjacent to the

center cell (if they exist) located to the top, bottom, left and right will be designated _neighborhood cells_ or just a _neighborhood_. The four neighborhood cells excluding the center cell will be referred to as the _deleted neighborhood_.

Let C represent the center cell and T, B, L and R represent the deleted neighborhood cells top, bottom, left and right, respectively. A single memory cell in the MUT may assume a dynamic state during each memory operation, while the remaining memory cells assume a static state. The following definition introduces four symbols convenient to the subsequent discussion.

Definition: There are four possible states for each memory cell during a memory operation as follows:

1. Static state zero, denoted by "0", is a state of a memory cell when the memory cell keeps the content zero from the beginning till the end of the memory cycle.
2. Static state one, denoted by "1", is a state of a memory cell when the memory cell keeps the content one from the beginning till the end of the memory cycle.

3. Dynamic state one, denoted by "↑", is the state of a memory cell after the memory cell changes its content from zero to one as the result of a memory write operation.
4. Dynamic state zero, denoted by "↓", is the state of a memory cell after the memory cell changes its content from one to zero as the result of a memory write operation (Suk 1978).

Definition: The Adjacent Neighborhood Pattern (ANP) for a five-cell neighborhood is the ordered quintuple, TBLRC, where T, B, L, R and C are the states of the top, bottom left, right and center cells, respectively. In order to verify the MUT for all APIFs, all valid ANPs must be generated in the five-cell neighborhood. This will ensure that if an APIF exists between the center cell and its deleted neighborhood, it will exhibit itself.

For example, the quintuple (↓0001) is an ANP where the top cell, T, of the neighborhood is in dynamic state zero to one while B, L, R and C are in static states 0, 0, 0 and 1, respectively. The contents of the center cell can be subsequently verified to determine if this

ANP exhibits an APIF. It can be easily seen that there are 64 distinct ANPs, given in Table 2, for the five-cell neighborhood.

The neighborhood pattern provides a convenient technique for detecting coupling faults between adjacent cells. The 64 ANPs determine if the center cell, C, is coupled to any of the deleted neighborhood cells T, B, L, and R (see Chapter 3 on the Fault Model).

CHAPTER 5

BOUNDING THE TEST PROBLEM

The vast majority of RAM test applications are concerned with fault detection and not localization. Determining the location of a fault is generally not of interest since the IC devices themselves are not repairable. In addition, testing is normally terminated upon detection of the first fault since identifying additional failures is of little consequence. However, device manufacturers themselves may require the capability to localize failures since they may point out defects in the RAM design or the manufacturing process itself (Cocking 1975; Fee 1977). They are not only interested in the ANP and the location of the affected center cell, but they will normally continue the test to completion in an attempt to find additional failures. Fault location can easily be determined by extending the test algorithm in a manner which will be described later.

Before developing memory test algorithms to detect APIFs, it is useful to determine the minimum number of memory operations required. In this way it can be

assessed whether the algorithm is indeed optimal and if not, the number of excess operations required. In order to simplify the derivation of the lower bound on the total number of memory operations, we will assume that all memory cells are surrounded by four deleted neighborhood cells. The fact that cells at the corners and edges of the rectangular memory array have less than four deleted neighborhood cells will be dealt with by assuming that cells at the top edge of the memory array are adjacent to cells at the bottom edge and cells at the left edge are adjacent to those at the right edge.

The following lemmas and proofs establish the minimum number of operations required to implement an optimal test algorithm to detect APIFs. The proof given results from inspection of Tables 2 and 3.

Lemma 1: To generate all ANPs, 16 write operations per cell are required.

Proof: Since the application of each ANP requires one write operation, 64 write operations on the four deleted neighborhood cells will require 16 write operations each. This can be further verified by counting the number of transitions occurring for any of the cells T, B, L or R shown in Table 2.

Lemma 2: To detect and locate all APIFs between the deleted neighborhood cells and the center cell, 64 read operations are minimally required.

Proof: In order to locate the failure, we must know precisely which ANP interfered with the center cell. To accomplish this, the contents of the center cell must be verified after applying each ANP. Therefore, since there are 64 ANPs, each center cell must be read at least 64 times.

Lemma 3: To detect (but not locate) all APIFs between the deleted neighborhood cells and the center cell, 24 read operations are minimally required.

Proof: Referring back to Table 2, 32 of the ANPs involve zero to one transitions and 32 involve one to zero transitions (shown in the top and bottom halves of Table 2, respectively). Any of the zero to one transitions can cause the center cell to erroneously go from a zero to a one if an APIF exists and similarly any of the one to zero transitions can cause the center cell to erroneously change in value from a one to a zero. Therefore, the center cell must be verified prior to changing over from one-to-zero transitions to zero-to-one transitions (and vice versa). This is because

TABLE 2

ADJACENT NEIGHBORHOOD PATTERNS FOR A
FIVE CELL NEIGHBORHOOD

T	B	L	R	C	T	B	L	R	C	T	B	L	R	C	T	B	L	R	C
↑	0	0	0	0	0	↑	0	0	0	0	0	↑	0	0	0	0	0	↑	0
↑	0	0	1	0	0	↑	0	1	0	0	0	↑	1	0	0	0	1	↑	0
↑	0	1	0	0	0	↑	1	0	0	0	1	↑	0	0	0	1	0	↑	0
↑	0	1	1	0	0	↑	1	1	0	0	1	↑	1	0	0	1	1	↑	0
↑	1	0	0	0	1	↑	0	0	0	1	0	↑	0	0	1	0	0	↑	0
↑	1	0	1	0	1	↑	0	1	0	1	0	↑	1	0	1	0	1	↑	0
↑	1	1	0	0	1	↑	1	0	0	1	1	↑	0	0	1	1	0	↑	0
↑	1	1	1	0	1	↑	1	1	0	1	1	↑	1	0	1	1	1	↑	0
↓	1	1	1	1	1	↓	1	1	1	1	1	↓	1	1	1	1	1	↓	1
↓	1	1	0	1	1	↓	1	0	1	1	1	↓	0	1	1	1	0	↓	1
↓	1	0	1	1	1	↓	0	1	1	1	0	↓	1	1	1	0	1	↓	1
↓	1	0	0	1	1	↓	0	0	1	1	0	↓	0	1	1	0	0	↓	1
↓	0	1	1	1	0	↓	1	1	1	0	1	↓	1	1	0	1	1	↓	1
↓	0	1	0	1	0	↓	1	0	1	0	1	↓	0	1	0	1	0	↓	1
↓	0	0	1	1	0	↓	0	1	1	0	0	↓	1	1	0	0	1	↓	1
↓	0	0	0	1	0	↓	0	0	1	0	0	↓	0	1	0	0	0	↓	1

NOTE: 1. ↑ DENOTES 0 TO 1 TRANSITION (DYNAMIC STATE ONE)
↓ DENOTES 1 TO 0 TRANSITION (DYNAMIC STATE ZERO)
2. EACH ROW OF FIVE ENTRIES TBLRC REPRESENTS ONE ANP
3. T - TOP, B - BOTTOM, L - LEFT, R - RIGHT, C - CENTER

if multiple APIFs exist, the contents of the center cell could be interfered with and yet go undetected due to two APIFs involving opposite transitions canceling each other out. From Table 3 it can be seen that 12 verification points exist for each set of 32 ANPs. Thus there are 24 points at which the contents of the center cells must be read.

Lemma 4: To initialize the memory for generating all 64 ANPs, 25 memory operations per cell are required.

Proof: First of all, each of the cells in the memory must be initialized to a predetermined value. Secondly, in order to generate the 64 ANPs, the state of the center cells must be opposite to that of the direction of the deleted neighborhood cell transitions. At each of the 24 center cell verification points (see Lemma 3), the contents of the deleted neighborhood cells is opposite to that of the center cells. Therefore, prior to generating the next set of ANPs, the contents of either the deleted neighborhood cells or the center cells must be complemented. This requires that each cell in the memory be written an additional 24 times.

TABLE 3

TABLE OF WRITE OPERATIONS REQUIRED TO GENERATE ALL THIRTY-TWO ANP'S INVOLVING ZERO TO ONE TRANSITIONS

	T	B	L	R	
1	↑	0	0	0	*
2	1	↑	0	0	*
3	1	1	↑	0	*
4	1	1	1	↑	*
5	0	↑	0	0	*
6	0	1	↑	0	*
7	0	1	1	↑	*
8	↑	1	1	1	*
9	0	0	↑	0	*
10	0	0	1	↑	*
11	↑	0	1	1	*
12	1	↑	1	1	*
13	0	0	0	↑	*
14	↑	0	0	1	*
15	1	↑	0	1	*
16	1	1	↑	1	*
17	↑	0	0	0	
18	1	0	↑	0	*
19	1	0	1	↑	*
20	1	↑	1	1	
21	↑	0	0	0	
22	1	0	0	↑	*
23	1	0	↑	1	*
24	1	↑	1	1	
25	0	↑	0	0	
26	↑	1	0	0	*
27	1	1	0	↑	*
28	1	1	↑	1	
29	0	↑	0	0	
30	0	1	0	↑	*
31	0	1	↑	1	*
32	↑	1	1	1	
33	0	0	↑	0	
34	↑	0	1	0	*
35	1	↑	1	0	*
36	1	1	1	↑	
37	0	0	↑	0	
38	0	↑	1	0	*
39	↑	1	1	0	*
40	1	1	1	↑	
41	0	0	0	↑	
42	0	0	↑	1	*
43	0	↑	1	1	*
44	↑	1	1	1	
45	0	0	0	↑	
46	0	↑	0	1	*
47	↑	1	0	1	*
48	1	1	↑	1	

NOTE: 1) Asterisks mark the first occurrence of each of the 32 ANP's.

2) The contents of the center cells must be verified after every four ANP's (twelve times in all).

Thus, the lower bound on the number of memory operations required to detect ANIFs associated with a memory composed of N cells is:

16N	Generate ANPs
24N	Verify center cells
25N	Initialization
65N	Memory operations total

The lower bound on the number of memory operations required to detect and localize APIFs for a memory composed of N cells is:

16N	Generate ANPs
64N	Verify center cells
25N	Initialization
95N	Memory operations total

In order to obtain a minimal or near minimal test algorithm which satisfies the lower bounds stated in this chapter, the first difficulty which has to be overcome is the problem of overlap among five-cell neighborhoods. In the following chapter a method of achieving the effect of non-overlapping neighborhoods will be demonstrated.

CHAPTER 6

RAM CELL ASSIGNMENT

As discussed earlier, the RAM cell array is a rectangular matrix with dimensions n by m, where n and m are integral powers of two. An 8 x 8 memory matrix will be utilized to illustrate RAM cell assignment. The resulting concepts and algorithms are applicable to all RAMs containing a rectangular cell array, regardless of their size.

At this point the neighborhood model will be used along with the applicable sequence of operations within the neighborhood necessary to detect all APIFs. The problem at hand is that every cell of the memory is a center cell relative to those surrounding it or, in other words, the five-cell neighborhoods of the memory all overlap with one another. It is imperative that a method resulting in non-overlapping neighborhoods is developed if our algorithm to detect APIFs is to be near optimum. The alternative is to deal with each neighborhood separately, requiring several times the optimal number of operations.

The memory array can first be subdivided into two complementary arrays, each looking like a checkerboard and each containing half of the cells in the memory. The following definitions establish the cell assignments for those arrays.

Definition: Each cell of the memory array can be uniquely addressed by its row and column address. The row address will be disignated i and the column address designated j. A memory cell can be specified in the array by its address (i, j). Since the memory array consists of n rows by m columns, the row address i can range from 0 to n-1 and the column address j can range from 0 to m-1. For convention the lowest addressed cell in the memory array, (0, 0) will be located at the upper left hand corner of the matrix.

Definition: Let CEVEN (CODD) represent a set of memory cells making up the subdivided memory cell array of deleted neighborhood cells whose center cells (C) all have addresses such that the sum of the row and column addresses is even (odd). Each of the cells of CEVEN (CODD) represent deleted neighborhood cells

whereby the corresponding center cells appear in CODD (CEVEN).

Definition: Let four symbols W, X, Y and Z be assigned to each cell in the memory such that every deleted neighborhood will consist of one cell with each symbol. Letting (i, j) represent the row and column address of each cell, respectively, Table 4 illustrates one possible set of assignments of these four cells.

The resulting memory array assignment is illustrated in Figure 4. The following observations can then be made:

Observation 1: Every cell in CEVEN (CODD) contains the four deleted neighborhood cells corresponding to those center cells making up CODD (CEVEN). Furthermore, this neighborhood contains each symbol W, X, Y and Z exactly once. All four possible pattern appearances are illustrated in Figure 4.

Observation 2: Any two neighborhoods containing the same symbol in the same position, either top, bottom, left or right, are non-overlapping.

These observations will play a major role in the derivation of a test algorithm. Figure 5 gives the final memory cell assignments of both CEVEN and CODD for the

TABLE 4

MEMORY ARRAY CELL ASSIGNMENT

CEVEN:

	row i	column j
W	$0 + 4\alpha$	$1 + 4\beta$
	$2 + 4\alpha$	$3 + 4\beta$
X	$1 + 4\alpha$	$0 + 4\beta$
	$3 + 4\alpha$	$2 + 4\beta$
Y	$3 + 4\alpha$	$0 + 4\beta$
	$1 + 4\alpha$	$2 + 4\beta$
Z	$2 + 4\alpha$	$1 + 4\beta$
	$0 + 4\alpha$	$3 + 4\beta$

CODD:

	row i	column j
W	$0 + 4\alpha$	$0 + 4\beta$
	$2 + 4\alpha$	$2 + 4\beta$
X	$1 + 4\alpha$	$3 + 4\beta$
	$3 + 4\alpha$	$1 + 4\beta$
Y	$3 + 4\alpha$	$3 + 4\beta$
	$1 + 4\alpha$	$1 + 4\beta$
Z	$2 + 4\alpha$	$0 + 4\beta$
	$0 + 4\alpha$	$2 + 4\beta$

NOTE: $\alpha = 0, 1, 2, \ldots, (N/4n)-1$
$\beta = 0, 1, 2, \ldots, (N/4m)-1$

where N is the size of the memory with n rows and m columns (i.e., $N = n \cdot m = 2^k$, where k is some positive integer)

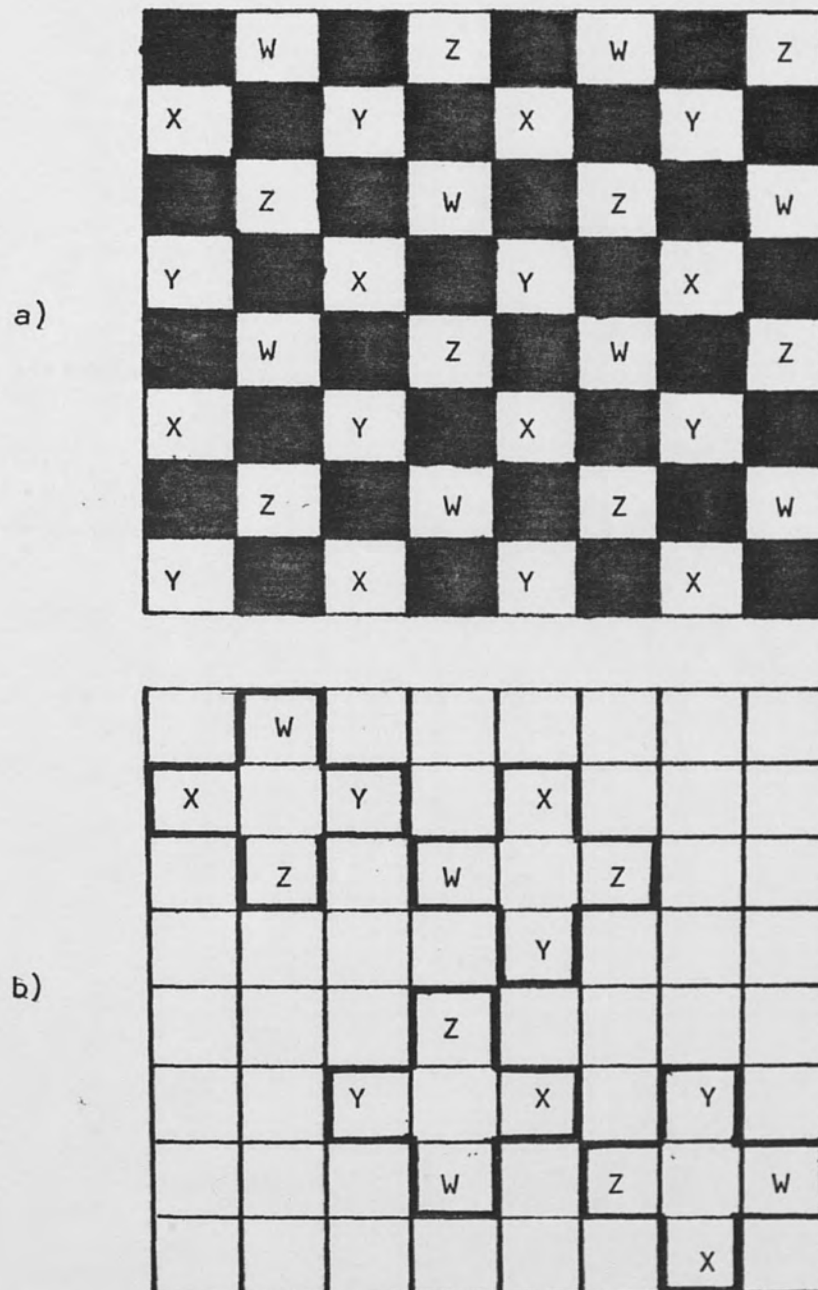
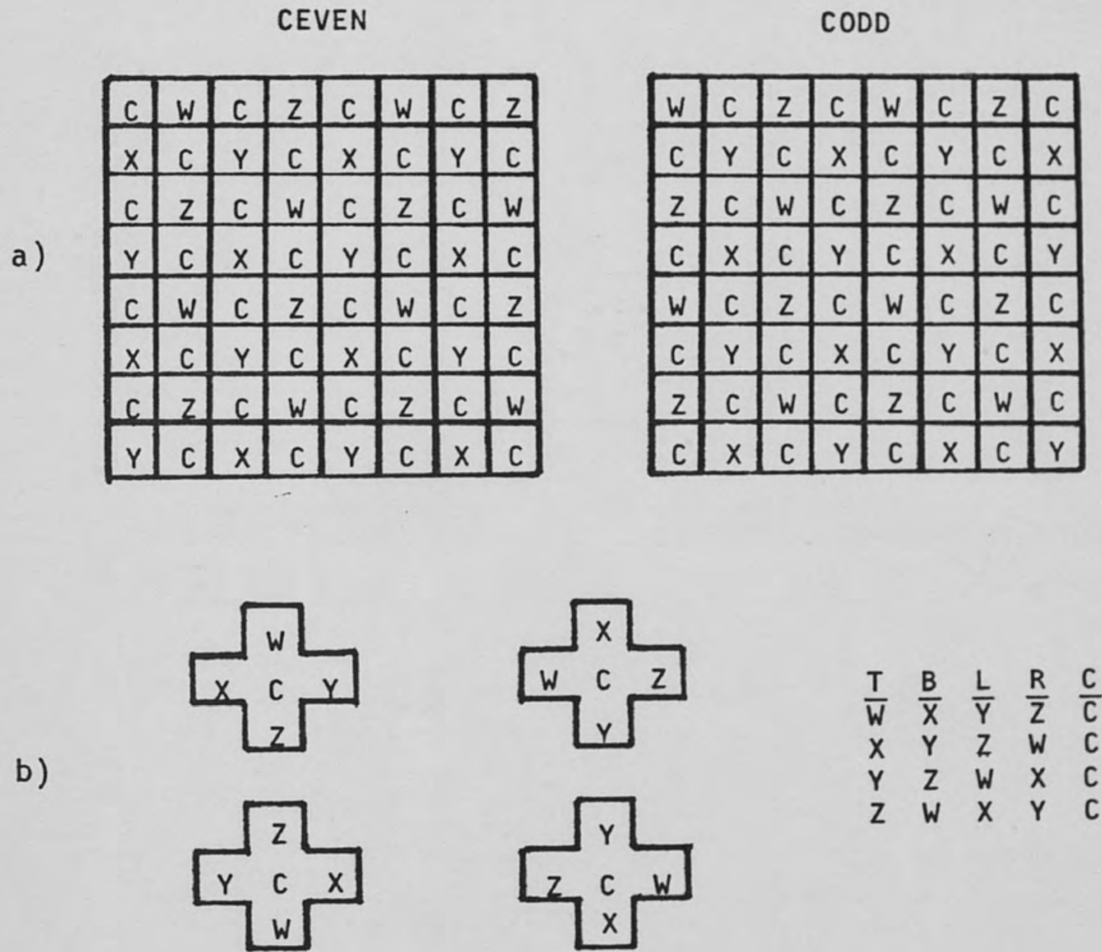


Fig. 4. Development of Five Cell Neighborhood Patterns

a) Memory Array Assignment Ceven with blackened cells representing center cells

b) Four possible Resulting Neighborhood Patterns



NOTE: T - TOP CELL, B - BOTTOM CELL
L - LEFT CELL, R - RIGHT CELL
C - CENTER CELL

Fig. 5. Cell Assignment in an 8 X 8 Memory Array

a) Array
b) Four Neighborhood Patterns

8 x 8 memory array. Note that with the center cells properly initialized, every time the contents of one of the deleted neighborhood cells is changed, one of the 64 possible ANPs (see Table 2) is generated for the corresponding center cell. It then follows that if every cell in CEVEN which is assigned the same symbol undergoes a transition, this generates one of the 64 ANPs for all N/2 cells in CODD. Thus it is possible to generate ANPs for N/2 cells by writing to N/8 cells in the MUT. The next chapter will make use of this property by determining a test sequence which will generate all 64 ANPs in this manner while requiring a minimal number of operations.

CHAPTER 7

DETERMINING A MINIMUM SEQUENCE

In order to determine the sequence of memory operations which will result in an optimal test for APIFs, it is necessary to analyze the state diagram of the four deleted neighborhood cells, W, X, Y and Z. It is desirable to make use of the theory of directed graphs, heretofore referred to as digraphs in performing this analysis. The application of digraph construction to determine a minimal sequence of memory operations was first used by D. S. Suk (1978).

In order to maintain standard terminology the source Suk referenced in his paper, "Structural Models: An Introduction to the Theory of Directed Graphs" by F. Harary, Norman, and Cartwright, (1965) was used for the definitions which follow:

Definition: A <u>relation</u> is a network of nodes and arcs in which no two distinct arcs are redundant. In the subsequent discussion we utilize the notation V_i to denote node V_i and V_iV_j to denote an arc from node V_i to V_j.

Definition: A relation is irreflexive if no node has an arc which is a loop back to itself.

Definition: A node-arc sequence is an alternating sequence of nodes and arcs such that each preceding arc of the sequence ends on the node of the succeeding arc. A node-arc sequence is written in the form: $V_1V_2...V_n$. The node V_1 is the initial node of the sequence and node V_n is the terminal node of the sequence.

Definition: A closed sequence is an arc-node sequence in which the initial and terminal nodes of the sequence are the same node. This is also referred to as a cycle.

Definition: A digraph is an irreflexive relation. In other words, it is a network with no loops or parallel arcs.

Definition: A symmetric digraph is one in which each pair of nodes is joined by two arcs, one from V_i to V_j (V_iV_j) and one from V_j to V_i (V_jV_i).

Definition: The indegree of node V_i, written $id(V_i)$, is the number of arcs entering the node V_i and the outdegree of node V_i, written $od(V_i)$ is the number of arcs leaving V_i.

Definition: A digraph is Eulerian if for every node V_i, the number of arcs entering the node is equal to the number of arcs leaving it. In other words, the indegree of node V_i is equal to the outdegree of node V_i, written $id(V_i) = od(V_i)$, for all i.

Definition: If a digraph is Eulerian, it is possible to start at any node V_i, travel along each arc exactly once, and return to node V_i. Such a cycle is called an Eulerian cycle.

Definition: A collection of cycles is said to be arc-disjoint if no two cycles have an arc in common. Two arc-disjoint cycles may, of course, have nodes in common.

Definition: A trajectory is an arc-node sequence in which no arc occurs more than once. A trajectory which contains every arc of a digraph is called arc-complete.

Definition: A digraph is called transversable if it has an arc-complete closed trajectory. A symmetric digraph is transversable.

Definition: A digraph may be decomposed into a collection of arc-disjoint subgraphs, groups of which may be joined together to form new digraphs which are subsets of the original digraph.

The four cell assignments W, X, Y and Z can be grouped to represent a four-digit binary number, WXYZ, as follows:

$$W \times 2^3$$
$$X \times 2^2$$
$$Y \times 2^1$$
$$Z \times 2^0$$

The value of WXYZ can range from 0000 thru 1111, allowing it to assume any of 16 binary values. Each of these 16 binary values can be assigned to a node on our deleted neighborhood digraph. Those nodes which are a Hamming distance one from each other (i.e., differing in only a single digit position) can be connected with bidirectional arcs. The resulting digraph is shown in Figure 6. As is seen in the figure, those nodes which are a Hamming distance of two or greater from each other are not connected by an arc. For example, the two nodes 0010 and 0110 are a Hamming distance of one from each other and connected by an arc while nodes 0010 and 0111 are a Hamming distance of two apart and therefore there is no arc between them.

The digraph resulting from the above is clearly Eulerian since the indegree is equal to the outdegree for each node. It is therefore possible to traverse each of the 64 arcs exactly once and return to the initial

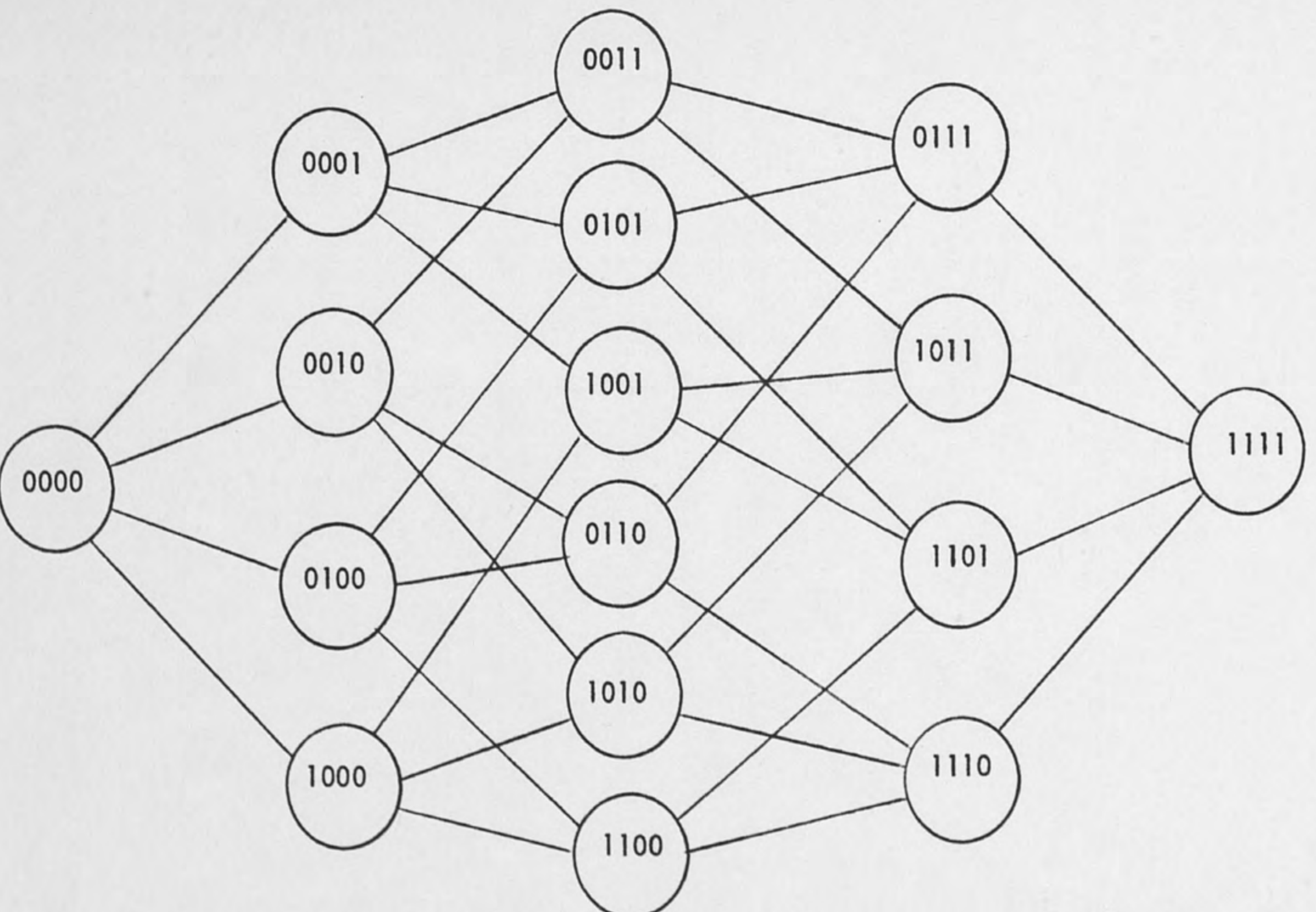


Fig. 6. Digraph of All Possible Five-Cell Neighborhood Patterns

node of the sequence. Each of the arcs represents a transition in one of the four binary digits. The arc between any two nodes WXYZ can be represented with an up or down arrow in the digit position which is in transition. An arrow pointing upwards will indicate a change in value from zero to one while one pointing downwards will indicate a change from one to zero. The 64 arcs of the deleted neighborhood digraph correspond to the ANPs in Table 2 discussed earlier.

Since the deleted neighborhood digraph is symmetric and Eulerian, there exists an arc-complete trajectory which will generate all 64 ANPs with a minimum number of memory write operations to the neighborhood cells. Referring back to our neighborhood model (Chapter 4), it was shown that to detect APIFs, the state of the center cell must be initially opposite to the direction of the transitions occurring within the deleted neighborhood. In order to minimize the total number of memory operations, one must therefore also minimize the number of read and write accesses to the center cells. The center cells must be verified after one or more deleted neighborhood transitions in the same direction have taken place and before the value of the deleted neighborhood cells is complemented to begin transitions in the opposite direction. The problem is then to minimize the number of

changes in transition direction that must take place while traversing the digraph.

The initial step is to decompose the deleted neighborhood digraph into a series of subgraphs, each of which is Eulerian as shown in Figure 7. The arcs of each subgraph are then further subdivided into groups of arc-disjoint sequences whose transitions are all occurring in the same direction. One such set of arc-disjoint sequences is illustrated in Table 5. It is possible to group the 16 arcs given in Table 5 a) with those in Table 5 b) resulting in 24 sets of transition sequences occurring in the same direction. These 24 arc-disjoint sequences are illustrated in Figure 8.

These 24 arc-disjoint sequences of Table 8 can be utilized by joining them into a series of closed sequences forming an Eulerian cycle for the deleted neighborhood digraph. The sequence in which the 64 arcs are traversed form the basis of our test algorithm to detect APIFs. In the process of transversing all 64 arcs, the contents of the center cell must be verified and changed a minimum of 24 times in all (see Table 3). A total of 88 transition operations are required, counting the 24 required center cell transitions. These 88 entries are required to generate all ANPs and are illustrated in Table 6.

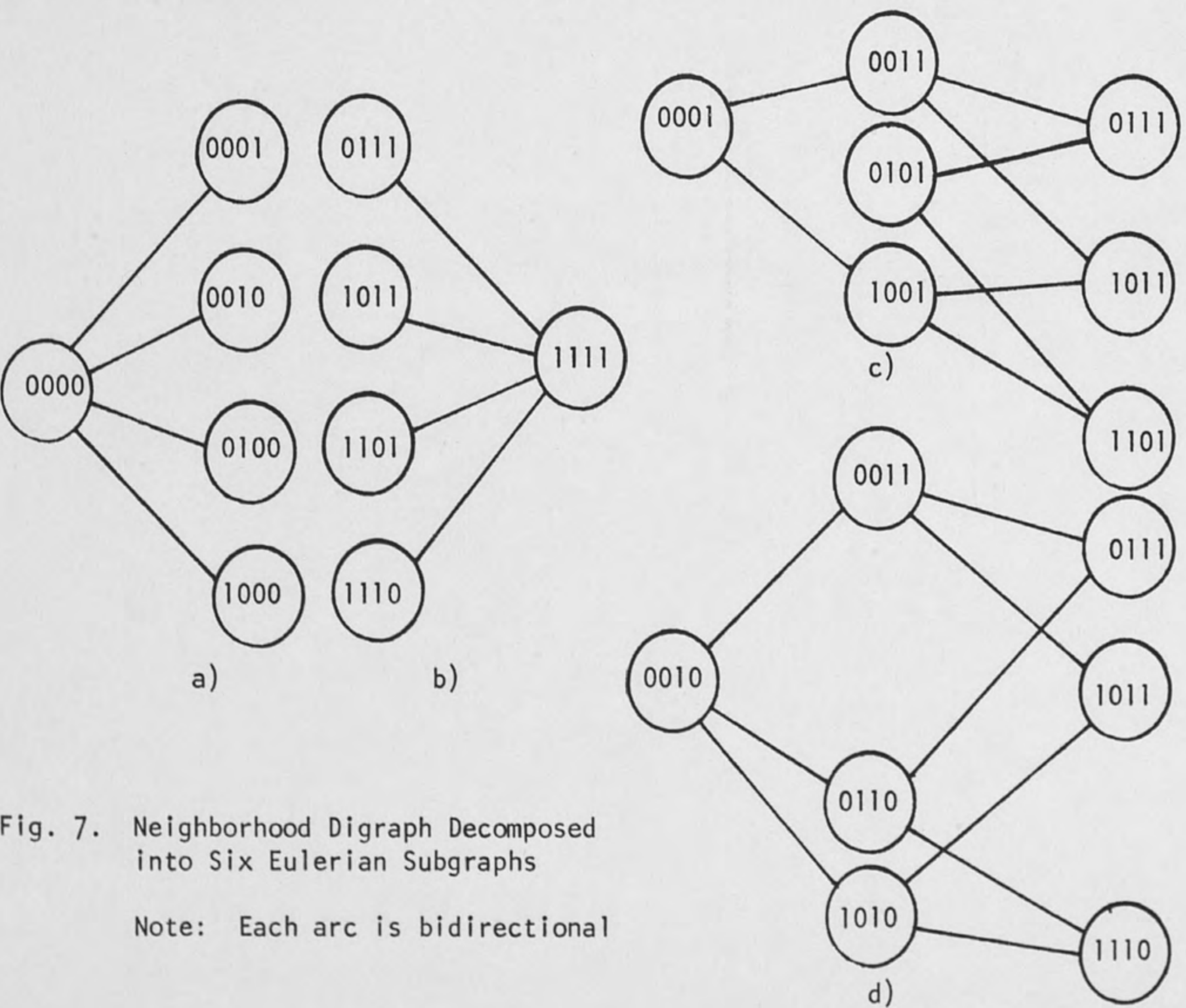


Fig. 7. Neighborhood Digraph Decomposed into Six Eulerian Subgraphs

Note: Each arc is bidirectional

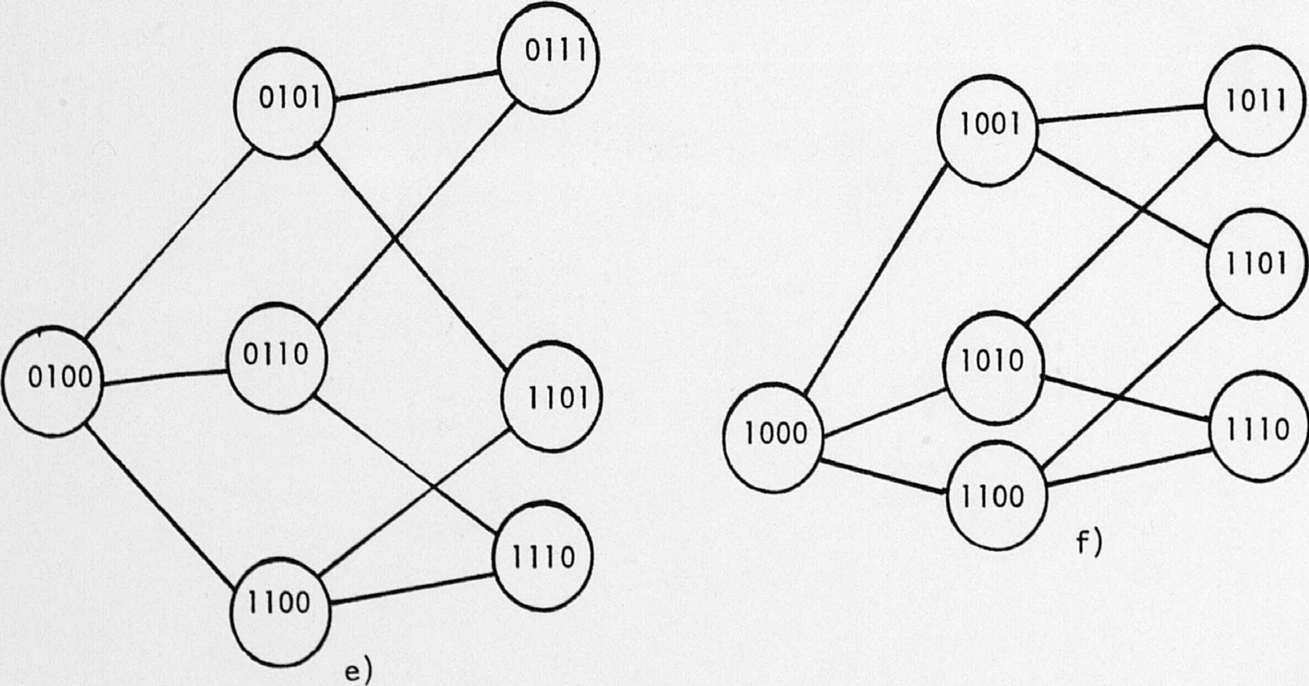


Fig. 7. (Con't)

TABLE 5

SIXTY-FOUR ANP'S CORRESPONDING TO PATHS ON THE STATE DIAGRAM OF A FIVE CELL NEIGHBORHOOD

W	X	Y	Z		W	X	Y	Z
0	0	0	0	←	0	0	0	1
0	0	0	0	←	0	0	1	0
0	0	0	0	←	0	1	0	0
0	0	0	0	←	1	0	0	0
1	1	1	1	←	1	1	1	0
1	1	1	1	←	1	1	0	1
1	1	1	1	←	1	0	1	1
1	1	1	1	←	0	1	1	1

W	X	Y	Z		W	X	Y	Z
0	0	0	0	→	0	0	0	1
0	0	0	0	→	0	0	1	0
0	0	0	0	→	0	1	0	0
0	0	0	0	→	1	0	0	0
1	1	1	1	→	1	1	1	0
1	1	1	1	→	1	1	0	1
1	1	1	1	→	1	0	1	1
1	1	1	1	→	0	1	1	1

a)

W	X	Y	Z		W	X	Y	Z		W	X	Y	Z
0	0	0	1	←	0	0	1	1	←	0	1	1	1
0	0	0	1	←	1	0	0	1	←	1	0	1	1
0	0	0	1	←	0	1	0	1	←	1	1	0	1
0	0	1	0	←	0	1	1	0	←	0	1	1	1
0	0	1	0	←	0	0	1	1	←	1	0	1	1
0	0	1	0	←	1	0	1	0	←	1	1	1	0
0	1	0	0	←	0	1	0	1	←	0	1	1	1
0	1	0	0	←	1	1	0	0	←	1	1	0	1
0	1	0	0	←	0	1	1	0	←	1	1	1	0
1	0	0	0	←	1	0	1	0	←	1	0	1	1
1	0	0	0	←	1	0	0	1	←	1	1	0	1
1	0	0	0	←	1	1	0	0	←	1	1	1	0

W	X	Y	Z		W	X	Y	Z		W	X	Y	Z
0	0	0	1	→	0	1	0	1	→	0	1	1	1
0	0	0	1	→	0	0	1	1	→	1	0	1	1
0	0	0	1	→	1	0	0	1	→	1	1	0	1
0	0	1	0	→	0	0	1	1	→	0	1	1	1
0	0	1	0	→	1	0	1	0	→	1	0	1	1
0	0	1	0	→	0	1	1	0	→	1	1	1	0
0	1	0	0	→	0	1	1	0	→	0	1	1	1
0	1	0	0	→	0	1	0	1	→	1	1	0	1
0	1	0	0	→	1	1	0	0	→	1	1	1	0
1	0	0	0	→	1	0	0	1	→	1	0	1	1
1	0	0	0	→	1	1	0	0	→	1	1	0	1
1	0	0	0	→	1	0	1	0	→	1	1	1	0

b)

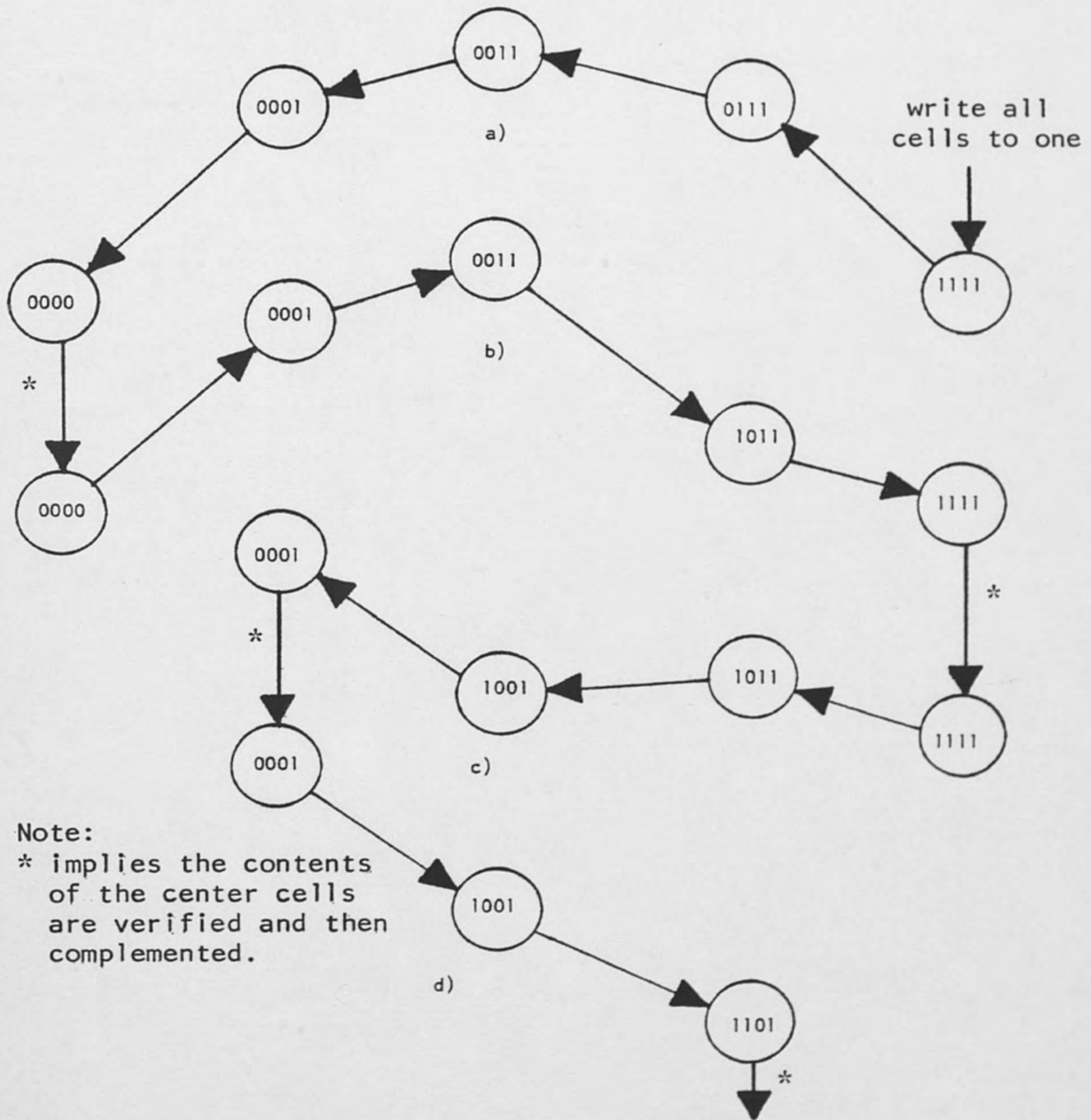


Fig. 8. Twenty-Four Arc-Disjoint Sequences Composing the Neighborhood Digraph

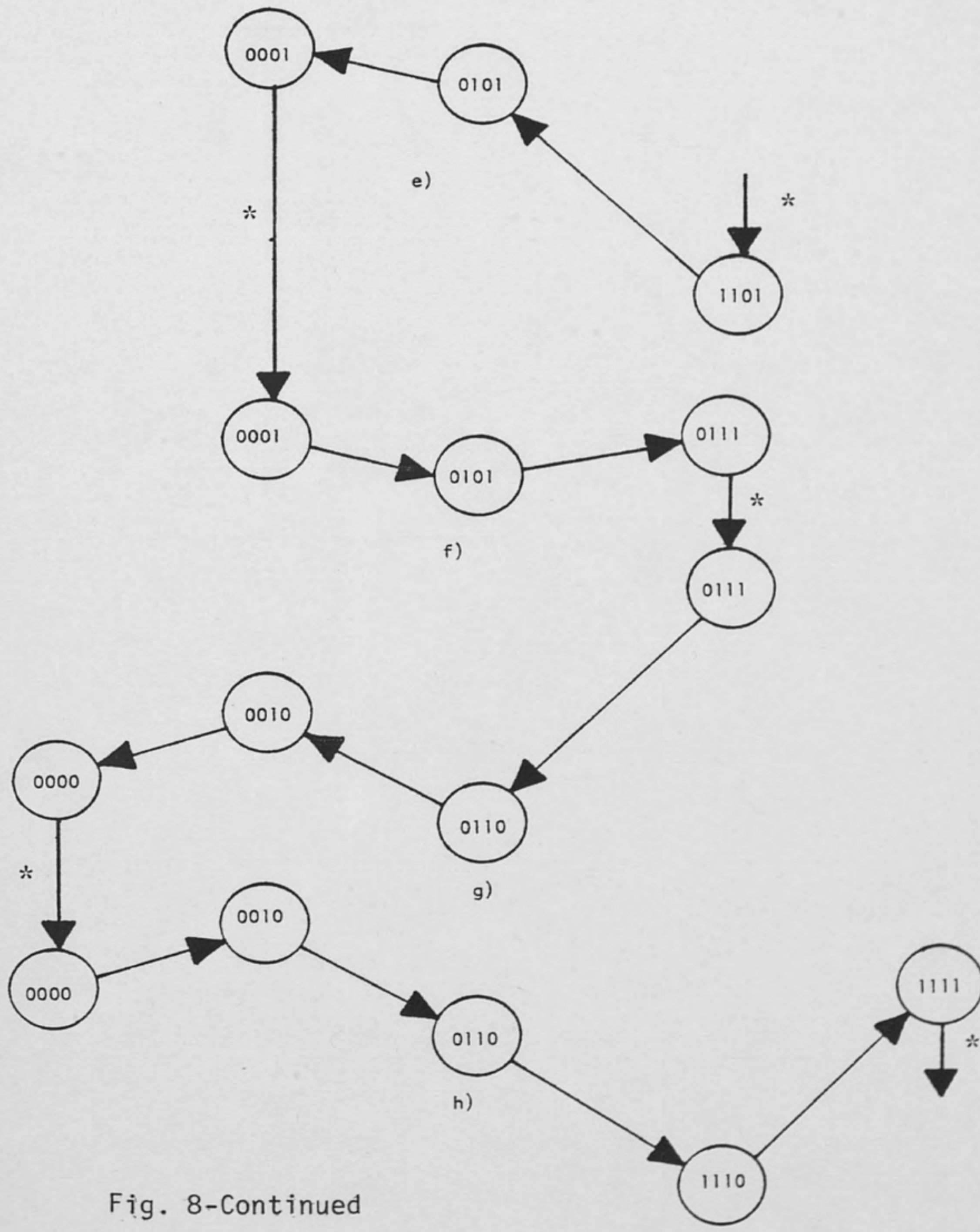


Fig. 8-Continued

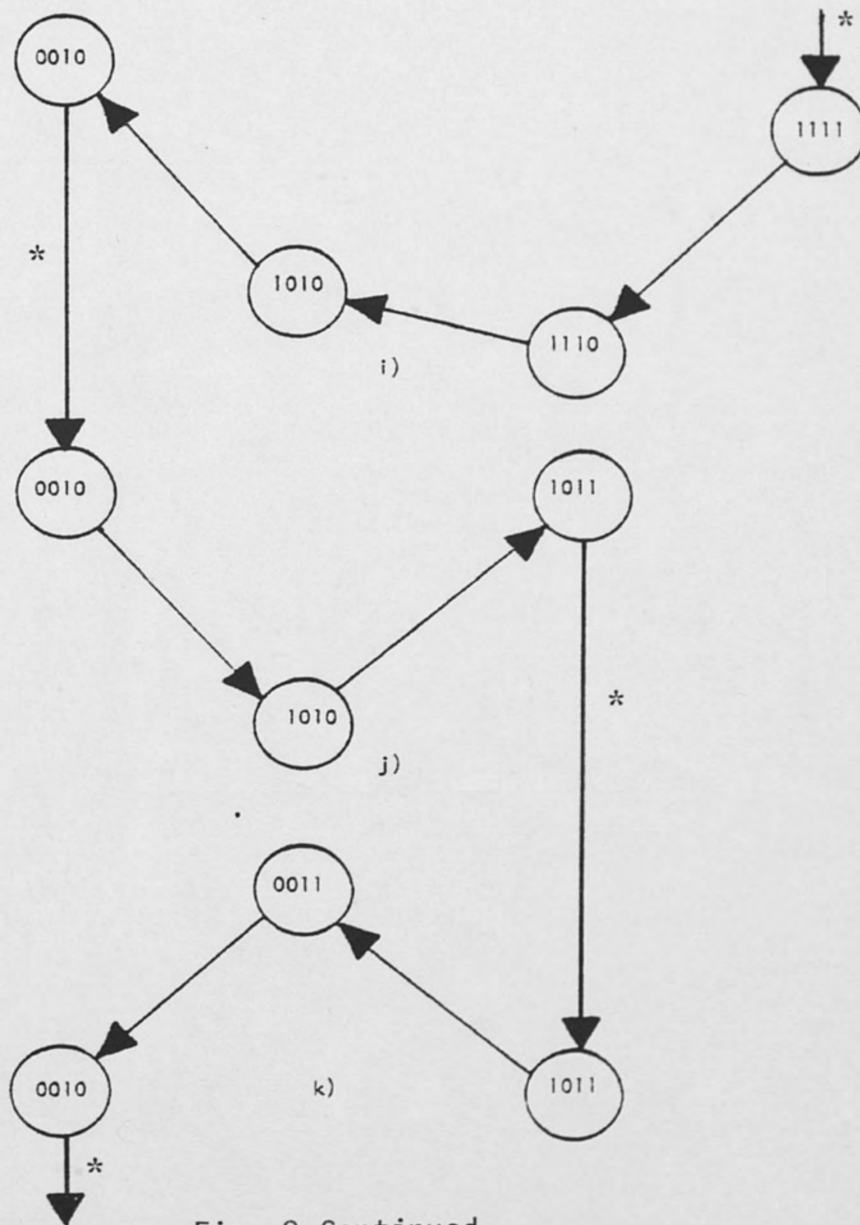


Fig. 8-Continued

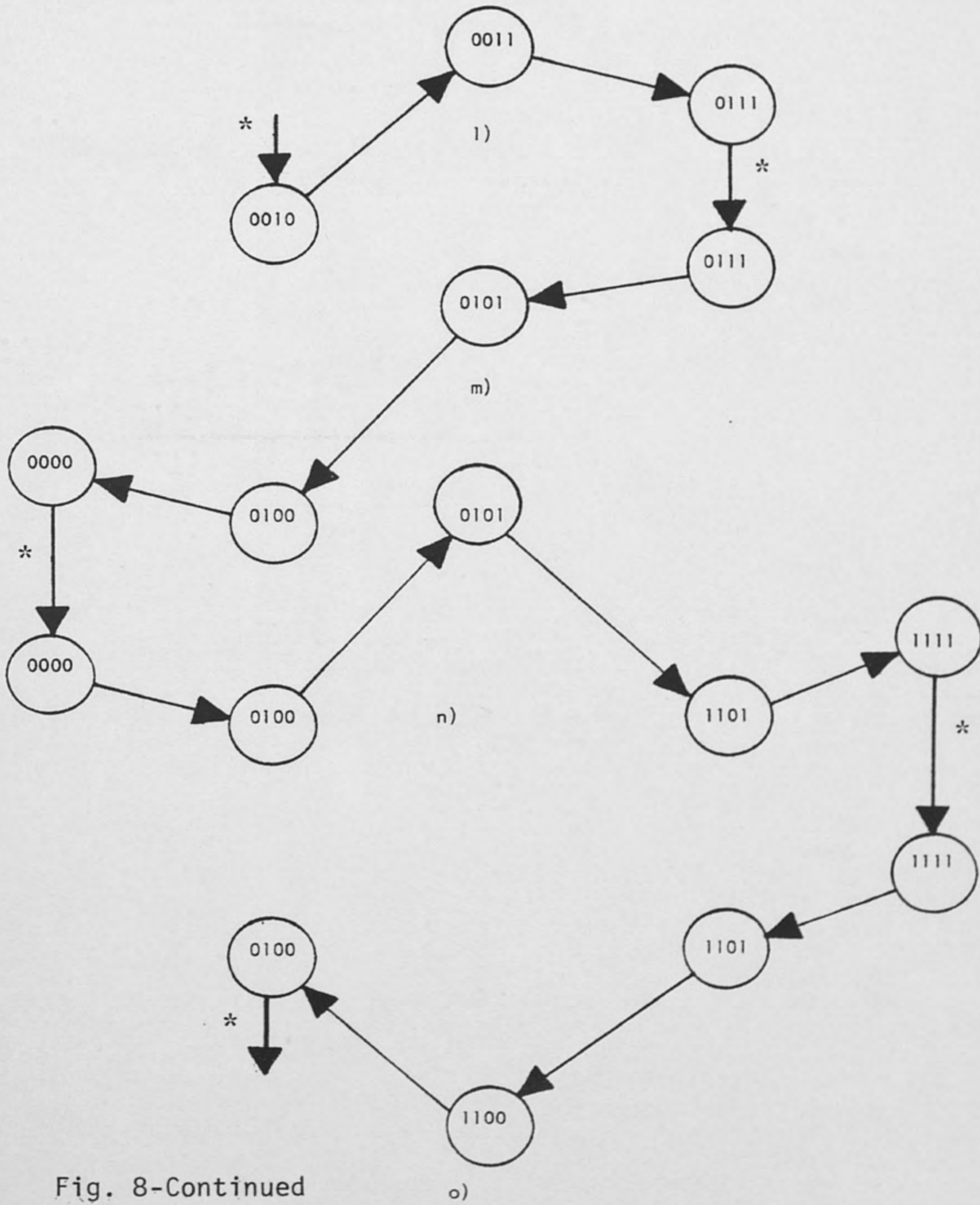


Fig. 8-Continued

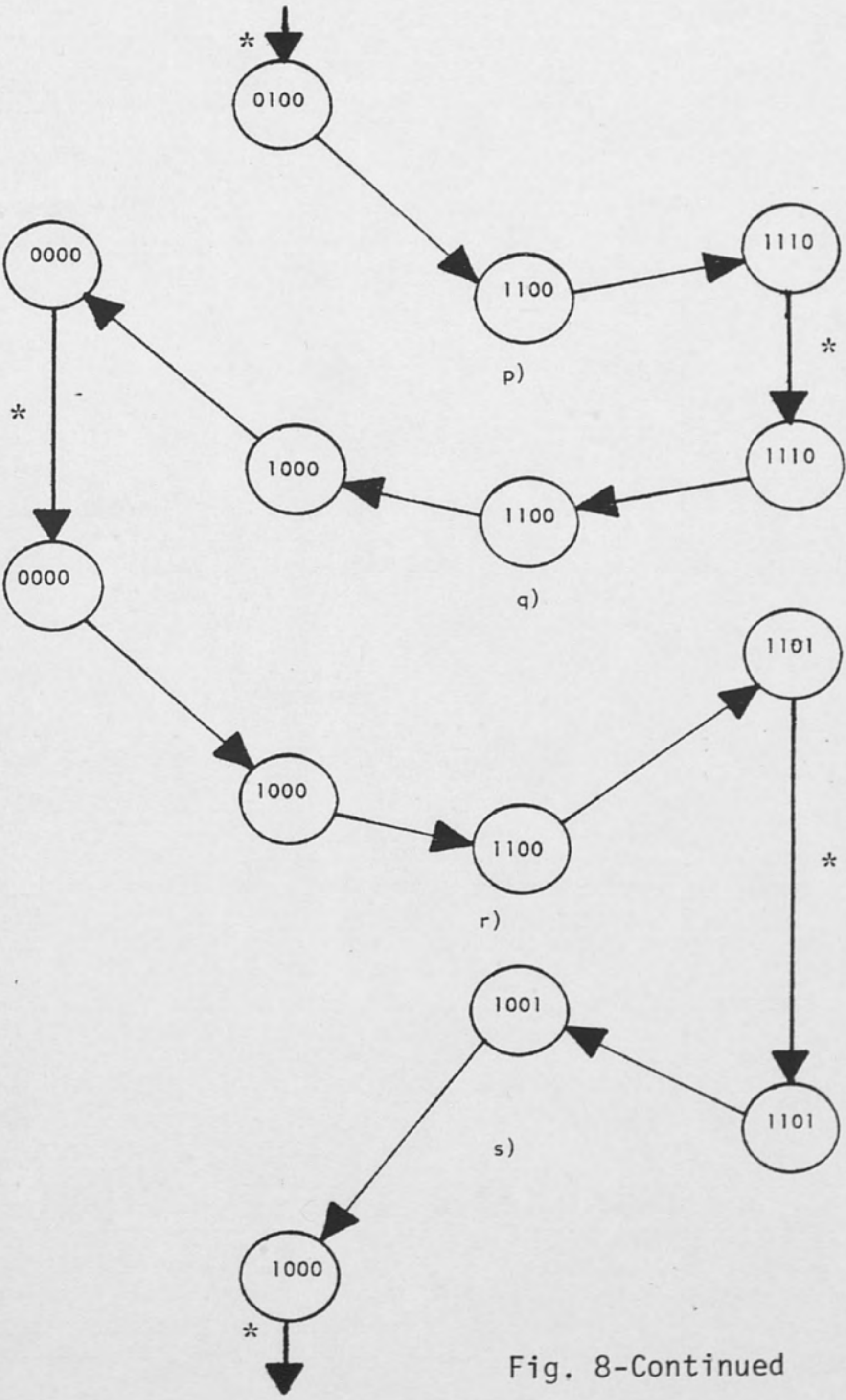


Fig. 8-Continued

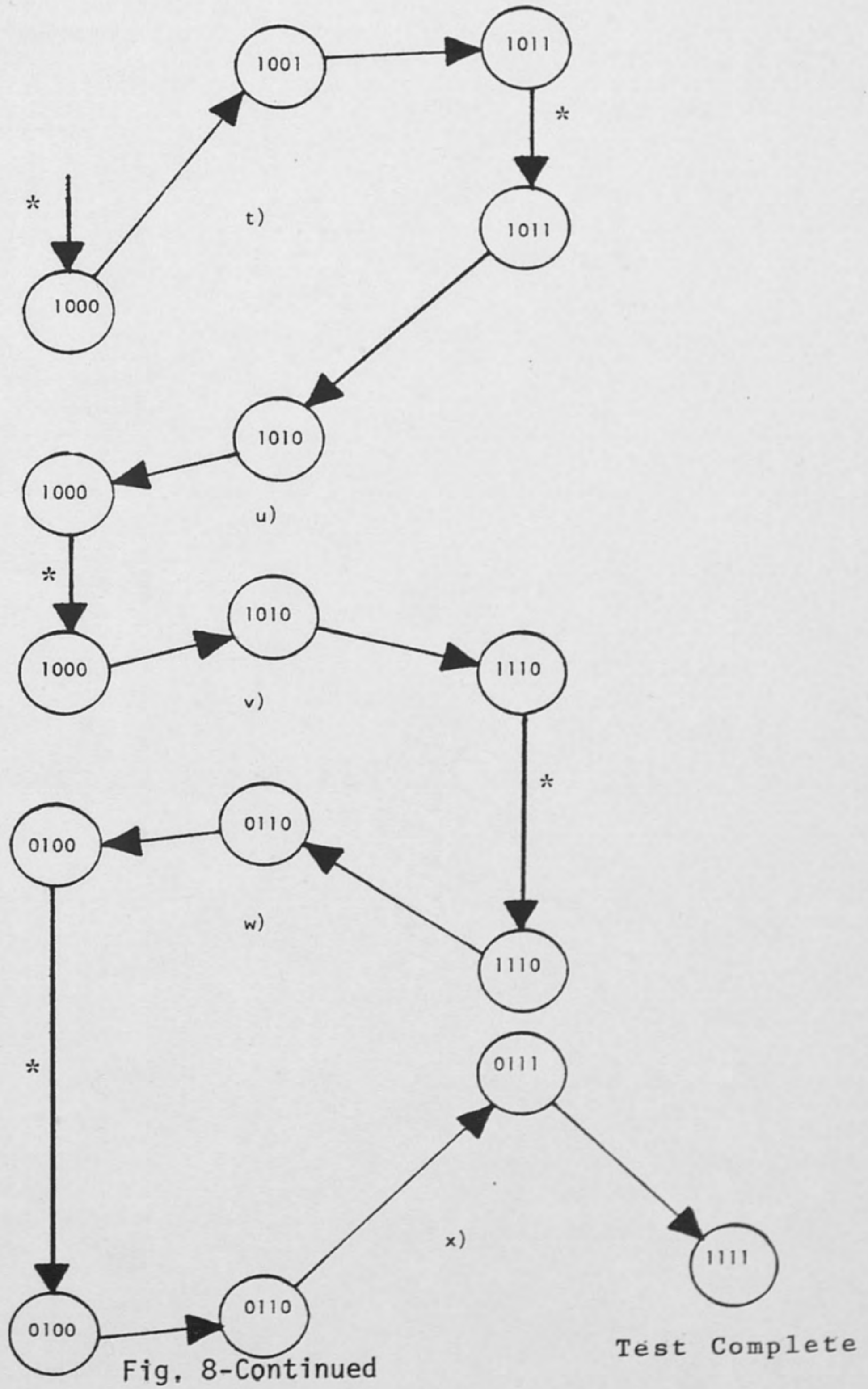


Fig. 8-Continued

TABLE 6

ANP OPERATIONAL SEQUENCE TABLE

	W	X	Y	Z	C
1	↓	1	1	1	1
2	0	↓	1	1	1
3	0	0	↓	1	1
4	0	0	0	↓	1
5	0	0	0	0	↓
6	0	0	0	↑	0
7	0	0	↑	1	0
8	↑	0	1	1	0
9	1	↑	1	1	0
10	1	1	1	1	↑
11	1	↓	1	1	1
12	1	0	↓	1	1
13	↓	0	0	1	1
14	0	0	0	1	↓
15	↑	0	0	1	0
16	1	↑	0	1	0
17	1	1	0	1	↑
18	↓	1	0	1	1
19	0	↓	0	1	1
20	0	0	0	1	↓
21	0	↑	0	1	0
22	0	1	↑	1	0

	W	X	Y	Z	C
23	0	1	1	1	↑
24	0	1	1	↓	1
25	0	↓	1	0	1
26	0	0	↓	0	1
27	0	0	0	0	↓
28	0	0	↑	0	0
29	0	↑	1	0	0
30	↑	1	1	0	0
31	1	1	1	↑	0
32	1	1	1	1	↑
33	1	1	1	↓	1
34	1	↓	1	0	1
35	↓	0	1	0	1
36	0	0	1	0	↓
37	↑	0	1	0	0
38	1	0	1	↑	0
39	1	0	1	1	↑
40	↓	0	1	1	1
41	0	0	1	↓	1
42	0	0	1	0	↓
43	0	0	1	↑	0
44	0	↑	1	1	0

	W	X	Y	Z	C
45	0	1	1	1	↑
46	0	1	↓	1	1
47	0	1	0	↓	1
48	0	↓	0	0	1
49	0	0	0	0	↓
50	0	↑	0	0	0
51	0	1	0	↑	0
52	↑	1	0	1	0
53	1	1	↑	1	0
54	1	1	1	1	↑
55	1	1	↓	1	1
56	1	1	0	↓	1
57	↓	1	0	0	1
58	0	1	0	0	↓
59	↑	1	0	0	0
60	1	1	↑	0	0
61	1	1	1	0	↑
62	1	1	↓	0	1
63	1	↓	0	0	1
64	↓	0	0	0	1
65	0	0	0	0	↓
66	↑	0	0	0	0

	W	X	Y	Z	C
67	1	↑	0	0	0
68	1	1	0	↑	0
69	1	1	0	1	↑
70	1	↓	0	1	1
71	1	0	0	↓	1
72	1	0	0	0	↓
73	1	0	0	↑	0
74	1	0	↑	1	0
75	1	0	1	1	↑
76	1	0	1	↓	1
77	1	0	↓	0	1
78	1	0	0	0	↓
79	1	0	↑	0	0
80	1	↑	1	0	0
81	1	1	1	0	↑
82	↓	1	1	0	1
83	0	1	↓	0	1
84	0	1	0	0	↓
85	0	1	↑	0	0
86	0	1	1	↑	0
87	↑	1	1	1	0
88	1	1	1	1	↑

It is now possible to develop an algorithm to detect APIFs utilizing the ANP Operational Sequence given in Table 6. The resulting three-step procedure to detect APIFs, denoted DAPIF, is given in Table 7. The first step is to initialize the memory. Steps two and three then consist of executing each of the ANP Operational Sequences for memory array assignments CEVEN and CODD, respectively. As discussed earlier, at the end of each sequence of deleted neighborhood cell transitions (in the same direction), the center cells are first verified and then written to their complement value. The number of memory operations required to perform algorithm DAPIF are shown in Table 8. For the neighborhood model under test, this test length, 65N, is optimal.

An example of algorithm DAPIF applied to a 4 x 4 memory array composed of 16 cells is given in Figure 9. For a memory of this size, 176 steps composed of (65) (16) = 1040 memory operations are required. In other words, after performing these 1040 memory operations, if all tests were successfully completed, the 16 cell memory can be presumed free of APIFs for the five-cell neighborhood model presented in this paper. The total number of memory operations required is demonstrated to be 65N, which is in fact the minimal number as shown in Chapter 5.

TABLE 7

ALGORITHM TO DETECT APIFs (DAPIF)

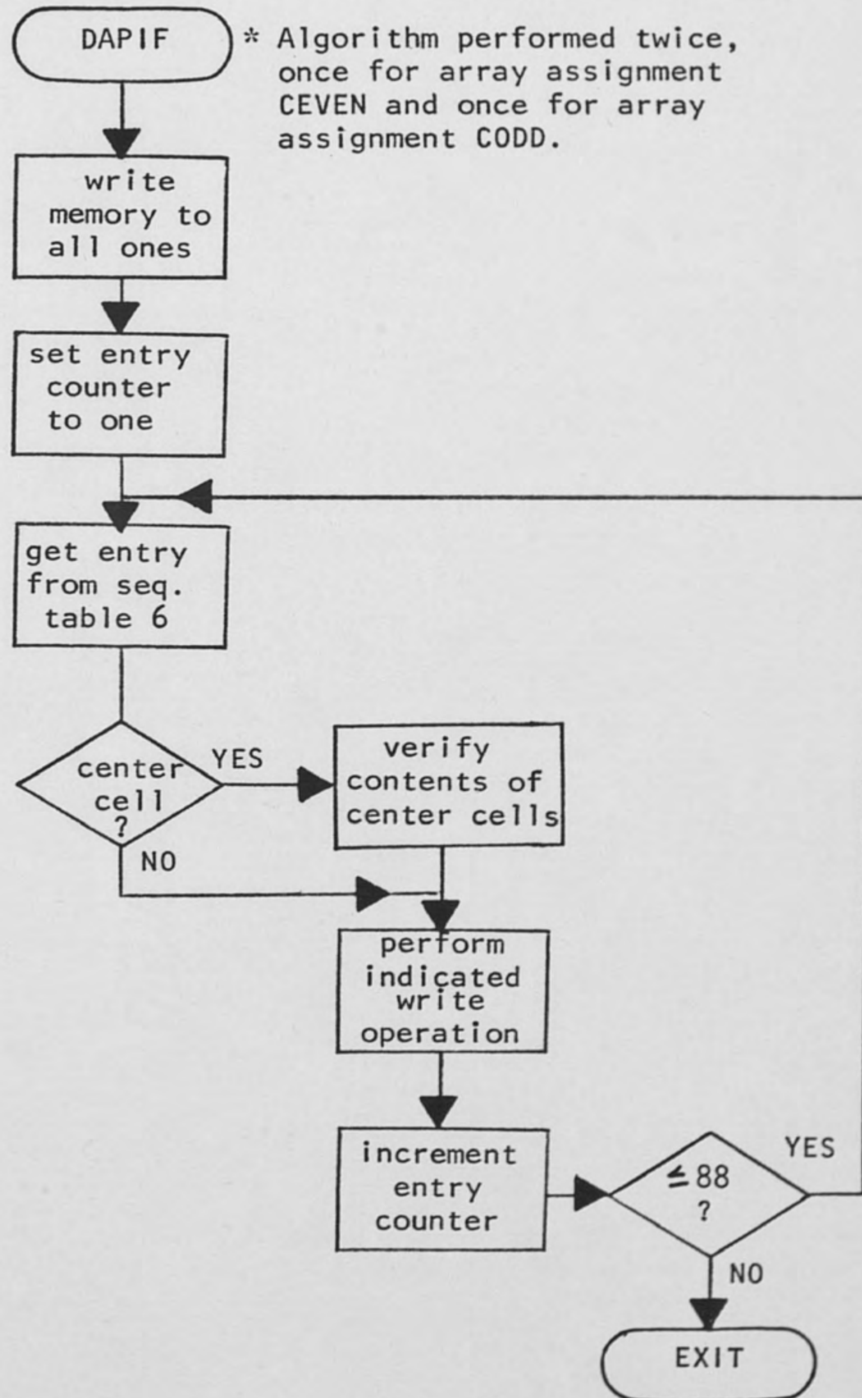


TABLE 8

ALGORITHM DAPIF MEMORY ACCESS TABULATION

	WRITE		READ C	TOTAL MEMORY OPERATIONS
	W, X, Y OR Z	C		
STEP 1	4 X N/8 = N/2	1 X N/2 = N/2	0	N
STEP 2	64 X N/8 = 8N	24 X N/2 = 12N	24 X N/2 = 12N	32N
STEP 3	64 X N/8 = 8N	24 X N/2 = 12N	24 X N/2 = 12N	32N
TOTAL	16.5 N	24.5 N	24 N	65N

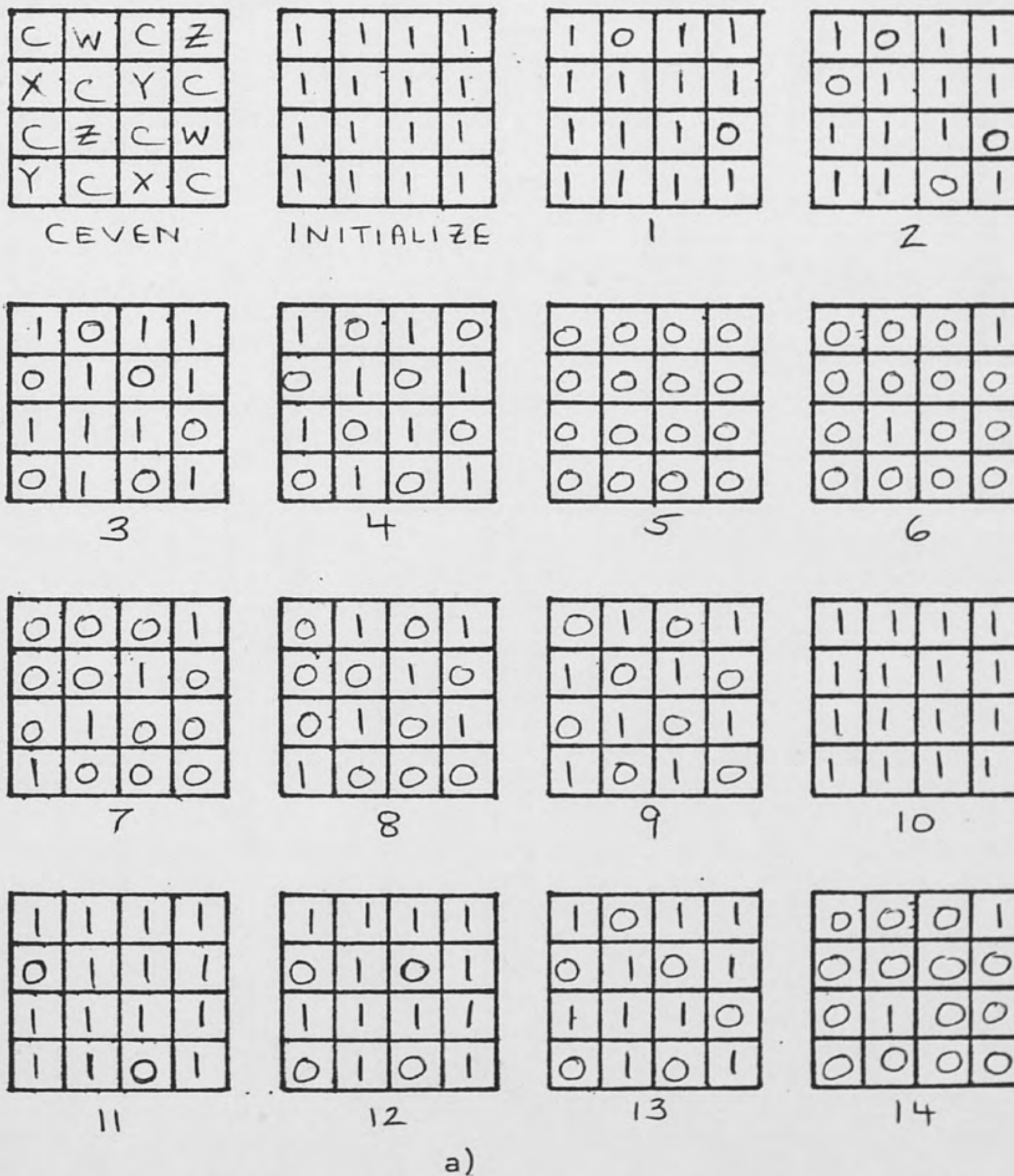


Fig. 9. Algorithm DAPIF applied to a 4 x 4 memory cell array

Note: Step numbers correspond to entries in Table 6

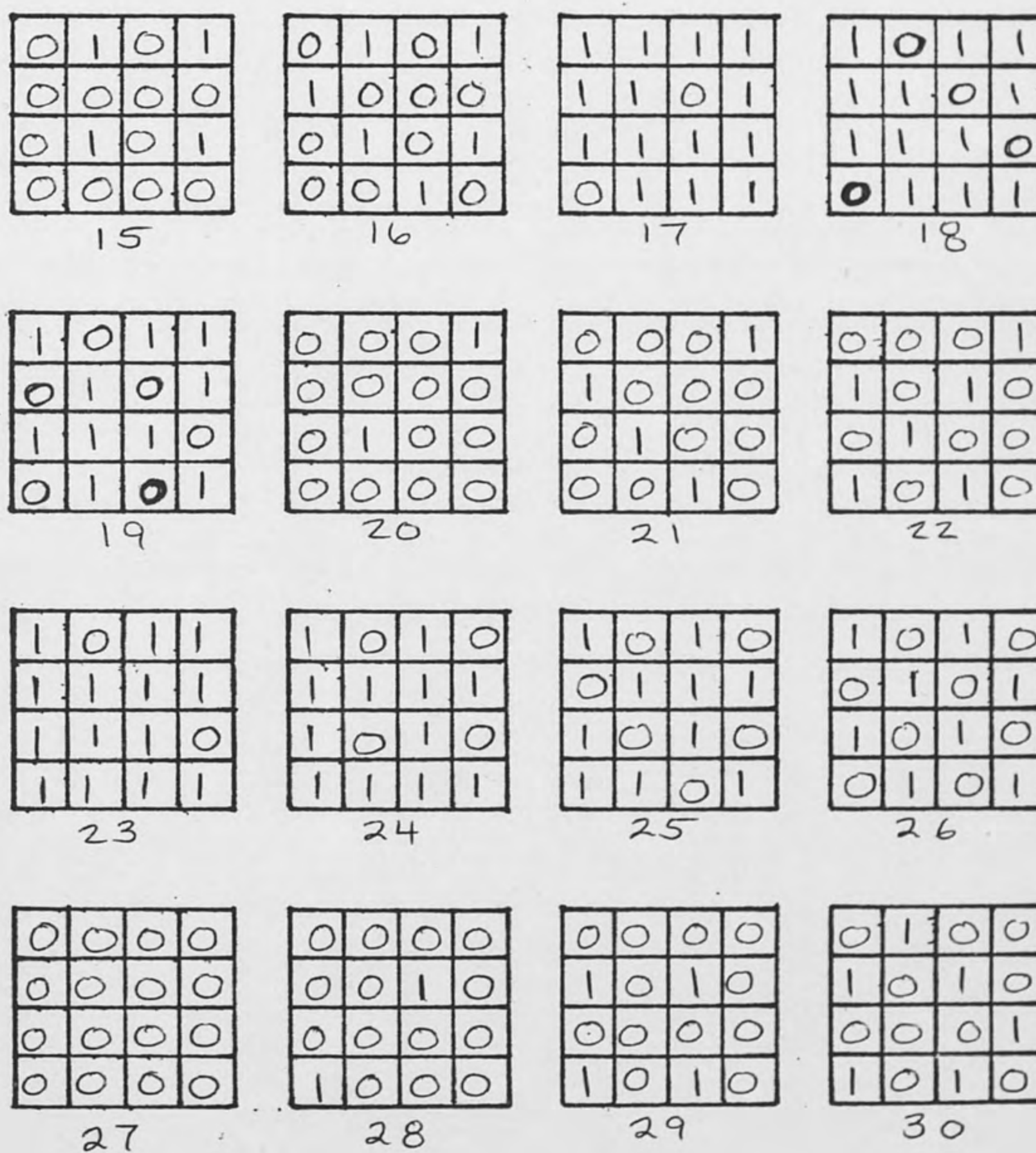


Fig. 9-Continued

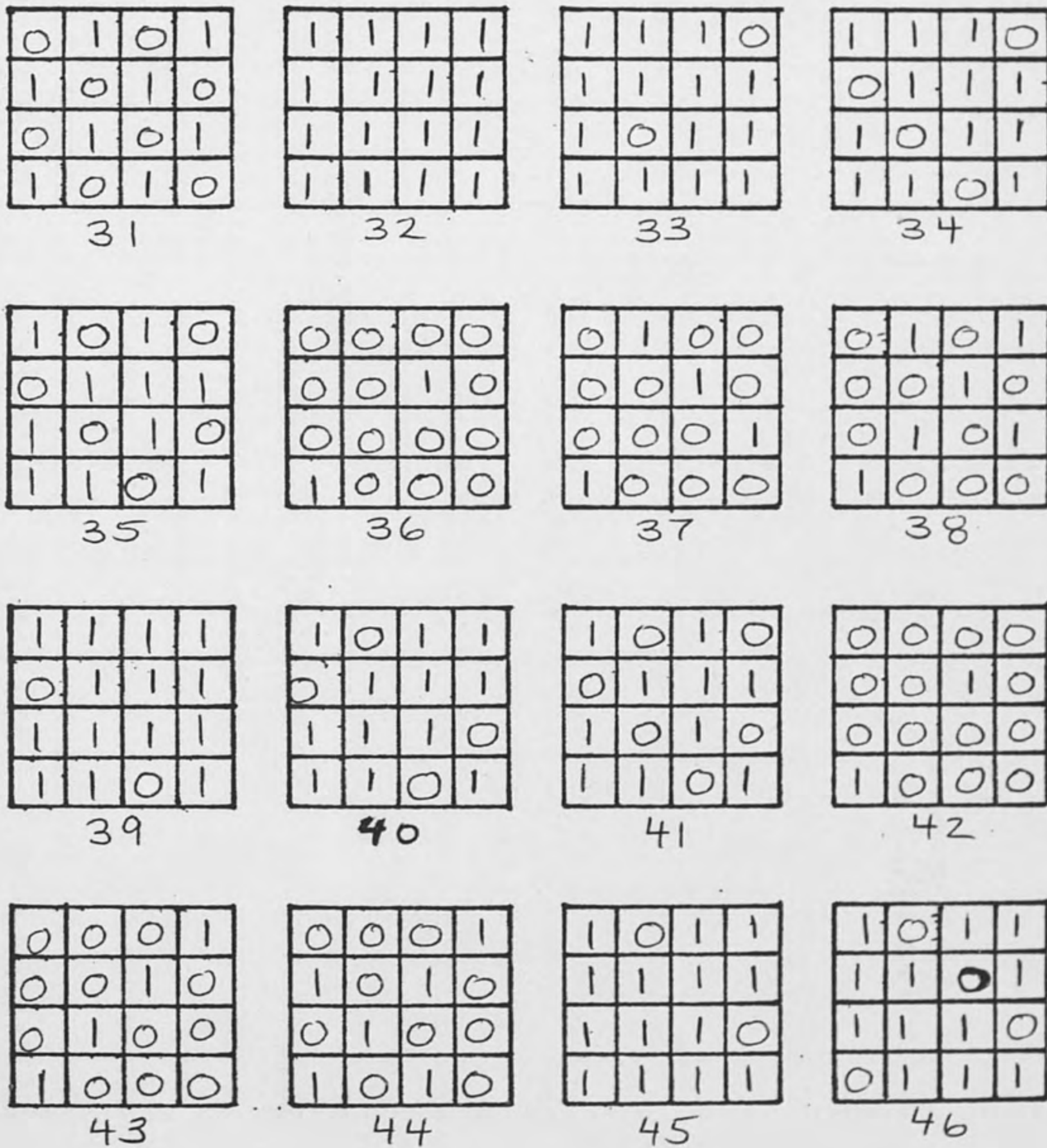


Fig. 9-Continued

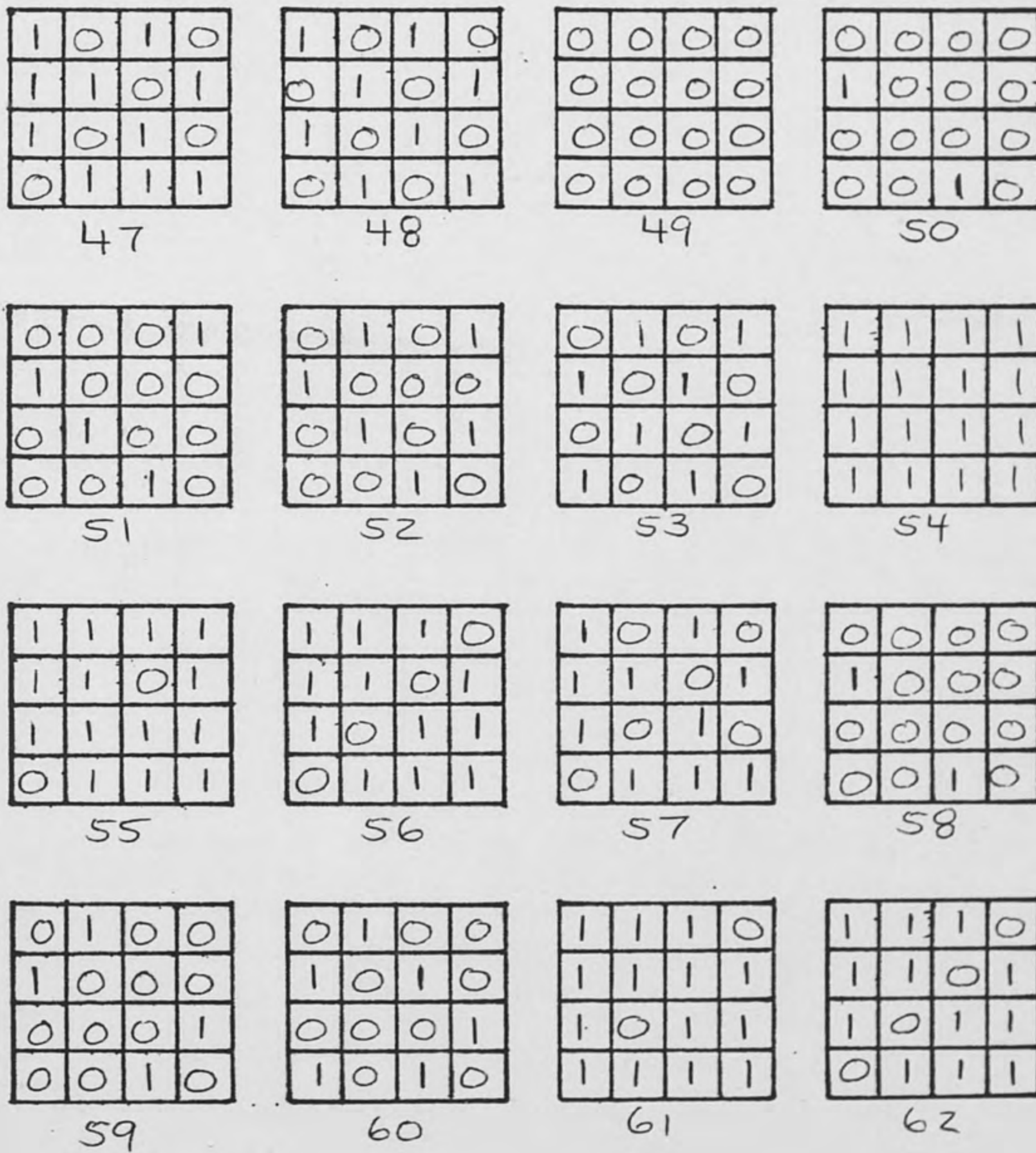


Fig. 9-Continued

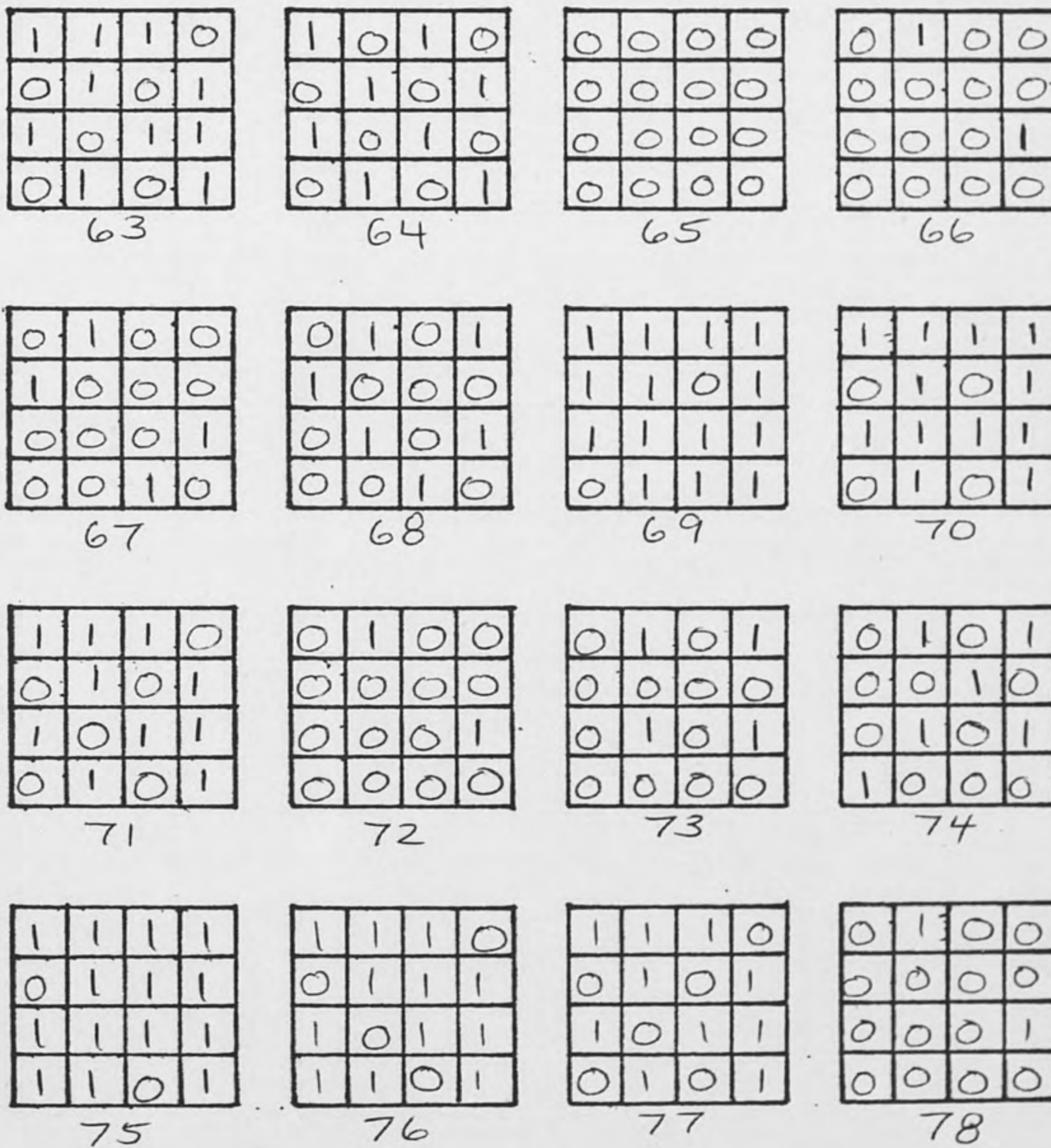


Fig. 9-Continued

0	1	0	0
0	0	1	0
0	0	0	1
1	0	0	0

79

0	1	0	0
1	0	1	0
0	0	0	1
1	0	1	0

80

1	1	1	0
1	1	1	1
1	0	1	1
1	1	1	1

81

1	0	1	0
1	1	1	1
1	0	1	0
1	1	1	1

82

1	0	1	0
1	1	0	1
1	0	1	0
0	1	1	1

83

0	0	0	0
1	0	0	0
0	0	0	0
0	0	1	0

84

0	0	0	0
1	0	1	0
0	0	0	0
1	0	1	0

85

0	0	0	1
1	0	1	0
0	1	0	0
1	0	1	0

86

0	1	0	1
1	0	1	0
0	1	0	1
1	0	1	0

87

1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

88

W	C	Z	C
C	Y	C	X
Z	C	W	C
C	X	C	Y

CODD

0	1	1	1
1	1	1	1
1	1	0	1
1	1	1	1

1

0	1	1	1
1	1	1	0
1	1	0	1
1	0	1	1

2

0	1	1	1
1	0	1	0
1	1	0	1
1	0	1	0

3

0	1	0	1
1	0	1	0
0	1	0	1
1	0	1	0

4

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

5

Fig. 9-Continued

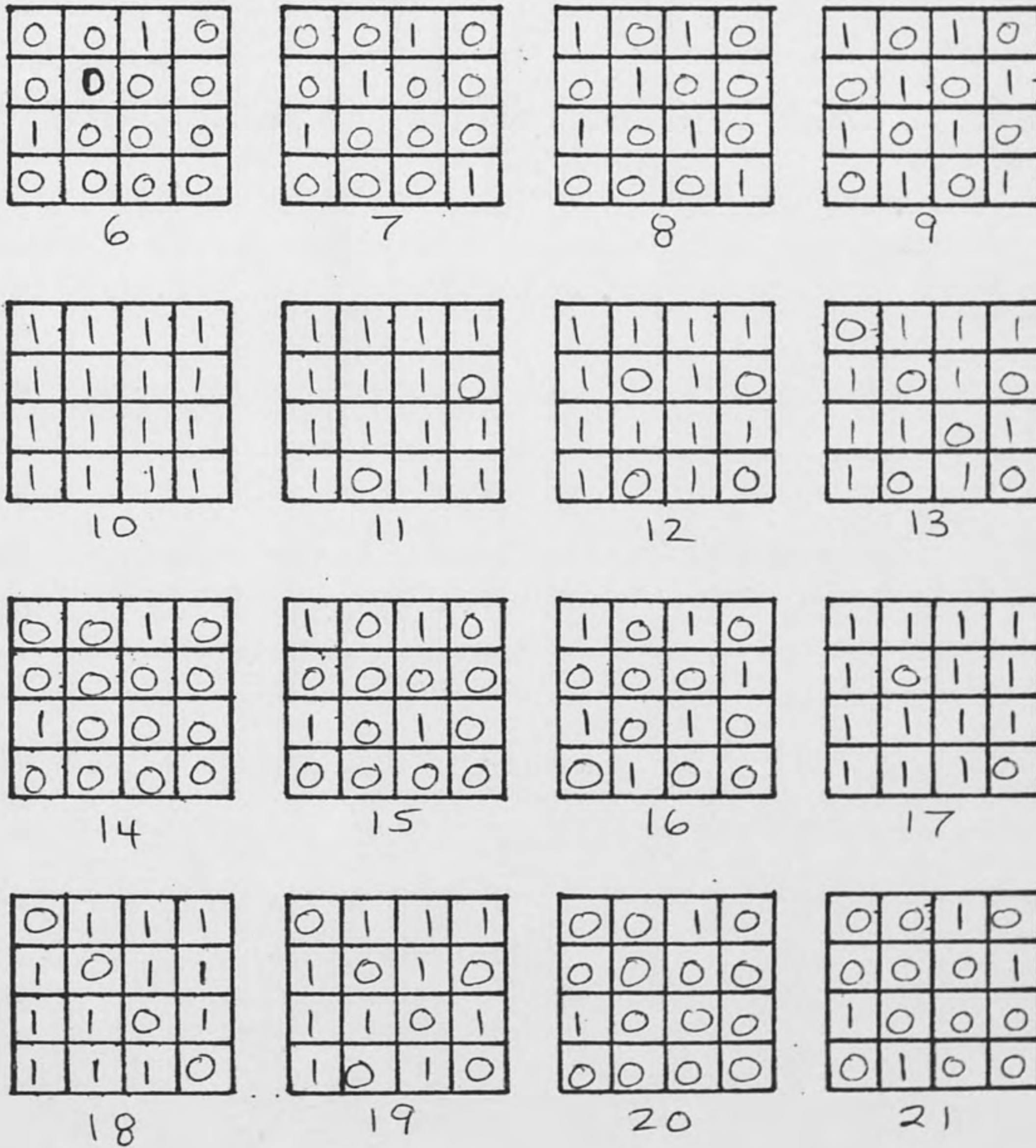


Fig. 9-Continued

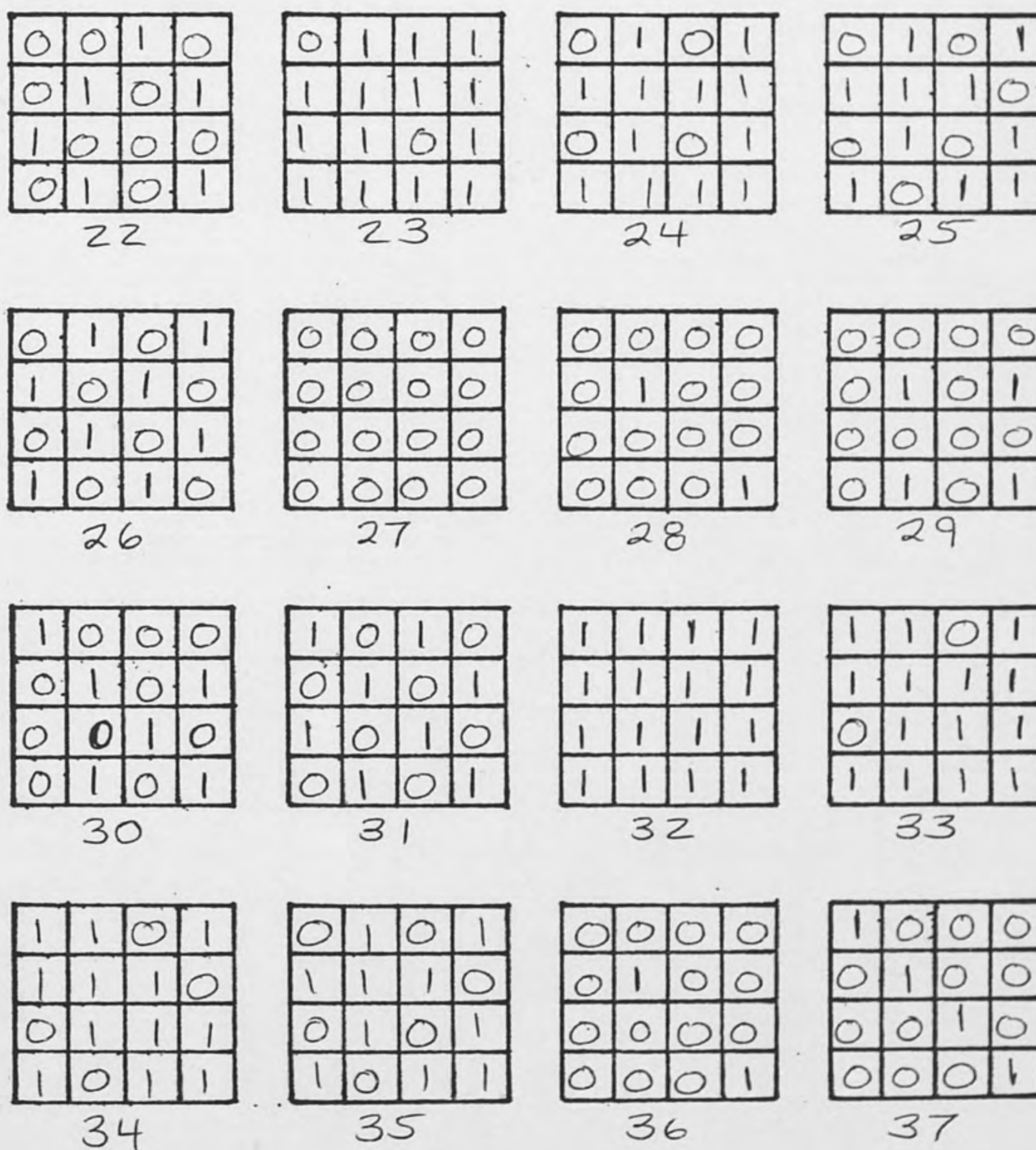


Fig. 9-Continued

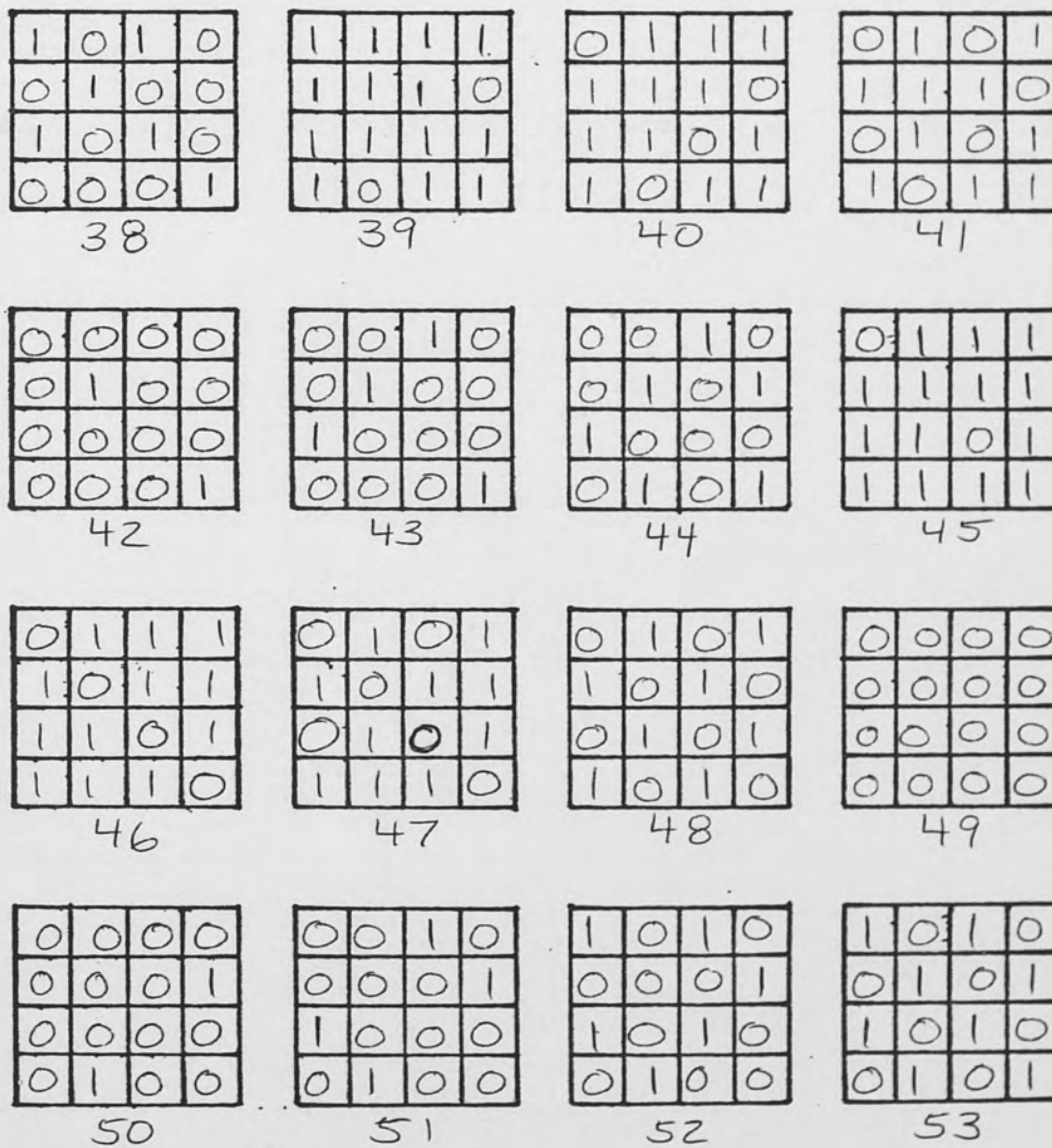
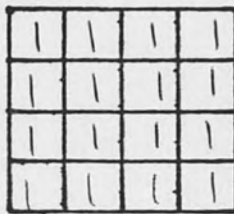
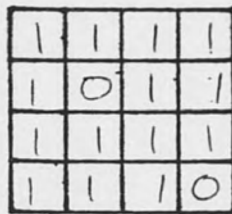
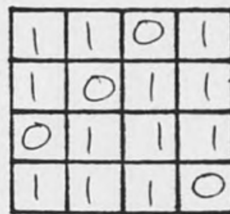
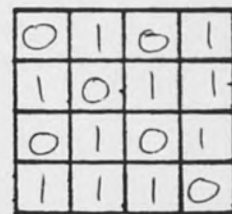


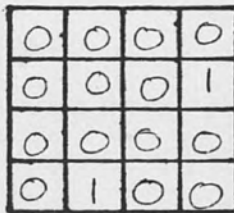
Fig. 9-Continued

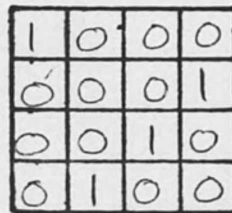

54

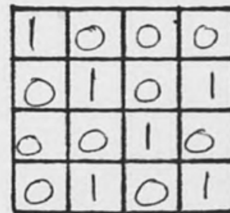

55

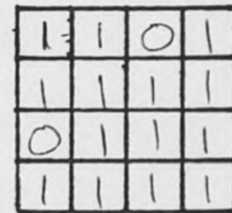

56

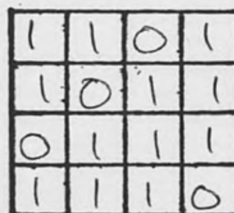

57

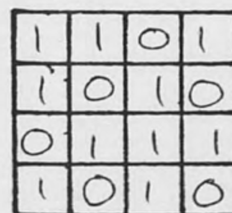

58

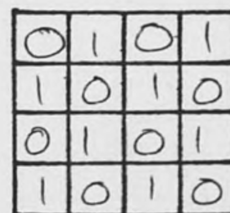

59

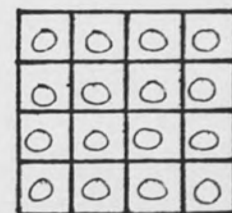

60

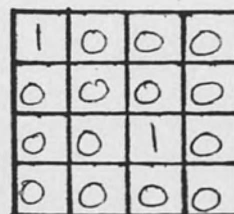

61

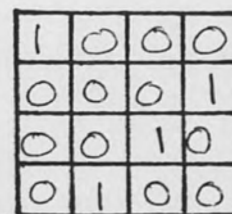

62

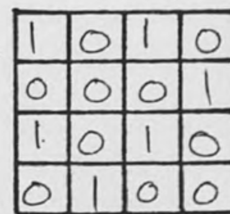

63


64


65


66


67


68

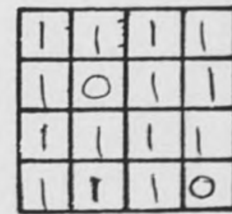
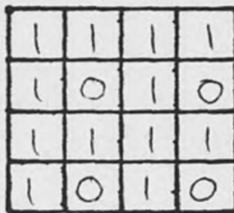
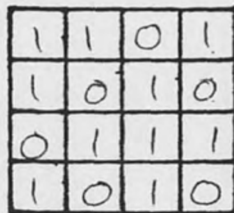
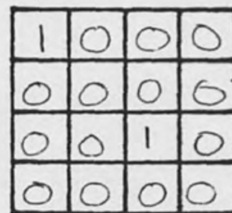
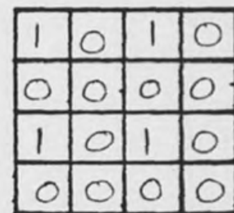

69

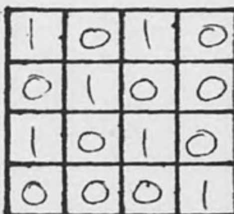
Fig. 9-Continued

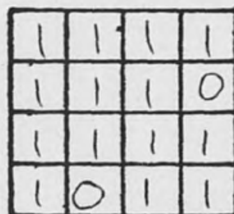

70

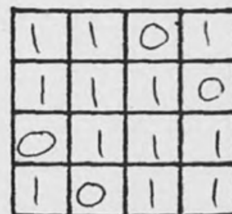

71

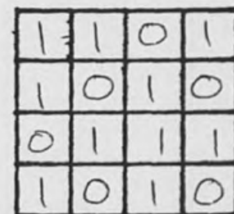

72

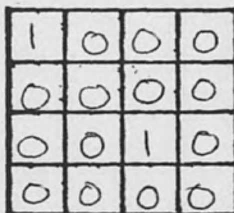

73

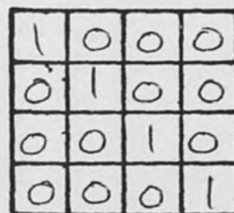

74

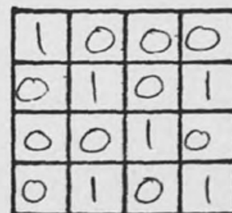

75

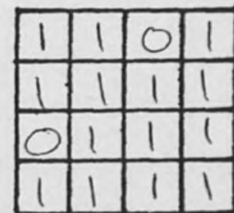

76

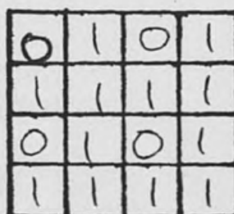

77

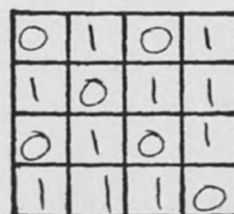

78

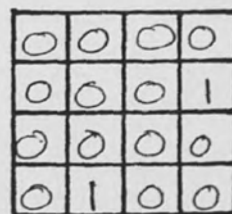

79


80


81


82


83


84

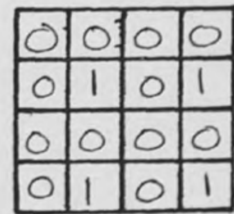

85

Fig. 9-Continued

0	0	1	0
0	1	0	1
1	0	0	0
0	1	0	1

86

1	0	1	0
0	1	0	1
1	0	1	0
0	1	0	1

87

1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

88

Fig. 9-Continued

CHAPTER 8

SUMMARY AND CONCLUSIONS

Fault testing algorithms for solid state RAMS generally fall into either of the following two categories: (1) when the test length is reasonably short, fault coverage is inadequate; (2) when a test algorithm has comprehensive coverage of faults, the test lengths become impractically long to be used for larger sized RAMS. In addition, most of the test algorithms were designed by using heuristic approaches rather than analytical.

The memory test algorithm presented in this paper is of a moderate test length without severely compromising the fault coverage. It differs from other algorithms primarily in the nature of the fault model and presumed realm of fault interactions. By adopting the restriction that all coupling faults are monotonic write faults occurring in a five-cell neighborhood, a comprehensive fault detection algorithm of minimal test length can be developed. This is an improvement over other PSF detection algorithms due to its reduced test length.

The DAPIF memory test algorithm should be used in conjunction with other tests which detect other types of failures and which verify the remainder of the RAM's internal circuitry. Because of the simplicity of algorithm DAPIF, it can be easily implemented for both chip and memory board applications. This is of particular interest since memory faults account for a disproportionate number of computer system failures.

CHAPTER 9

SUGGESTIONS FOR FUTURE RESEARCH

The principle factor in the fault coverage of PSF test algorithms is the restrictions placed on the neighborhood. In this paper, the neighborhood was restricted to five cells, a center cell and the four cells at its top, bottom, left and right. Even though this restriction can be justified to a certain extent, it is more desirable if the size of the neighborhood can be expanded. While it was shown that it is impractical to consider unrestricted PSFs, some neighborhoods which merit investigation include the entire row and column of the center cell or the four cells at the upper left, lower left, upper right and lower right corners of the center cell making a nine-cell neighborhood.

DEFINITION OF TERMS

Adjacent Neighborhood Pattern (ANP) - A pattern of ones and zeros and a transition in the neighborhood cells.

Adjacent Pattern Interference Fault (APIF) - See monotonic write fault.

Apparent State - The value resulting from a read of the memory cell.

Arc-complete - A trajectory which contains every arc of a digraph.

Arc-disjoint - A collection of cycles which have no arc in common.

Bipolar device - An electronic device whose operation depends on the transport of both holes and electrons (i.e., positive and negative charges).

Cell - A uniquely addressable storage location in the memory.

Cell Assignment - The use of symbols to map out the memory array to enable patterns applicable to the neighborhood model to be identified.

CEVEN - The subdivided memory array of deleted neighborhood cells whose center cells all lie on even (row plus column) addresses).

Closed Sequence - A node-arc sequence in which the initial and terminal nodes of the sequence are the same. This is also referred to as a cycle.

CODD - The subdivided memory array of deleted neighborhood cells whose center cells all lie on odd (row plus column) addresses.

Coupled Cell - When a transition in cell C_i causes the value of cell C_j to change, C_j is referred to as the coupled cell.

Coupling Cell - When a transition in cell C_i causes the value of cell C_j to change, C_i is referred to as the coupling cell.

Coupling Fault - A zero-to-one or one-to-zero transition in a memory cell which changes the state of another cell in the memory to that same value.

Cycle - See closed sequence.

DAPIF - Acronoym for an algorithm to Detect Adjacent Pattern Interference Faults.

D.C. Parameter Testing - Tests for measurable parameters such as power dissipation, noise margins and leakage currents.

Decomposed digraph - A digraph broken down into a collection of arc-disjoint subgraphs.

Deleted Neighborhood - The set of neighborhood cells excluding the center cell. In a five-cell neighborhood, it is the four cells to the top, bottom, left and right of the center cell.

Digraph - A directed graph which has the property of being an irreflexive relation (i.e., a network with no loops or parallel arcs).

Dynamic State - See transition.

Eulerian - A digraph is Eulerian if for every node, the indegree is equal to the outdegree (i.e., the number of arcs entering each node is equal to the number of arcs leaving it).

Eulerian cycle - A cycle whereby each arc is traversed exactly once and the starting and terminating node is the same.

Expected state - The expected contents of the memory cell following a write operation.

Fault detection - When a difference between the expected state and the apparent state of one of the memory cells under test occurs.

Fault location - The failed cell and the ANP which enabled the APIF to be detected.

Fault model - Representation of the manner in which the faults of interest are expected to occur.

Five-cell neighborhood - A memory cell surrounded by four cells at its top, bottom, left and right.

Hamming distance - The number of digits differing between two binary numbers.

IC - Integrated Circuit (Solid State) - A combination of interconnected circuit elements inseparably associated on or within a continuous substrate.

Indegree - The indegree of a node is the number of arcs entering the node.

Internal State - Actual stored contents of the memory cell.

Irreflexive - A relation is irreflexive if no node has an arc which is a loop back to itself.

Lemma - A subsidiary preposition assumed to be valid and used to demonstrate a principal proposition.

Marching Ones and Zeros - A basic memory test which gives the appearance of a parade of ones or zeros marching through the memory.

Memory Array - The portion of the RAM device containing the storage elements. It is made up of a planar arrangement of cells consisting of n rows and m columns.

Memory Operation - The performance of a RAM read or write access.

Memory under test (MUT) - The memory cell array to which the test algorithm is being applied.

Metal-Oxide-Semiconductor (MOS) transistor - An active semiconductor device in which a conducting channel is induced in the region between two electrodes by a voltage applied to an insulated electrode on the surface of the region. The transition region is a majority carrier conductor, available in either positive-hole or negative, free-electron, types.

Minimum Sequence - See Optimal Test Algorithm.

Monotonic Write Fault - A PSF such that an ANP involving a transition opposite to the state of the center cell will cause the state of the center cell to change.

Neighborhood - A memory cell and the surrounding cells making up the defined area in which restricted PSFs will be considered.

Nine-Cell Neighborhood - An extension of the five-cell neighborhood with the cells to the upper left, upper right, lower left and lower right of the center cell also considered in the neighborhood.

Node-arc sequence - An alternating sequence of nodes and arcs.

Operation - See Memory Operation.

Optimal Test Algorithm - The minimum sequence of memory operations required to detect all faults defined in the fault model.

Outdegree - The outdegree of a node is the number of arcs leaving the node.

Pattern Sensitive Fault (PSF) - Device anomolies such as dynamic timing parameters, current leakage and parastic capacitance which cause the RAM to be sensitive to transitions and/or patterns in memory.

Pattern Sensitivity Testing - Tests which detect the RAM's sensitivity to transitions and/or patterns in memory.

Random Access Memory (RAM) - A semiconductor storage device in which the access time is independent of the location of the data.

Relation - A network of nodes and arcs in which no two distinct arcs are parallel.

Semiconductor - A material with conductivity roughly midway between that of conductors and insulators and with which the conductivity increases with temperative over a certain temperature range.

State - See internal state, apparent state and expected state.

Static state - The value of the memory cell maintained across the duration of the memory cycle.

Stuck-at-Fault - A cell in the memory whose state remains independent of reads and writes to it.

Symmetric digraph - A digraph in which each pair of nodes is joined by two arcs in opposite directions.

Test Algorithm - A predetermined sequence of memory read and write operations designed to test the memory for faults defined in the fault model.

Trajectory - A node-arc sequence in which no arc occurs more than once.

Transition - A memory cell whose state is changing from one to zero or zero to one during a memory write operation.

Transition Fault - When a cell fails to undergo a zero-to-one and/or a one-to-zero change in state on a write cycle.

Traversable - A non-trivial digraph which has an arc-complete closed trajectory.

Walking Ones and Zeros - An extensive memory test which gives the appearance of a single "one" bit walking through a sea of zeros after which a "zero" bit is walked through a sea of ones.

DEFINITION OF NOTATION

C_i, C_j - Symbol for cells at locations i and j, respectively, in the memory where $0 \leq i, j \leq N-1$.

m - Number of columns in the memory cell array.

n - Number of rows in the memory cell array.

N - The number of cells in the memory array.

T, B, L, R, C - Symbols representing the top, bottom, left, right and center cells, respectively, in a five-cell neighborhood.

V_i - Symbol for node V_i.

V_iV_j - Symbol for an arc from node V_i to node V_j.

W, X, Y, Z - Symbols representing the four cell assignments of cells in CEVEN AND CODD.

$\uparrow$, $\downarrow$ - Symbols for zero-to-one transition and one-to-zero transition, respectively.

REFERENCES CITED

Anderson, K. "Device Manufacturers Test Problems." In _Digest of Papers 1972 Semiconductor Test Symposium, Cherry Hill, NJ, October 4, 1972_, pp. 17-26. New York: Institute of Electrical and Electronic Engineers, 1972.

Batdorf, H. A.; Hensler, D. H.; and Wasson, R. D. "Reliability Evaluation Program and Results for a 4K Dynamic RAM." In _16th Annual Reliability Physics Symposium_, _San Diego, CA, April 18-20, 1978_, pp. 14-18. New York: Institute of Electrical and Electronic Engineers, 1978.

Clary, J. B., and Sacane, R. A. "Self-Testing Computers." _Computer_ 12 (October 1979): 49-59.

Cocking, J. "RAM Test Patterns and Test Strategy." In _Digest of Papers 1975 Semiconductor Test Symposium_, _Cherry Hill, NJ, October 14-16, 1975_, pp. 1-8. New York: Institute of Electrical and Electronic Engineers, 1975.

Colbourne, E. D.; Coverley, G. P.; and Behera, S. K. "Reliability of MOS LSI Circuits." _Proceedings of the IEEE_ 62 (February 1974): 244-259.

Fee, W. G. "Memory Testing." In _Digest of Papers 1977 Spring COMPCON, San Francisco, CA, February 1977_, pp. 51-58. New York: Institute of Electrical and Electronic Engineers, 1977.

Fischer, J. E. "Test Problems and Solutions for 4K RAMS." In _Digest of Papers 1974 Semiconductor Test Symposium, Cherry Hill, NJ, November 5-7, 1974_, pp. 53-71. New York: Institute of Electrical and Electronic Engineers, 1974.

Green, C. W. "PMOS Dynamic RAM Reliability: A case Study." In _17th Annual Reliability Physics Symposium, San Francisco, CA, April 24-26, 1979_, pp. 213-219. New York: Institute of Electrical and Electronic Engineers, 1979.

Harary, P.; Norman, R. Z.; and Carwright, D. Structural Models: An Introduction to the Theory of Directed Graphs. New York: John Wiley & Sons, 1965.

Hayes, J. P. "Detection of Pattern-Sensitive Faults in Random Access Memories." IEEE Transactions on Computers C-24 (February 1975): 150-157.

Hayes, J. P. "Testing Memories for Single-Cell Pattern-Sensitive Faults." IEEE Transactions on Computers C-29 (March 1980): 249-254.

Hnatek, E. R. "4-Kilobit Memories Present a Challenge to Testing." Computer Design 14 (May 1975): 117-125.

Hnatek, E. R. "Semiconductor Memory Attrition Summary." In Digest of Papers 1976 Semiconductor Test Symposium, Cherry Hill, N J, October 19-21, 1976, pp. 35-40. New York: Institute of Electrical and Electronic Engineers, 1976.

Huston, R. E. "Testing Semiconductor Memories." In Digest of Papers 1973 Semconductor Test Symposium, Cherry Hill, N J, October 2-3, 1973, pp. 27-55. New York: Institute of Electrical and Electronic Engineers, 1973.

Nair, R; Thatte, S. M.; and Abraham, A. "Efficient Algorithms for Testing Semiconductor Random Access Memories." IEEE Transactions on Computers C-27 (June 1978): 572-576.

Nickel, V. V. "VLSI - The Inadequacy of the Stuck-at-Fault Model." In Digest of Papers from 1980 Test Conference, Philadelphia, PA, November 11-13, 1980, pp. 378-381. New York: Institute of Electrical and Electronic Engineers, 1980.

Patch, F. D. and Zumbach, R. G. "Evaluation of Array Tests." In Digest of Papers from 1980 Test Conference, Philadelphia, PA, November 11-13, 1980, pp. 131-136. New York: Institute of Electrical and Electronic Engineers, 1980.

Rosenfield, P. "Memory Testing - Characterization Timing and Patterns." Electronics and Power 25 (January 1979): 26-31.

Shah, A. A. "Design and Application of Semiconductor Memory Test Hardware." _GTE Automatic Electric Journal_ 15 (April 1976): 65-79.

Srini, V. P., "Fault Location in a Semiconductor Random-Access Memory Unit." Ph.D. dissertation Virginia Polytechnic Institute and State University, 1976.

Srini, V. P. "API Tests for RAM Chips." _Computer_ 10 (July 1977): 32-35.

Suk, D. S. "Functional and Pattern Sensitive Testing Algorithms for Semiconductor Random Access Memories." Ph.D. dissertation, University of Iowa, 1978.

Thatte, S.M. "Fault Diagnosis of Semiconductor Random Access Memories." Ph.D. dissertation, University of Illinois, 1977.