DIAGNOSIS OF INTERMITTENT FAULTS IN DIGITAL SYSTEMS

Thesis for the Degree of Ph. D.
MICHIGAN STATE UNIVERSITY
SAMIR KAMAL
1972

ABSTRACT

DIAGNOSIS OF INTERMITTENT FAULTS IN DIGITAL SYSTEMS

By

SAMIR KAMAL

Digital computers are being relied upon as integral parts of an increasing number of systems handling all aspects of our life. The proper operation of computers is vital to the functioning of these computerized systems. One of the major approaches to achieve proper operation of computers is fault diagnosis plus repair. This thesis lends itself to one aspect of this approach, namely: the diagnosis of intermittent faults in combinational circuits.

Intermittent faults in digital systems are those faults whose effects are not observed all the time. A system having an intermittent fault may show the effect of such a fault when an input test is applied one time, and possibly not show it when the same test is applied many other times.

A probabilistic model is suggested for intermittent faults in logical circuits. The model assumes that the circuit is irredundant and that it can have only one of a

possible set of faults (the single fault assumption). It also assumes that the faults are well behaved and are signal independent. Testing for such faults is done through the repeated application of tests that would detect these faults had their effect been permanent. These tests are generated using any of the methods employed for generating tests that detect permanent faults. The prior probability of the circuit having any of the intermittent faults is assumed to be known in the model. Also the probability of observing the effect of each fault, if that fault exists, is assumed.

A procedure for the detection of intermittent faults is suggested. It is analogous to a sequential statistical decision problem. At any stage during testing, the procedure terminates if a failure is observed or if the decision rules decides that the circuit is fault free. Otherwise, it selects an appropriate test to be applied at the next stage. The decision rule used in the detection procedure is selected such that it insures the procedure termination in a finite number of steps. Least upper bounds on the number of repetitions of tests that detect a particular fault are derived. They are employed in designing optimum detection experiments. Such an optimization problem is found to be equivalent to an integer programming problem.

A diagnosis procedure, also employing the repetition of tests that detect permanent faults, is proposed. It is proved

that the conditions required in the procedure guarantee that the expected length of the diagnosis experiment is finite. Test properties that are needed for the diagnosis of intermittent faults are determined. Some fundamental differences between those properties needed for the intermittent fault case and those needed for the permanent fault case are pointed out. The problem of optimizing the diagnosis experiment is examined. Unfortunately, the true optimum solution can be obtained only through impossibly lengthy enumeration. Two suboptimal approaches that are heuristic in nature are suggested.

DIAGNOSIS OF INTERMITTENT FAULTS IN DIGITAL SYSTEMS

By

SAMIR KAMAL

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science

1972

TO

MY PARENTS

ACKNOWLEDGMENTS

I am grateful to Dr. Carl V. Page, the chairman of my guidance committee, for his guidance and encouragement during the course of this thesis. My thanks also go to Dr. Richard C. Dubes, Dr. Edgar M. Palmer, Dr. Morteza A. Rahimi, and to Dr. Bernhard L. Weinberg for serving on my guidance committee and for reviewing this work.

I wish to express my thanks to the Division of Engineering Research for their financial support during the writing of this thesis, and further, to the Egyptian Government for the financial support that made my graduate studies possible.

Finally, I am deeply indebted to my wife Lillian for her encouragement, loyalty, patience and understanding; and also to my daughter Heba whose joyous smile is always a source of happiness and reassurance.

TABLE OF CONTENTS

Table of Contents

Table of Contents

LIST OF TABLES

LIST OF FIGURES

List of Figures

GLOSSARY

Term	Meaning	Page
a_{ij}	An entry in the fault table matrix	40
A	Fault table matrix	40
A_0	Initial uncertainty	59
$A_{1,0}$	Uncertainty after the application of a single test that did not fail	59
$A_{1,1}$	Uncertainty after the application of a single test that failed	59
A^*	Reduced fault table matrix	94
b	k-dimensional random variable	69
b_j	Number of blocks, or j-th bias	52 94
D	A signal that is normally 1 but becomes 0 when a fault is present	31
$\bar{D}$	A signal that is normally 0 but becomes 1 when a fault is present	31
e_i	Conditional probability of malfunction	70
E(.)	Expectation	114

Glossary

f_0	The fault free condition	40
f_i	Permanent fault number i	19
$\vec{f}_i$	The row corresponding to fault f_i in the fault table	41
$f_i(x_1,x_2,...)$	A Boolean function of $x_1,x_2,...$	7
$f_i'(x_1,x_2,...)$	A Boolean function of $x_1,x_2,...$	10
F	Permanent fault space	64
F_p	Set of permanent faults corresponding to Ω_p	101
F_{p_i}	Subset of F_p that is detected by i-th test of τ_p	123
$g_k(x_1,x_2,...)$	A Boolean function of $x_1,x_2,...$	10
i_0,i_1,i_n	Fault parameters	38
k	Least upper bound on length of the detection experiment for a simple case, or	81
	Number of repetitions of τ_p	114
k_i	Least upper bound on number of tests detecting f_i that are needed for the detection experiment, or	91
	Number of times f_i is covered by τ_p	113
$\bar{k}$	Expected value of k	114
$k\to\infty$	As k approaches infinity	78

Glossary

ℓ	Number of tests detecting a given fault, or	89
	Experiment length	92
$\bar{\ell}$	Expected length of experiment	56
ℓ_i	Experiment length if i-th fault is present	56
l_i	Number of 1's in column i of the fault table	52
$l_{i,k}$	Number of 1's in portion of column i that belongs to block k	53
m	Number of output variables, or	7
	Number of tests in τ	40
$M_j(.)$	The j-th membership function	84
n	Number of input variables, or	7
	Number of faults	40
p	Number of secondary variables in a sequential circuit	9
p_i	Prior probability of fault number i	56
$p_{0,k}$	Posterior probability of ω_0 of a simple case after applying k tests, none of them failed	77
$p_{1,k}$	Posterior probability of ω_1 of a simple case after applying k tests, none of them failed	76

Glossary

P(.)	Probability of	67
q_i	Probability of failure of test t_i, or	59
	Number of 1-entries in i-th row of the fault table matrix	97
r	Resolution figure of merit	123
s	Threshold for posterior probability of a simple case	80
δ_0	Sample point denoting all components being fault free	65
δ_i	Sample point denoting that component pertaining to i-th fault is faulty	65
s-a-0	Stuck at logical value 0	12
s-a-1	Stuck at logical value 1	12
S	Sample space	65
t_j	Test number j	19
T_j	Random variable corresponding to test t_j	66
u	Threshold for likelihood ratio of a simpel case	82
u_i	Threshold for the i-th likelihood ratio	90
W	Figure of merit	123
W_i	Weight of test t_i	52

Glossary

$W_{i,j}$	Weight of test t_i after application of j tests	53
x	A don't care value	32
x_j	The j-th input of a switching circuits, or	7
	Number of repetitions of test t_j	92
$\bar{x}_i$	Complement of the switching variable x_i	8
y	Output of a gate	12
y_i	Current value of i-th secondary variable of a sequential circuit	9
y*	Output function with fault parameters substituted	38
Y_i	Next value of i-th secondary variable of a sequential circuit	9
z	Output function	28
z_i	The i-th output of a switching circuit	7
z'	Output function	37
ΔA_i	Information gain due to the application of test t_i	59
ϵ	Member of (set membership)	69
η	Number of elements in Ω_p or in F_p	101
η_i	Number of elements in F_{p_i}	123

Glossary

θ_i	Number of 0's in column i of the fault table	52
$\theta_{i,k}$	Number of 0's in portion of column i that belongs to block k	53
λ_k	Likelihood ratio of a simple case after applying k tests, none of them failed	79
$\lambda_{i,k}$	Likelihood ratio for ω_i after applying k tests, all detecting f_i and none of them failed	88
μ	Number of elements in τ_p	95
$\sum$	Summation	53
τ	Test set	65
τ_i	Subset of τ containing all tests detecting f_i	66
τ_p	Set of tests that cover F_p	102
T	Random variable set	66
T_i	Random variable subset that corresponds to τ_i	67
$T_p = \overrightarrow{0^{\mu}}$	$T_1=0, T_2=0,\ldots, T_\mu=0$ where $T_p = \{T_1,T_2,\ldots,T_\mu\}$	112
$T_p^k = \overrightarrow{0^{\mu k}}$	$T_i=0, T_i=0,\ldots$ k times for all i $(1 \leq i \leq \mu)$ where $T_p = \{T_1,T_2,\ldots,T_\mu\}$	113

Glossary

ω_0	State of being fault free	62
ω_i	State of having i-th intermittent fault	63
Ω	State space	62
Ω_p	Set of possible intermittent faults	101
$\Omega \times S$	Action space (cartesian product of Ω and S)	67
$\overrightarrow{0^k}$	k-dimensional binary vector all of its entries are 0's	69
$(\Omega \times S)^k$	Cartesian product of $(\Omega \times S)$ by itself k times	113
{a,b,...}	A set whose elements are a,b,...	20
$\frac{dz}{dx_i}$	Boolean difference of z with respect to x_i	39
$\rightarrow$	Function mapping from the set on the left hand side of the arrow to the right hand side	66
$\lceil x \rceil$	Smallest integer greater than or equal to x	121

CHAPTER I

INTRODUCTION

The last quarter of a century has witnessed major changes in many systems organizations which affect every facet of life. The reliance on digital computers as intergral parts of these systems has introduced irreversible changes. The speed and efficiency of computers in handling massive amounts of information are the main motives for these changes. For these same reasons, huge and complex systems are now designed that were unthinkable before the use of computers, e.g., systems for space flights. These so-called computerized systems span a very wide range of applications, from the simple to the extremely complex, from government agencies to private enterprise, from space missions to patient monitoring. Air line reservations, production line scheduling, highway traffic control, ABM, are just a few examples of these systems.

Prior to computerization, systems had manual backups for use in case of failures. With the current fast and sophisticated systems, manual backup is infeasible except for a limited number of cases for a short period of time.

Proper operation of computers, being important parts of these systems, must be assured, particularly for real-time applications, where down time for an extended period of time would be very costly, if not disastrous. An auto assembly plant will be shut down and all the labor force sent home for the day, if the production line is down for half an hour. With this type of strigent requirements, it is imperative that we use more reliable computers, even though today's machines are built from components far more reliable than their counterparts that were used a decade or so ago.

1.1 FAULT-TOLERANT COMPUTING

Fault-tolerant computing is a field of study whose major concern is the assurance of proper operation of digital computers. A definition of the term was given by Ramamoorthy [30] as: "Fault-tolerant computing can be defined as the ability to execute specific algorithms correctly regardless of hardware failures or software errors." This definition, even though comprehensive, neglects the importance of time. It is very critical, especially with real-time applications, that algorithms be executed correctly within a tolerable period of time. Clearly, the goal implied in the above definition is far reaching and it is quite a challenge to achieve, even partially, this goal. The difficulty forseen in achieving

this goal should not be a discouraging factor in the work toward that end, at least we should try to reduce the penalty we are likely to pay, in case of failures or errors, to a minimum.

Even though the roots of this field started with the early days of computers [27, 28, 38], there has been considerable recent interest in it. Related research papers appear regularly in the literature. In addition, two recent international symposia* were solely devoted to this subject. Fault-tolerant computing encompasses theory and techniques for fault and error detection and correction, modeling, simulation, analysis, synthesis and architecture of fault-tolerant systems.

Structural reliability is of major importance to fault-tolerant systems. Several approaches are used to ensure structural reliability, some of which are presented in the following section. It should be emphasized that the term "reliable" is used here in the generic sense and not in the formal probabilistic sense as in the theory of reliability in engineering parlance. The term "credible" was used in lieu of that by Carroll [5].

---

*First International Symposium on Fault-Tolerant Computing, March 1971, and Second International Symposium on Fault-Tolerant Computing, June 1972.

1.2 SOME APPROACHES TO ATTAIN STRUCTURELY RELIABLE SYSTEMS

Several traditional approaches to the problem of assuring proper operation of digital systems have been studied. The most significant are:

(a) Use of Better Components and Better Designs.

An integrated circuit, for example, could be made more reliable if it were designed to be more noise discriminant, and to be capable of withstanding greater power supply fluctuations without suffering damage or malfunctioning than is presently possible. This approach appears to be an obvious and straightforward one, but it is limited by the available technology and by the economics governing the design. Often, the application still requires more reliability than this approach can provide.

(b) Redundancy.

Through this approach, it is possible, using additional hardware, to build a system that will function properly even after the failure of one or more of its components. The redundancy employed in space flights is a typical example of this approach. A good portion of the early work in fault-tolerant computing concentrated on the study of different redundancy systems. By guadrupling the number of contacts in a relay switching system, Moore and

Shannon [28] have shown that it is possible to come up with a system more reliable than the individual relays. This work was later extended to gate-networks by Tryon [37]. Other classic work in this area was due to Von Neumann who introduced the notion of the "Restoring Organ" [38]. This concept was later developed by Lyons [22] in the study of TMR (Triple Modular Redundancy). Redundancy is quite expensive and just postpones the inevitable: given sufficient time, enough failures will occur and the system will eventually malfunction. It is quite suitable for short-term applications such as space missions where correct operation must be guaranteed for a relatively short period of time and repair is rather difficult or even impossible.

(c) Fault Diagnosis Plus Repair.

Economic considerations make this approach the most favored one. With this approach, a system is tested to determine whether or not it is functioning as intended (fault detection), and if not, which part caused the trouble (fault diagnosis). Such testing is needed through the life of the system. When it is first installed, an acceptance test is needed. Thereafter, routine testing, e.g., performed by the CE (Customer Engineer), is performed as part of a maintainance program. Studies about automatic

testing for the so-called "self-repairing" systems were carried by Avizienis [1, 2] and others.

This approach has received a great deal of attention in recent years. It can be safely stated that, so far, it has been the backbone of fault-tolerant computing.

A combination of approaches (b) and (c), even though more expensive than the third approach, will lead to ultimate reliability. Such a combination is very suitable when the system is used for vital applications where uninterrupted operation is a must. Combining these two approaches must be done with care, since redundancy is incompatible with diagnosis because it will tend to mask the effect of a fault when it occurs so it will go undetected. However, clever use of redundancy (e.g., via additional outputs) could amount to greater reliability, if the redundant components are used in such a way that they are not redundant for test purposes. A system that made use of this combined approach is the ESS (Electronic Switching System) of the Bell System, where the failure time is limited to 2 hours in 40 years (acknowledged off the record by the designers).

1.3 SWITCHING CIRCUITS

Most of the work done in the area of fault diagnosis deals with faults in switching circuits. The

basic elements of switching circuits are the logical elements. Schematic notations for some of these elements are shown in Figure 1.1. Switching circuits are usually divided into two types: combinational and sequential circuits.

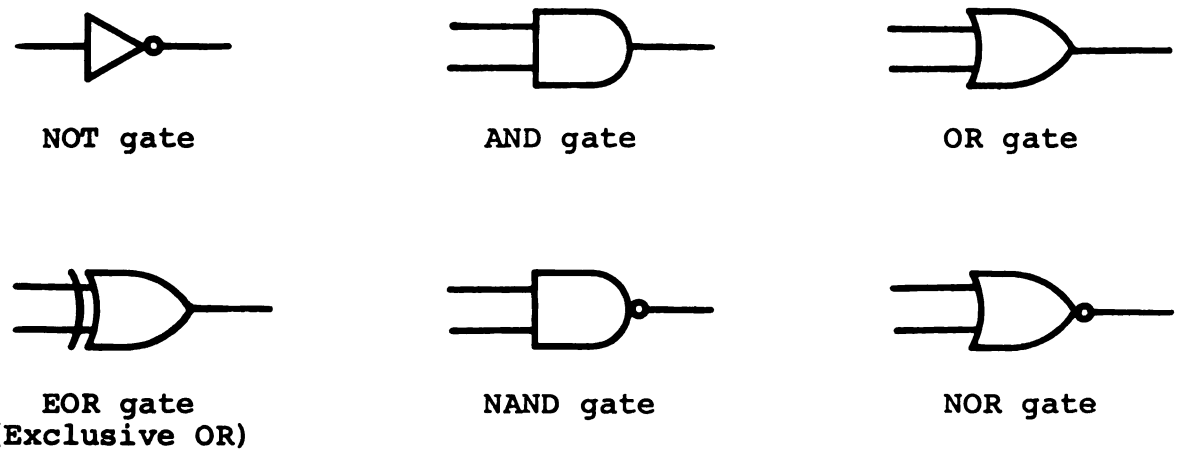


Figure 1.1 Some Logical Elements.

1.3.1 Combinational Circuits

A switching circuit is combinational [24] if its outputs z_i $(1 \leq i \leq m)$ can be written as Boolean functions of its inputs x_j $(1 \leq j \leq n)$

$$z_i = f_i\ (x_1, x_2, \ldots, x_n)\ , \qquad (1.1)$$

i.e., each output is a function only of the present values of the inputs. A block diagram for a combinational switching circuit is shown in Figure 1.2. The realization

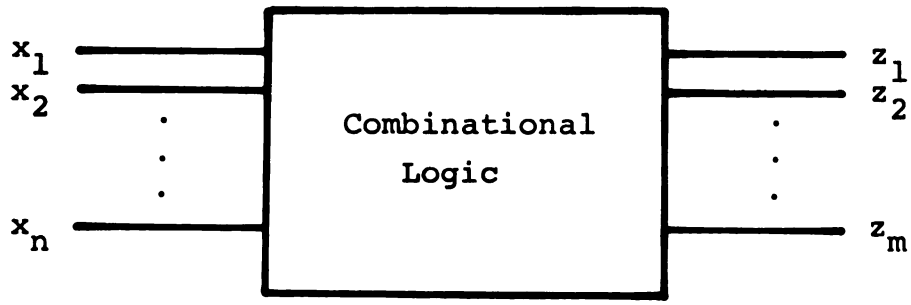


Figure 1.2. Block Diagram of a Combinational Circuit.

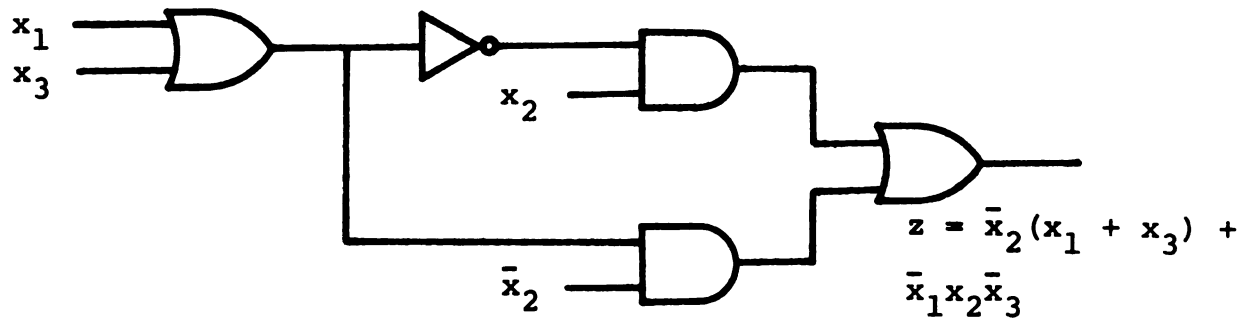


Figure 1.3. Example of a Combinational Circuit.

of such a circuit is characterized by the absence of feedback. An example of a combinational circuit is given in Figure 1.3.

1.3.2 Sequential Circuits

In a sequential circuit [24], the output at any time is a function of the current inputs and also of previous inputs. The history of previous inputs is summarized in the state of the circuit. The realization of a sequential circuit is characterized by feedback. The combined value of the feedback lines y_k $(1 \leq k \leq p)$, represent the current state of the circuit. A typical block diagram is shown in Figure 1.4.

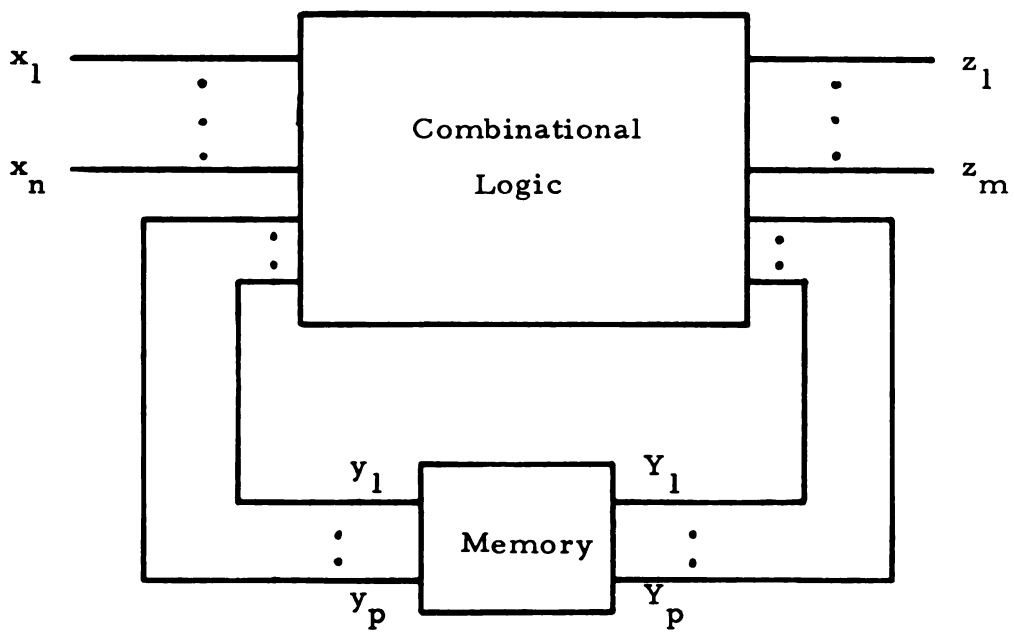


Figure 1.4. Block Diagram of a Sequential Circuit.

Sequential circuits are classified as "synchronous" or "asynchronous" depending on whether or not it is

operating under the control of clock pulses (additional input to the "memory" block of Figure 1.4 that is not shown). Asychronous sequential circuits are characterized by level inputs and outputs; while the inputs and outputs of synchronous sequential circuits could be either level or pulses.

Two different models for synchronous sequential circuits have been in use, one is due to Mealy [25] and the other is due to Moore [27]. In both models, the next state is a function of current input and current state, i.e.,

$$Y_k = g_k\ (x_1, x_2, \ldots, x_n; y_1, y_2, \ldots, y_p) \qquad (1.2)$$

The difference between the two models is in the output function. In the Mealy model it is a function of the current state and current input, i.e.,

$$z_i = f_i\ (x_1, x_2, \ldots, x_n; y_1, y_2, \ldots, y_p) \qquad (1.3)$$

while in the Moore model, the output is a function only of the current state, i.e.,

$$z_i = f_i'\ (x_1, x_2, \ldots, x_n) \qquad (1.4)$$

The two models can be shown to be quivalent, see Gill [18].

1.4 LOGICAL FAULTS

A fault is a physical defect that causes the circuit to malfunction. There are numerous factors that could give rise to faults, among them:

(1) Aging and gradual deterioration with time. This usually would result in what is called marginal faults, e.g., the gap between the ON and OFF thresholds of a transistor gets smaller.

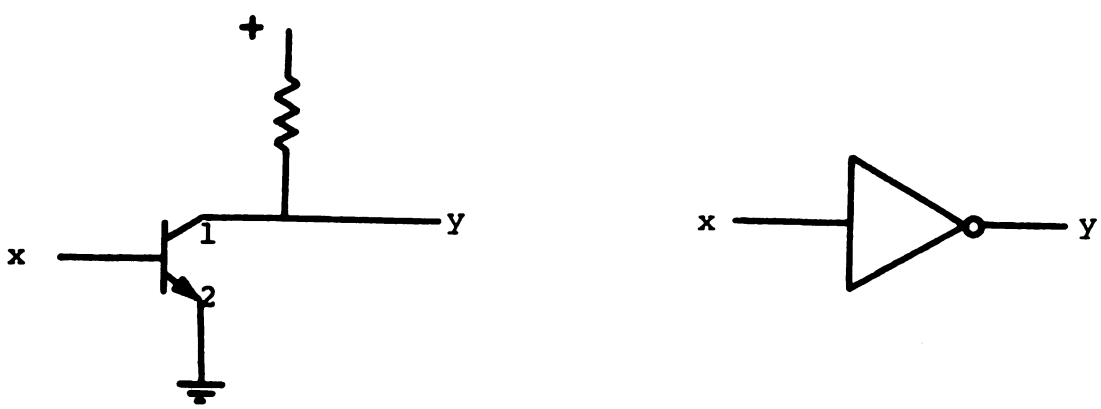
(2) Critical timing, noise, close design tolerances and loose wires. These would cause some sort of intermittent faults. Some intermittent faults eventually will become permanent faults.

(3) Solid failures such as a permanently open collector or base lead of a transistor, a broken wire, or a short circuit between adjacent connections. These will result in what is called permanent faults.

This thesis deals only with logical faults. These are the faults which affect changes in the logical behavior of the circuit. Failures that cause, say, changes in pulse shapes, but do not alter the logical functions realized by the circuit will not be considered. Also, power supply and clock failures are not considered here. Hereafter, the term fault would mean logical fault, unless otherwise indicated. The following are examples of faults and how they are logically described.

Example 1.1 (Faults in an inverter)

Consider the inverter circuit shown in Figure 1.5. An open collector failure (lead 1 open) will cause the output y to be at a high voltage regardless of the value of the input signal x. Logically, this can be represented as line y being s-a-1 (stuck at logical value 1) if positive logic is assumed. On the other hand, if the failure is of the form of a short between the collector and the emitter (short between 1 and 2), the output y will always be at ground voltage regardless of the input signal x. This is represented logically as line y being s-a-0 (stock at logical value 0).



1 open (y high) — y s-a-1

1 and 2 shorted (y low) — y s-a-0

Figure 1.5. Faults in an Inverter Circuit.

Example 1.2 (Faults in and AND gate)

The circuit diagram of a DRL (Diode Resistor Logic) AND gate is shown in Figure 1.6. If the diode at input x_1 is open, the circuit will behave as if the x_1 input is not present. This can be described logically as x_1 being s-a-1.

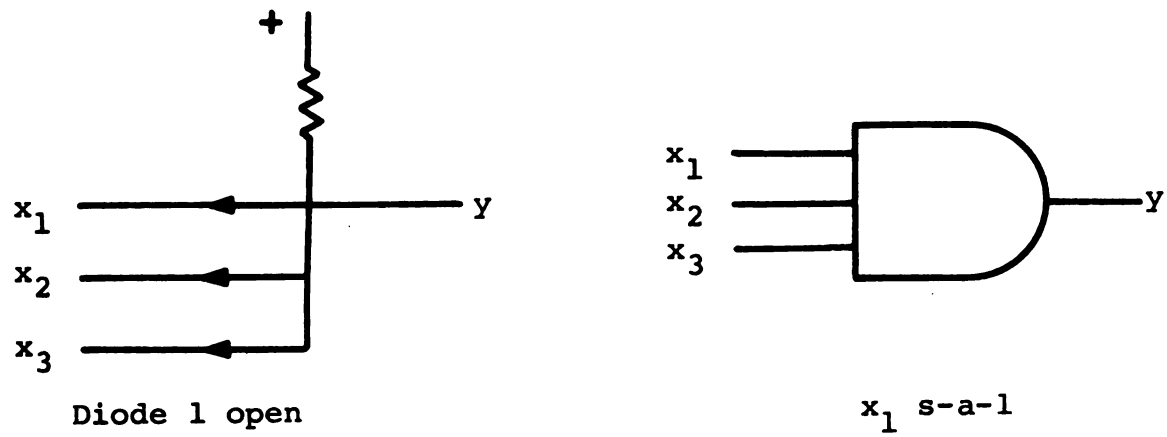
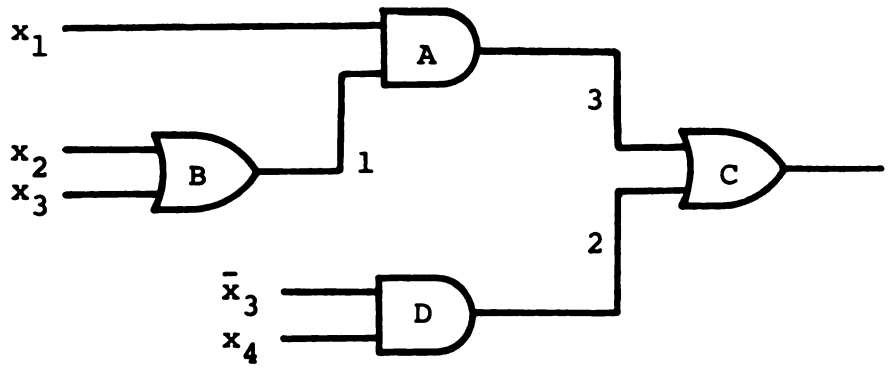


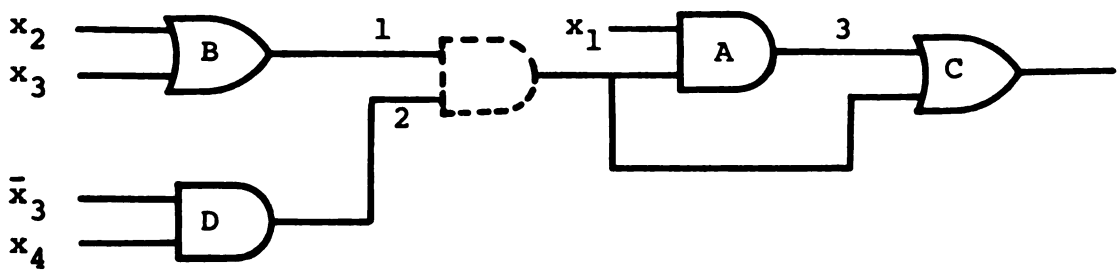
Figure 1.6. Input Diode Failure of an AND gate.

Example 1.3 (Bridging Faults)

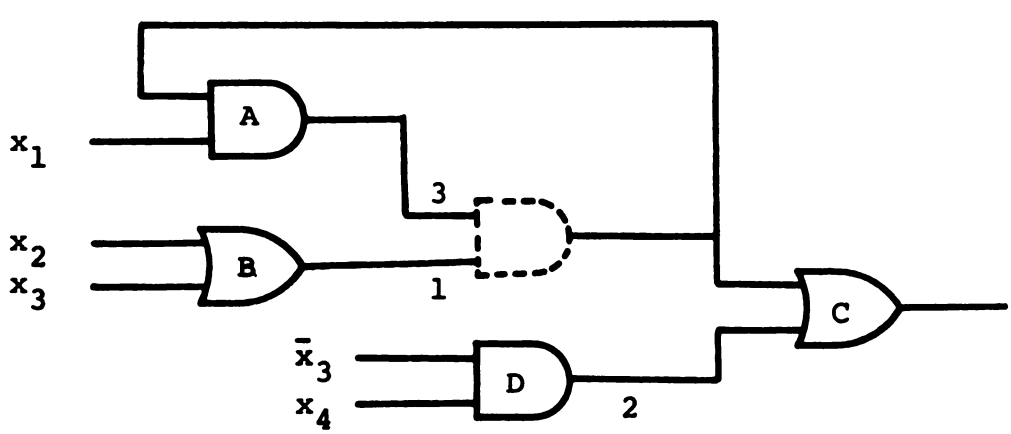
If a short circuit occurs between two lines, say the outputs of two gates, both outputs will take a common signal value. The value of such a signal could be evaluated by detailed circuit analysis. In general, it depends on the type of technology used in the realization. With current technologies, which are mainly TTL (Transistor-Transistor Logic), ANDing



(a) Normal Circuit.



(b) Equivalent Circuit if Lines 1 and 2 are Shorted.



(c) Equivalent Circuit if Lines 1 and 3 are Shorted.

Figure 1.7. Bridging Faults.

the two signals is an acceptable logical description of the fault. Thus, the effect of a short between lines 1 and 2 of the TTL circuit shown by its logical diagram in Figure 1.7(a), on the behavior of the circuit can be analyzed by inserting a virtual AND gate as shown in Figure 1.7(b). The equivalent circuit when a short occurs between lines 1 and 3 is shown in Figure 1.7(c). Notice how this fault has transformed the combinational circuit into a sequential one.

The failures indicated in examples 1.1 and 1.2, are represented logically as stuck-at faults. This is typical of a good proportion of known failures. Again, this kind of representation depends on the technology in use. A considerable amount of work dealt only with the stuck-at faults; for example [8, 21].

1.5 INTERMITTENT FAULTS

Intermittent faults in logical circuits are those faults whose effects are not present all the time. A circuit having an intermittent fault may show the effect of such a fault when an input test is applied one time, and possibly not show it when the same input test is applied other times. As indicated in the previous section, many factors could result in intermittent faults, e.g., stray capacitances, close design tolerances, fatigue, or irregular physical structure of components [34, 40].

Almost all of the work cited in the literature in the area of modeling, detection and diagnosis of faults in logical circuits deals only with permanent faults [7, 14, 31, 36]. Only one recent paper, Breur [3], deals with intermittent faults. Breur has a first order Markov chain model for the faults, but most of his results are based on the simplified assumption of a zero order Markov chain. His work deals mainly with detecting whether the circuit has a certain intermittent fault or not, i.e., the case where the circuit could have only one possible intermittent failure.

1.6 CONTRIBUTION AND ORGANIZATION OF THE THESIS

Intermittent faults constitute a respectable portion of the faults that occur in digital systems. Ad-hoc methods have been used to handle these faults in practice, while formal treatment has been completely ignored despite the need for such a tool. This thesis investigates this problem. It develops a probabilistic model for these faults and defines a criterion for fault detection in combinational circuits. The detection problem is treated as a sequential statistical decision problem using techniques similar to those employed in pattern recognition methodology [15]. A method is given for the design of optimum detection experiments, which was found to be equivalent to an integer programming problem. A diagnosis

philosophy is then presented leading to a diagnosis methodology. A method for finding suboptimal diagnosis experiments is presented since optimum experiments could only be found through enumeration of an impossibly large class of experiments.

Chapter I presents some background material and a general discussion of fault-tolerant computing. In Chapter II, an overview of diagnosis of permanent faults is presented. Chapter III presents a model for intermittent faults and discusses the problem of their detection in combinational circuits. Diagnosis of intermittent faults is dealt with in Chapter IV, while Chapter V summarizes the thesis and recommends problems for further research.

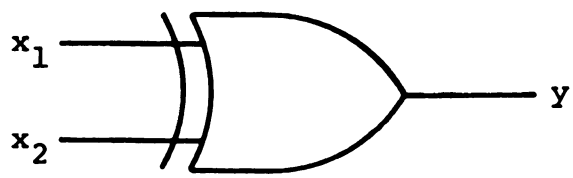
CHAPTER II

DIAGNOSIS OF PERMANENT FAULTS IN COMBINATIONAL CIRCUITS

It was indicated in Chapter I that one of the most common approaches to ensure proper operation of digital computers is fault diagnosis plus repair. Fault diagnosis deals not only with detecting faults when they occur, but also with pinpointing the locations of failures to enable repair. These diagnostic tasks are accomplished by testing the system at hand. Testing, in this sense, means applying inputs and observing the corresponding outputs. In this chapter, several test generation methods are surveyed. The total number of tests generated for a large circuit could be enormous; hence it is often desirable to select a minimal or a near-minimal subset of these tests that is sufficient for detection or diagnosis. Schemes for selection of such optimal test subsets are explored later in the chapter.

2.1 TESTS

A test for a combinational circuit is an input vector together with the observed output. A test t_j is said to detect fault f_i if, upon the application of t_j, the output vectors are different when the circuit is fault-free and when it has fault f_i. For example, consider the exclusive-OR circuit shown in Figure 2.1. Let f_1 be the fault "x_1 s-a-0," and f_2 be the fault "y s-a-1." Test t_1, denoting the input vector (0,1) (i.e., $x_1 = 0$ and $x_2 = 1$), detects neither f_1 nor f_2 since upon the application of t_1 the output is 1 when the circuit is fault-free, when it has f_1, and when it has f_2. On the other hand, tests t_2 (input vector (1,0)) and t_3 (input vector (1,1)) detect f_1 since they result in



f_1 : "x_1 s-a-0" , detected by (1,0) and (1,1).
f_2 : "y s-a-1" , detected by (0,0) and (1,1).

Figure 2.1. Example of Faults in an Exclusive-OR Circuit.

outputs of 0 and 1 respectively, if the circuit has f_1, and in outputs of 1 and 0 respectively if the circuit is fault-free. Similarly, tests t_4 (input vector (0,0)) and t_3 detect f_2.

The following remarks are now clear:

(1) In general, a test detects more than one fault. For example, test t_3 above.

(a) A fault can generally be detected by more than one test. For example, fault f_1 above.

If f_1 and f_2 are the only possible faults that could occur in the above circuit, then t_3 is sufficient to determine whether the circuit is faulty or not. If t_3 fails, i.e., produces an output that is different from that of a fault-free circuit, then the circuit is faulty. Actually this is true if any test fails. If t_3 does not fail, then the circuit must be fault-free. For that reason $\{t_3\}$ is called a <u>detection</u> set for this circuit. Generally, a detection set would contain more than one test. $\{t_2,t_4\}$ is also a detection set, while $\{t_2\}$ is not. It is clear that $\{t_3\}$ is an optimal detection set (since it has only one test) unless test costs are considered.

Detection tests tell us whether a system is faulty or fault-free, but, in general, do not completely identify the failure. Diagnosis tests will isolate the faults to a specific component or a group of components depending on both the diagnosis resolution needed and on the technology in use. For example, if the system is built with discrete

component technology, it might be necessary to pinpoint the faults down to the gate level. On the other hand, with LSI (Large Scale Integration) technology, fault identification down to the module level would be sufficient.

It is possible that a failure occurs, but no test will detect it; i.e., the effect of the fault does not change the function realized by the circuit. This is due to some sort of redundancy in the circuit. Such redundancy is not necessarily of the copious type discussed in Chapter I. For example, the circuit of Figure 2.2 exhibits some redundancy for test purposes; faults "3 s-a-0" and "4 s-a-0" will go undetected.

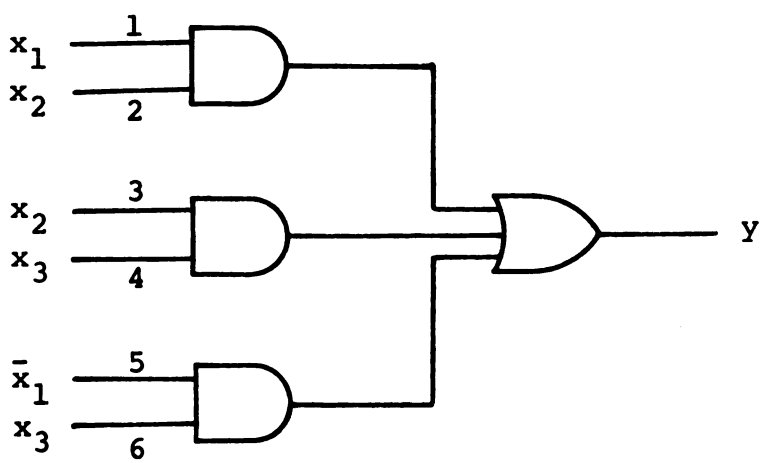


Figure 2.2. Redundancy and Fault Masking. Line "3 s-a-0" or line "4 s-a-0" will go undetected.

2.2 TEST GENERATION

In this section we discuss several methods to generate tests that would detect a given fault. In the methods presented, the circuit is assumed to be irredundant, so that redundancy would not mask the effect of the fault. Friedman [13] pointed out some of the difficulties that may arise when masked faults in redundant circuits interact with otherwise detectable faults. Also, it will be assumed that only one fault can be present at a time (single fault assumption). This assumption is quite reasonable if testing is done routinely as part of a maintainance program; then, the probability of having two faults is negligible [7]. However, this would not be a reasonable assumption for an acceptance test of a new installation where whole sections of the machine may be constructed incorrectly.

2.2.1 Truth Table Method

This is the most obvious and straightforward method for test generation. The truth tables for the normal (fault-free) circuit and for the faulty circuit are compared. An input combination is a test which detects the fault under consideration if it results in two different output vectors for the two circuits. A different truth table is constructed for every fault considered.

Example 2.1

Consider the circuit shown in Figure 2.3 and let the line α be "s-a-1." The truth tables of the output functions of the normal and faulty circuits are shown in Table 2.1.

From Table 2.1, it is clear that three tests will detect "α s-a-1." For these tests, the input vector (x_1,x_2,x_3,x_4) will take the values (0,0,0,0), (0,0,1,0) and (0,1,0,0). Alternatively, these test could be represented by the min-terms: $\bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4$, $\bar{x}_1\bar{x}_2x_3\bar{x}_4$ and $\bar{x}_1x_2\bar{x}_3\bar{x}_4$.

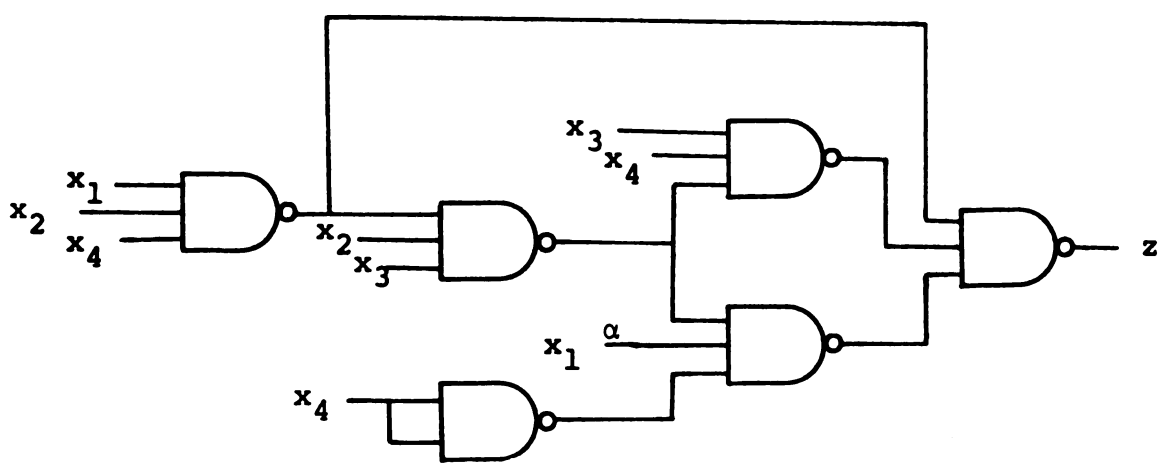


Figure 2.3. Truth Table Test Generation for "α s-a-1". z' is the output when "α s-a-1".

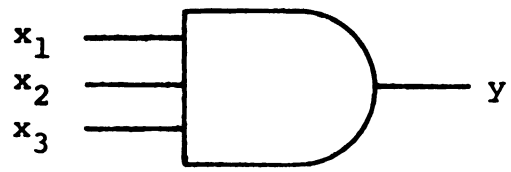
	x_1	x_2	x_3	x_4	z	z´
*	0	0	0	0	0	1
	0	0	0	1	0	0
*	0	0	1	0	0	1
	0	0	1	1	1	1
*	0	1	0	0	0	1
	0	1	0	1	0	0
	0	1	1	0	0	0
	0	1	1	1	0	0
	1	0	0	0	1	1
	1	0	0	1	0	0
	1	0	1	0	1	1
	1	0	1	1	1	1
	1	1	0	0	1	1
	1	1	0	1	1	1
	1	1	1	0	0	0
	1	1	1	1	1	1

Table 2.1. Truth Table for Normal and Faulty Circuits when "α s-a-1." (0,0,0,0), (0,0,1,0) and (0,1,0,0) detect this fault.

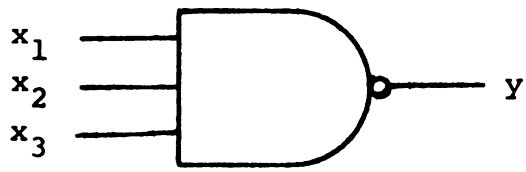
This method is effective for small circuits. If the circuit has n inputs, the number of computations needed is proportional to 2^n for every fault, which makes it prohivitive to use this method for even the moderate size circuits.

2.2.2 Path Sensitizing

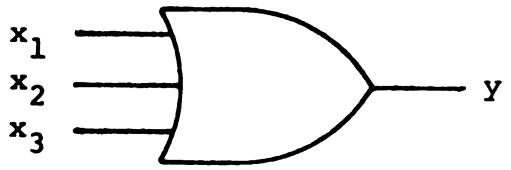
Many investigators have worked on some form or the other of this method. The name of Armstrong is usually linked with it even though he did not publish his work that is related to it. This method attempts to generate tests faster and using less memory relative to the exponential requirements of the previous method. Path sensitizing deals with stuck-at faults in circuits consisting only of NOT, AND, NAND, OR and NOR gates. The idea is to propagate a change in signal value on a faulty line in the circuit to an output terminal. A path is chosen from that location to an output, and the inputs to the gates along this path are adjusted, depending on the type of the gate, so that the gate output is sensitive only to that input that is part of the path. For AND and NAND gates, all inputs except the changing one should be 1. For OR and NOR gates, these inputs should be 0. For example, in Figures 2.4(a) and 2.4(b), if x_2 and x_3 are assigned 1, the output of the AND gate will be x_1, i.e., sensitive only to x_1, and the output of the NAND gate will be $\bar{x}_1$, i.e., sensitive only to x_1. Similarly in Figures 2.4(c) and 2.4(d), if x_1 and x_2 are assigned 0, the output of the OR gate will be x_3, and of the NOR gate will be $\bar{x}_3$, i.e., sensitive only to x_3.



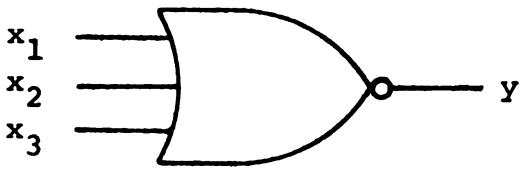
(a) AND gate.
$x_2 = x_3 = 1$ implies $y = x_1$.



(b) NAND gate.
$x_2 = x_3 = 1$ implies $y = \bar{x}_1$.



(c) OR gate.
$x_1 = x_2 = 0$ implies $y = x_3$.



(d) NOR gate.
$x_1 = x_2 = 0$ implies $y = \bar{x}_3$.

Figure 2.4. Adjusting Gate Inputs to Make Output Sensitive Only to a Single Input.

The general procedure can be summarized as follows:

(1) A failure at a point is assumed. The faulty location is assigned a value opposite to that of the fault condition. That is, a value of 1 is

assigned to a line with "s-a-0" fault and vice versa for a "s-a-1" fault.

(2) A path is chosen from the fault location to an output terminal. The inputs to the gates along this path are assigned values so as to propagate any change on the faulty line, along the chosen path, to the output terminal. This path is said to be sensitized. This phase of the procedure is called the forward-trace phase.

(3) An input vector (test) is determined by tracing back from the inputs of the gates, along the path, to the inputs of the circuit, and assigning input values to obtain the desired signals for these gates. An arbitrary choice is made when different possibilities exist. This portion of the procedure is called the backward-trace phase. It could result in more than one input vector or even in a contradiction. In case of a contradiction, the process is repeated with a different choice for the signals assigned arbitrarily, if such a choice exists, otherwise a different path should be chosen.

An example follows to illustrate this method.

Example 2.2

Consider the circuit of Figure 2.5. In the following discussion, a gate label may indicate its output to simplify notation.

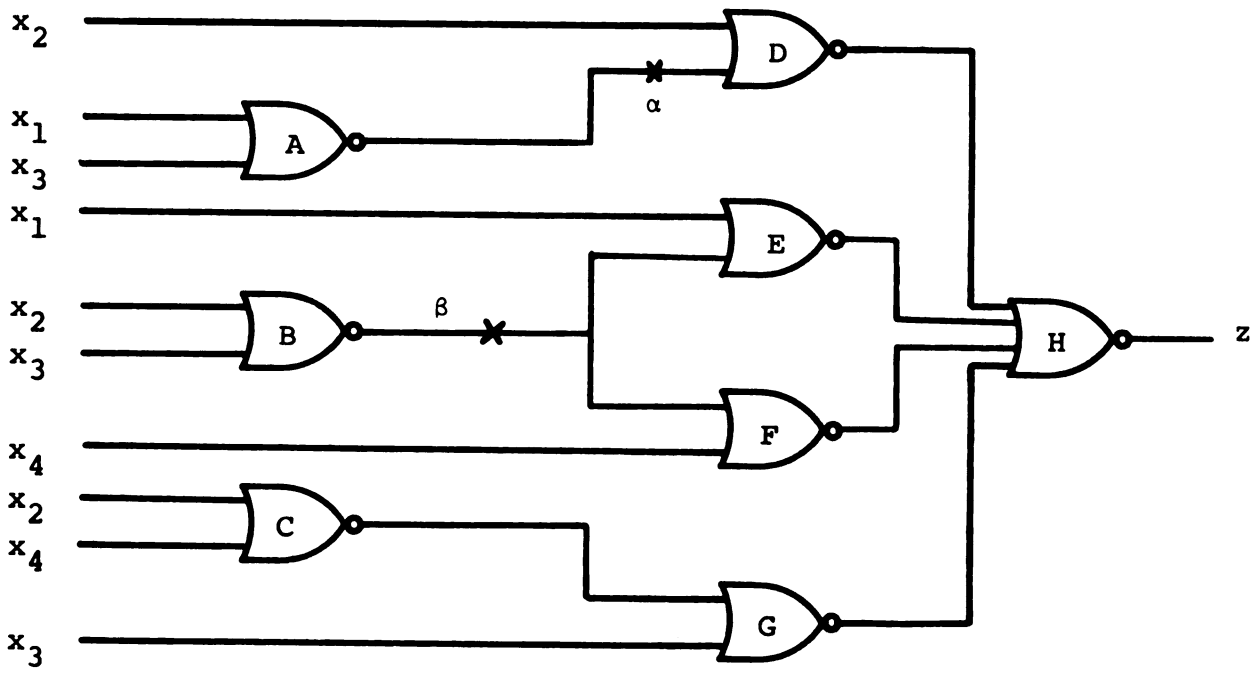


Figure 2.5. Path Sensitizing.

(1) "α s-a-1" : sensitize path DH; (1,0,1,1) and (1,0,0,0) detect it.

(2) "β s-a-0" : sensitize path EH; contradiction;
sensitize path FH, contradiction;
however, (0,0,0,0) detects it.

(1) To generate tests that detect the fault "α s-a-1":

Sensitize path DH.

Assign 0 to α.

Assign 0 to the other input of gate D, i.e., $x_2 = 0$.

Assign 0 to outputs of gates E, F and G.

This completes the forward-trace phase.

$\alpha = 0$ implies $x_1 = 1$ or $x_3 = 1$, say $x_1 = 1$.

$G = 0$ implies $x_3 = 1$ or $C = 1$ (i.e., $x_2 = 0$ and $x_4 = 0$), say $x_3 = 1$.

$F = 0$ implies $x_4 = 1$ or $B = 1$ (i.e., $x_2 = 0$ and $x_3 = 0$), say $x_4 = 1$.

$E = 0$ implies $x_1 = 1$ or $B = 1$, already satisfied by $\alpha = 0$.

This completes the backward-trace phase.

Thus we see that (1,0,1,1) is a test that detects "α s-a-1." Have we selected another signal choice, we would have obtained (1,0,0,0) as another possible test.

(2) To generate tests that detect "β s-a-0":

Sensitize path EH.

Assign 1 to β, i.e., $x_2 = 0$ and $x_3 = 0$.

Assign 0 to the other input of gate E, i.e., $x_1 = 0$.

Assign 0 to outputs of gates D, F and G.

$G = 0$ implies $x_3 = 1$ (not possible), or $C = 1$ (i.e., $x_2 = 0$ and $x_4 = 0$).

$F = 0$ implies $x_4 = 1$.

This is a contradiction since x_4 is required to be 0 and 1 at the same time.

Sensitize path FH.

Due to the symmetry of the circuit, we will end up with a similar contradiction.

However, the input vector (0,0,0,0) detects this fault since it results in a 1-output for the normal circuit and in a 0-output for the faulty circuit.

This example is due to Schneider [32] to show that this method is not algorithmic, i.e., even though a test exists, this method did not generate it. The main flaw is that only one path is allowed to be sensitized at one time. For this reason, this method is sometimes called "one-dimensional path sensitizing." The key to an algorithmic method is to simultaneously sensitize all possible paths from the site of the fault to an output. This will be necessary if the circuit has reconvergent fan-out at the site of the failure, i.e., there are two or more paths that fan-out from the fault location then subsequently reconverge as inputs to the same gate, e.g., paths EH and FH in the above example.

2.2.3 The D-Algorithm

This method was developed by Roth [31] to overcome the limitations of the path sensitizing method. This method is applicable to a wider class of faults than stuck-at type faults. Also, its use is not restricted to circuits constructed only of NOT, AND, NAND, OR and NOR gates. Most important, this method is algorithmic due to its ability to simultaneously sensitize all possible paths from the site of the fault to a circuit output. This method is sometimes referred to as two-dimensional path sensitizing. Only an overview of the algorithm is presented here. Details are found in Roth's paper. Roth's formulation is in terms of the D-Calculus; a calculus for cubical complexes. In what follows, the symbol D represents a signal that is 1 in the normal circuit and 0 in the faulty circuit. The symbol $\bar{D}$ represents a signal that is normally 0, but becomes 1 when a fault is present. The definitions of D and $\bar{D}$ could be interchanged as long as they are consistent throughout the circuit.

The elements of the D-calculus are:

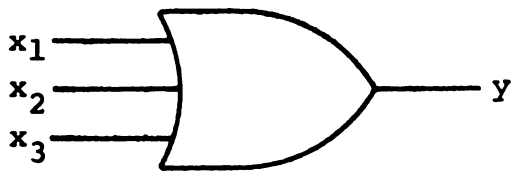
(a) Singular Cover.

The singular cover of a gate (or a block) can be considered as a concise form of its truth table. It is used to obtain the other elements of the D-calculus.

For example, Figure 2.6 shows a three-input OR gate together with its singular cover. An "x" denotes a "don't care" value. The cube 1xx1 means that the output y will take a 1-value if x_1 takes the value 1 regardless of the values of x_2 and x_3. Notice that no "x" appears in the output coordinate of the cubes of the singular cover. For details about how to obtain these cubes see [7, 14, 31].

(b) Primitive D-Cubes of a Fault.

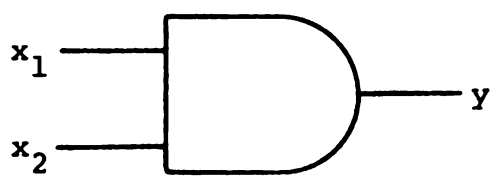
The primitive D-cubes of a fault define the inputs to a gate (block) which cause the output of the gate (block) to be different from its normal value if a given fault is present in the gate (block). These cubes are obtained by intersecting the singular covers of the normal and faulty gates (blocks). For cube



x_1	x_2	x_3	y
1	x	x	1
x	1	x	1
x	x	1	1
0	0	0	0

Figure 2.6. OR Gate and Its Singular Cover. "x" denotes a don't care.

intersection rules see [7, 14, 31]. For example consider the two-input AND gate of Figure 2.7. Suppose this gate were realized by a threshold element and that the actual threshold, due to some malfunction, has dropped below the proper value; so the gate behaves as an OR gate. The singular covers for the normal and faulty gates are shown. Intersecting these two sets of cubes we obtain $01D$ and $10\bar{D}$ as the primitive D-cubes of the fault. This means that to test for this fault, apply $0(1)$ on x_1 and $1(0)$ on x_2, if the output is 0, the circuit is normal; if



x_1	x_2	y
1	1	1
0	x	0
x	0	0

Singular cover of normal circuit.

x_1	x_2	y
1	x	1
x	1	1
0	0	0

Singular cover of faulty circuit.

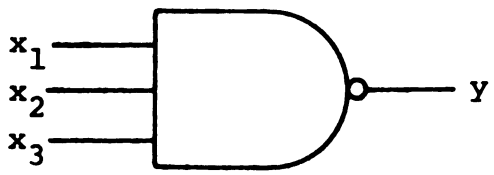
Figure 2.7. AND gate Behaving as an OR gate When Faulty. $01\bar{D}$ and $10\bar{D}$ are the primitive D-cubes of this fault.

it is 1, the circuit is faulty. Primitive D-cubes of a fault can also be obtained for blocks with more than one output. Notice that D and $\bar{D}$ appear only in the output coordinate(s).

(c) Propagation D-Cubes for Input changes(s).

These cubes define the inputs that cause the output of a gate (block) to be sensitive only to one or more of its specified inputs, thus propagating a fault on these inputs to the output. If the output is to be sensitive to more than one input, then, of course, these inputs must be related, e.g., have identical signals or one is the complement of another. This allows simultaneous multiple path sensitizing. These cubes are obtained from the singular cover of the gate (block). Some of the coordinates of the singular cover are complemented. The newly obtained cubes are intersected with the singular cover to obtain the propagation D-cubes. For example, consider the three-input NAND gate of Figure 2.8. The propagation D-cube for a change in x_1 is D11$\bar{D}$, i.e., to make the output sensitive to x_1, apply 1 to both x_2 and x_3, then the output will be the complement of the signal on x_1.

To use the D-algorithm for generating tests, the singular covers and all the propagation D-cubes for single input change for all the gates (blocks) of the circuit are



x_1	x_2	x_3	y
0	x	x	1
x	0	x	1
x	x	0	1
1	1	1	0

Singular Cover.

Figure 2.8. $D1\bar{1}\bar{D}$ is a Propagation D-cube of a NAND gate for a Change in x_1.

obtained. Only single-input propagation D-cubes are computed initially. Propagation D-cubes for multiple input changes are computed as necessary. They will be necessary when reconvergent fan-out paths are to be sensitized. A D-cube which represents partial signal values on the lines of the circuit is called a test cube.

The procedure for deriving a test for a given fault consists of two parts: the D-drive, which is analogous to the forward-trace phase of the path sensitizing method, and the consistency operation, which is analogous to the backward-trace phase of the path sensitizing method.

In the D-drive, one of the primitive D-cubes of the fault under consideration is chosen as the initial test cube. It is then intersected with one of the propagation

D-cubes. An activity vector is kept to help determine the next propagation D-cube to intersect with. The activity vector and the test cube are updated after every intersection. The process is repeated until at least one output coordinate of the circuit is obtained in the test cube. It is possible that intersections involving single-input propagation D-cubes may terminate prematurely before reaching an output terminal if the circuit has reconvergent fan-out. In this case, suitable propagation D-cubes for multiple input changes are computed and intersection proceeded. This is the case when more than one path needs to be sensitized simultaneously.

After completion of the D-drive, the consistency operation is begun. The test cube is successively intersected with the cubes of the singular cover until enough circuit inputs have been assigned to generate the signal values specified by the test cube. It is possible that some intersections will be empty, in this case it is necessary to return to the D-drive phase and obtain a new D-chain before the consistency operation can be successfully completed.

2.2.4 Algebraic Methods

The basic ideas of three algebraic methods for test generation are presented. Even though some of these methods are mathematically neat, they are only suitable

for small circuits due to the large computation and memory requirements for larger circuits. These methods are:

(a) Method of Complements.

The output function z is computed for the normal circuit as is its complement, $\bar{z}$. The corresponding output z' and its complement $\bar{z}'$ are computed for the faulty circuit. The above functions are obtained in normal form expressions. The Boolean product of z and $\bar{z}'$ and of $\bar{z}$ and z' are computed. The Boolean sum of these products represent the tests that detect the fault under consideration.

This method is somewhat better than the truth table method since it deals with terms of the normal form rather than with all the minterms (rows of the truth tables). However, we need to store all the terms of these functions which may very well exceed the available memory. It is estimated that it would take 10^9 reels of tape to store the terms of the minimal normal form of the parity check circuit for a 60 bits per word computer, even though the circuit would contain only about 63 logical blocks.

(b) Poage's Method.

Poage [29] has developed a complete and thorough method for generating tests to detect all possible stuck-at faults in combinational circuits. His work is applicable to single faults as well as multiple

faults. The disadvantage of the method is that it is practical only for relatively small circuits. In order to introduce the effect of a fault on the function realized by a circuit, he uses a kind of ternary algebra. For every line i in the circuit, three Boolean variables i_0, i_1, and i_n, called fault parameters, are defined as follows:

$i_0 = 1$ iff line i is "s-a-0",
$i_1 = 1$ iff line i is "s-a-1",
$i_n = 1$ iff line i is normal.

Only one of these parameters is equal to 1 while the other two are 0's. If the signal on line i is supposed to be y, it is replaced in the analysis by the literal y^* defined as:

$$y^* = y \cdot i_n + i_1 \ . \tag{2.1}$$

The complement $\bar{y}^*$ is defined as

$$\bar{y}^* = \bar{y} \cdot i_n + i_0 . \tag{2.2}$$

These substitutions are successively carried on until expressions z^* and $\bar{z}^*$ for the output and its complement are obtained. Substitution for fault parameters is then made to insert the fault condition. Expressions for the faulty function and its complement

are then obtained. The method of complements, described above, is then used to generate the tests.

(c) Boolean Differences.

Sellers et al. [35] have developed the Boolean difference method to generate tests, that detect stuck-at faults, from the Boolean forms representing the output functions.

The Boolean difference of an output $z(x_1,x_2,\ldots,x_n)$ with respect to one of its variables x_i is defined as follows:

$$\frac{dz}{dx_i} = z(x_1,\ldots,x_{i-1},0,x_{i+1},\ldots,x_n) \oplus z(x_1,\ldots,x_{i-1},1,x_{i+1},\ldots,x_n) \qquad (2.3)$$

In general, $\frac{dz}{dx_i}$ will be a function of some or all of the x_j's, $j \neq i$. If $\frac{dz}{dx_i} = 1$, then any change in x_i will result in a change in z regardless of the other signals x_j's, $j \neq i$. If $\frac{dz}{dx_i} = 0$, then z is independent of x_i. For single output circuits, the tests that detect "x_i s-a-0" are represented by $x_i \frac{dz}{dx_i}$, while the tests detecting "x_i s-a-1" are represented by $\bar{x}_i \frac{dz}{dx_i}$.

2.3 FAULT TABLE

In the previous section, several methods for generating tests that detect a particular fault were presented. A set of tests, detecting all faults that are likely to occur is generated. Let the set of faults of interest, $\{f_1, f_2, \ldots, f_n\}$, contain n elements, and the set of tests generated to detect these faults, $\{t_1, t_2, \ldots, t_m\}$, have m elements. As noted earlier, some of these tests detect more than one fault. An $n \times m$ fault table could be constructed from these two sets. The columns correspond to the tests and the rows correspond to the faults. The entries are zeros and ones. This table can be denoted by an $n \times m$ fault table matrix A.

Definition 2.1

The $n \times m$ fault table matrix $A = (a_{ij})$ is defined as:

$$a_{ij} = \begin{cases} 1 & \text{if test } t_j \text{ detects fault } f_i. \\ 0 & \text{if test } t_j \text{ does not detect fault } f_i. \end{cases} \tag{2.4}$$

For completeness, the set of faults may contain an additional element f_0 corresponding to the fault free circuit condition since it represents a circuit condition to be distinguished from the rest of the faults. A complete fault table is defined to be a fault table appended to it an additional row corresponding to f_0. All entries

of the row corresponding to f_0 in the complete table will be zeros. The m-dimensional binary vector corresponding to f_i $(0 \leq i \leq n)$ will be denoted by $\vec{f}_i$.

It is possible to save some of the effort used to generate tests if some relations among faults are known. If a class of faults is known to have indentical test sets, it is sufficient to generate only a test set for one of the faults in that class. Such faults form an equivalence class that has indentical rows in the fault table. This technique is called fault collapsing and is due to Schertz and Metze [33]. For example, "s-a-0" faults at an input of an AND gate and at the output of the same gate have indentical detection test sets. Practically, it does not matter which fault of these occur, since the gate has to be replaced anyway. Similarly, if all tests that detect a particular fault f_i also detect another fault f_j, then it is not necessary to generate tests to detect f_j. In this case, for every 1-entry in $\vec{f}_i$, there is a corresponding 1-entry in $\vec{f}_j$, this is referred to as row $\vec{f}_j$ dominates row $\vec{f}_i$. For example, any test that detects an input "s-a-1" for an AND gate also detects the output "s-a-1" for that gate.

If two or more tests in the fault table have identical columns, all but one of these columns can be removed since they correspond to redundant tests. Similarly a column whose entries are all zeros can be eliminated since no information will be gained when the corresponding test is

applied. A fault table (complete fault table) is said to be a reduced fault table (reduced complete fault table) if it contains no zero or redundant columns.

The following are some properties of fault tables.

Theorem 2.1

An upper bound on the number of tests in a reduced fault table with n faults is given by:

$$m \leq 2^n - 1 \tag{2.5}$$

Proof

Every column will have n entries. There are at most 2^n different binary vectors of n co-ordinates each. One of these vectors is all zeros. Thus, there are at most $2^n - 1$ non-zero different possible columns.

Q.E.D.

Theorem 2.2

Maximum diagnostic resolution (i.e., every fault is diagnosable) is possible if and only if no two rows in the complete fault table are identical.

Proof

If two rows, say $\vec{f}_i$ and $\vec{f}_j$ $(i \neq j)$, are identical, then it is impossible to distinguish between f_i and f_j since the outcome of any test applied when either fault exists will be the same.

If no two rows are identical, the following method constructs a diagnosis procedure with maximum diagnostic resolution. At any stage in the procedure apply the test with the lowest index which does not result in the same outcome for all faults which are still possibilities for the unknown circuit condition. As long as there are at least two remaining possible faults, such a test must exist, because for each pair of faults there is at least one test which distinguishes them. The procedure always terminates in fault identification since the possibilities of unknown faults are reduced at each stage. This is an existence proof. It does not necessarily mean that this is the only procedure with maximum diagnostic resolution.

Q.E.D.

Theorem 2.3

For maximum diagnostic resolution, a lower bound on the number of tests m that a fault table with n faults should have is given by:

$$m \geq \log_2 (n + 1) \qquad (2.6)$$

Proof

Suppose $m < \log_2 (n + 1)$. With m tests, there can be at most 2^m distinct rows in the complete fault table. However, we have

$$2^m < 2^{\log_2(n+1)} = n + 1, \quad \text{i.e.,}$$

$$2^m < n + 1.$$

Notice that a complete fault table with n faults has $n + 1$ rows. Thus at least two rows of the complete fault table must be identical, and, by theorem 2.2, maximum diagnostic resolution is not possible.

Q.E.D.

Example 2.3

A circuit can have one of six faults. Five tests t_1, t_2, t_3, t_4 and t_5 were generated. The detection sets for f_1, f_2, f_3, f_4, f_5 and f_6 are $\{t_1\}, \{t_1, t_2, t_4\}$,

	t_1	t_2	t_3	t_4	t_5	
f_0	0	0	0	0	0	(fault free)
f_1	1	0	0	0	0	
f_2	1	1	0	1	0	
f_3	0	1	0	1	1	
f_4	1	1	0	1	1	
f_5	0	1	1	1	1	
f_6	1	0	0	1	0	

Table 2.2. Example of a Complete Fault Table.

$\{t_2,t_4,t_5\}$, $\{t_1,t_2,t_4,t_5\}$, $\{t_2,t_3,t_4,t_5\}$ and $\{t_1,t_4\}$ respectively. The corresponding complete fault table is given in Table 2.2.

2.4 EXPERIMENTS AND THE DIAGNOSIS TREE

The process of applying tests and drawing conclusions from the observed outputs is called an experiment. Thus, a detection experiment is a sequence of tests to be applied in order to determine whether the circuit is fault free or not. On the other hand, a diagnostic experiment is a sequence of tests whose outcome is used to decide which fault, or class of faults (depending on the diagnostic resolution required), is present in the circuit. The tests used for either the detection experiment or the diagnostic experiment are selected from the set of tests generated, using any of the methods of Section 2.2, to detect the set of faults of interest. The application of all generated tests is sufficient for either detection or diagnosis. However, the tests generated are usually more than necessary for either experiment.

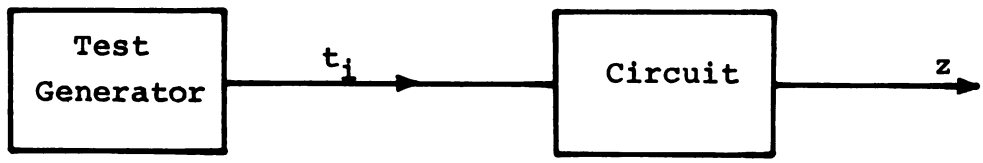
Experiments are classified into two types:

(a) Preset Experiments, where tests to be applied are completely determined in advance.

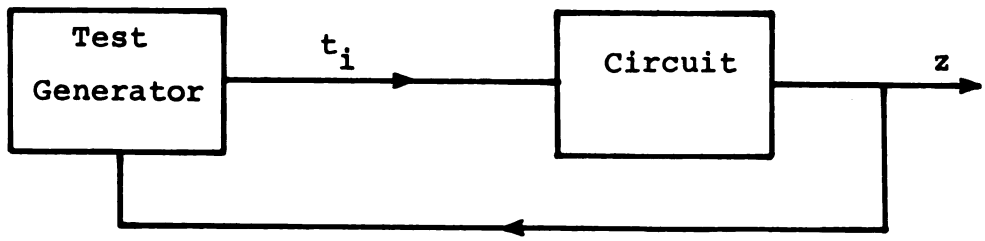
(b) Adaptive Experiments, where the test to be applied at a given stage depends on the outcome of the previous test.

In a preset experiment, the order of test application is immaterial. The experiment length (number of tests to be applied) is the same regardless of the existing fault condition. In an adaptive experiment, the order of test application is essential. The experiment length generally varies depending on the existing fault condition. Adaptive experiments are more efficient since they tend to be shorter in average length. A schematic representation of the two types of experiments is shown in Figure 2.9.

An experiment can be represented by a binary tree structure. This is due to the fact that every test t_i



(a) Perset Experiment.



(b) Adaptive Experiment.

Figure 2.9. Experiment Types.

partitions a set of fault conditions (possibly including f_0) into two classes: those that are detected by t_i (those faults with a 1 in column i in the complete fault table), and those that are not (faults with a 0 in the column i). The nodes of the tree are classes of faults representing the diagnosis resolution obtained thus far in the experiment. The root corresponds to the class of all faults including f_0. The root is defined to have level 0. The edges out of a node correspond to the two possible outcomes of the test applied at that stage. The level of a node, say node a, is one larger than the level of the node having an edge directed to node a. A tree corresponding to a preset experiment will have the same test applied at all nodes of the same level. This is not the case in a tree corresponding to an adaptive experiment. For maximum diagnostic resolution, the leaves of the tree should correspond to classes of single faults. Notice that for permanent faults, any test need only be applied once in either type of experiment.

Example 2.4

Consider the complete fault table given in Table 2.2. A preset diagnosis tree is shown in Figure 2.10. Notice that at every level the same test is applied. The symbol ϕ denotes the empty set. Whenever a class of a single fault is reached, no further edges are shown. Figure 2.11 shows an

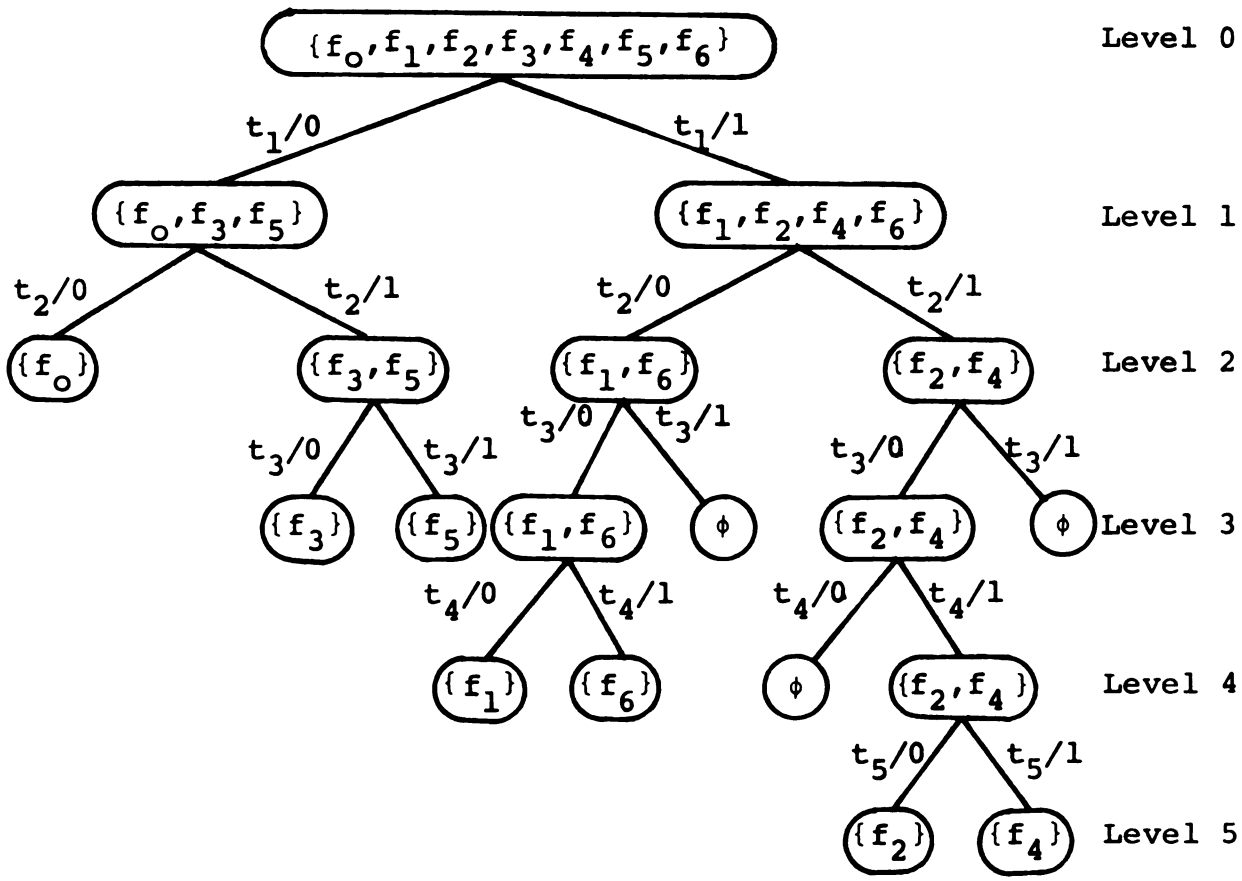


Figure 2.10. Diagnosis Tree of a Preset Experiment.

(Experiment length = 5)

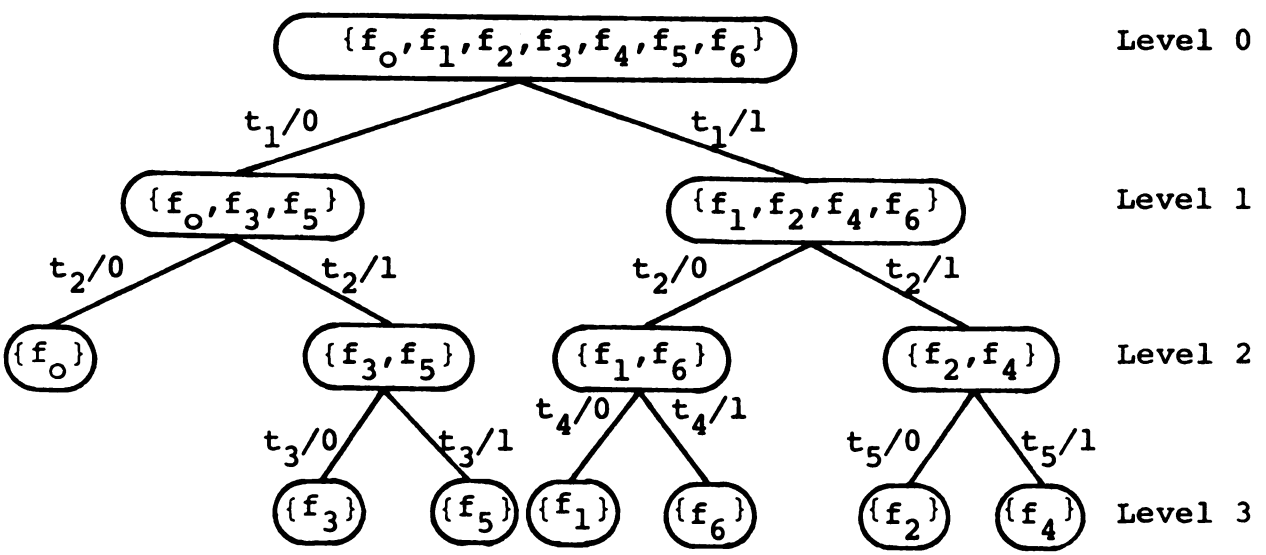


Figure 2.11. Diagnosis Tree of an Adaptive Experiment

(Maximum Experiment Length = 3)

adaptive diagnosis tree. In this case, the experiment has three tests at most.

2.5 MINIMIZATION OF PRESET EXPERIMENTS

A minimum preset experiment is a preset experiment with the least number of tests. Four approaches for selecting minimal test sets will be discussed. Exhaustive enumeration and the prime implicant method are two approaches that lead to a true minimal test sets. However, they are lengthy for large circuits. The method of test intersection and the method of distinguishibility criteria produce suboptimal solutions that are not necessarily minimal, but often close to minimal.

2.5.1 Exhaustive Enumeration

Exhaustive enumeration can be accomplished by ordering all possible tests subsets and selecting the smallest subset which is sufficient for detection or diagnosis. Obviously, this method is impossibly lengthy for even small circuits.

2.5.2 The Prime Implicant Method

This method makes use of the similarity between the problem of finding a minimal detection set and the problem of minimal cover is switching theory. Tests are analogous to prime implicants, while faults are analogous

to the minterms to be covered. Several solutions have been proposed for obtaining minimal covers, most notably, the Quine-McCluskey algorithm [23], also linear and integer programming solutions have been suggested [9, 10]. Any such solution can be directly applied to minimizing detection experiments. This method can be extended to handle diagnosis experiments. The extension is due to Poage [29]. A difference table is constructed from the original fault table. It has all of the rows of the original fault table, plus a new row for each pair of faults. An entry of a new row formed from $\vec{f}_i$ and $\vec{f}_j$ $(i \neq j)$ is the exclusive-OR of the corresponding entries in $\vec{f}_i$ and $\vec{f}_j$. The 1-entries of this formed row denote the tests that distinguish between f_i and f_j. The 1-entries of an original row can also be thought of as denoting tests that distinguish between two faults, one of them being f_0. This approach is elegant and guarantees a minimal experiment, but it is impractical for moderate or large size circuits since the solution to a covering problem for that many rows in the difference table is too long.

2.5.3 The Test Set Intersection Method

This method produces near minimal test sets. It is due to Galey, Norby and Roth [16]. It is actually a method for reducing the fault table. After table reduction, any other method of test selection is used. f_1 and f_2

are intersected by ANDing the corresponding entries. If the result is a non-zero vector, the two rows are replaced by the intersection. A 1-entry in the intersection represents a test that detects both f_1 and f_2. The process is repeated by intersecting the resultant vector with $\vec{f}_3$ and so on. If the intersection of $\vec{f}_1$ and $\vec{f}_2$ is the zero vector, we intersect $\vec{f}_1$ and $\vec{f}_3$ and proceed. If this intersection also happens to be zero, we intersect $\vec{f}_2$ and $\vec{f}_3$. The process is carried on until all rows are considered. The result is a reduced fault table. We then apply any test selection method to this table, such as the prime implicant method, or selecting one test corresponding at a 1 in every row. The test set obtained will be a suboptimal detection test set.

Example 2.5

Consider the fault table given in Table 2.2. If we intersect $\vec{f}_1$, $\vec{f}_2$, $\vec{f}_3$, $\vec{f}_4$, $\vec{f}_5$ and $\vec{f}_6$ respectively, we obtain the reduced table shown in Table 2.3. Selecting tests that cover every row in the reduced table, we obtain $\{t_1, t_4\}$ as the detection set, which happens to be a true minimum in this example.

It is to be noted that the outcome of this method depends on the order of row intersection.

If we start with the difference table then do the intersection, we can obtain a near-minimal diagnosis test set.

	t_1	t_2	t_3	t_4	t_5
$(f_1\ f_2)$	1	0	0	0	0
$(f_3,\ f_4,\ f_5,\ f_6)$	0	0	0	1	0

Table 2.3. Reduced Fault Table for Example 2.3.

2.5.4 Method of Distinguishability Criteria

This method, due to Chang [6], selects a near-minimal diagnosis test set. The basic idea is to assign weights to tests. The weight reflects the test's ability to distinguish faults. Tests are systematically selected on the basis of their weights. The weight W_i of test t_i is defined to be the number of pairs of fault conditions (including f_0) which it distinguishes, i.e.,

$$W_i = \theta_i \, \ell_i \tag{2.7}$$

where, θ_i and ℓ_i are the number of 0's and the number of 1's in the i-th column of the complete fault table. We select the test for which W_i is greatest as the first test. When j tests have been chosen, the faults would have been partitioned into b_j ($b_j \leq 2^j$) disjount blocks depending on the possible outcomes of these j tests. To select the j+1-st test of this procedure, the weights of the remaining tests are computed. At this stage, the weight of a test is the sum (over the b_j blocks) of the number of pairs of faults that it can distinguish within each

block. Let $\theta_{i,k}(\ell_{i,k})$ be the number of 0's (1's) in that portion of the i-th column of the complete fault table that correspond to block k. Thus, the weight $W_{i,j}$ of test i after j tests.

$$W_{i,j} = \sum_{k=1}^{b_j} \theta_{i,k}\, \ell_{i,k} \tag{2.8}$$

The test for which $W_{i,j}$ is maximized is chosen as the j+1-st member of the test set. The selection of tests is continued until the partition of faults can be refined no further; that is, until the weights of the unselected tests are all zeros.

Example 2.6

Consider the complete fault table given in Table 2.4.

After calculating the initial weights, we select t_1 (t_4 or t_5 will do) and rearrange the complete fault table to form Table 2.5. Select t_4 since it has the largest weight at this stage, then obtain the rearrangement shown in Table 2.6. Test t_5 has the highest weight, so it is selected. The next rearrangement is shown in Table 2.7. Select t_2. The fourth rearrangement is shown in Table 2.8. Here, every test has a zero weight, so the process terminates. The test set $\{t_1, t_2, t_4, t_5\}$ is the near-minimal diagnosis test set obtained. It happened

	t_1	t_2	t_3	t_4	t_5	t_6	t_7
f_0	0	0	0	0	0	0	0
f_1	0	0	0	1	1	0	0
f_2	0	0	0	0	1	0	0
f_3	1	0	1	0	0	0	0
f_4	1	1	1	0	0	1	0
f_5	1	0	0	1	1	0	0
f_6	1	0	0	0	1	0	0
f_7	0	0	0	1	0	1	1
f_8	1	0	1	1	0	0	0
weight	(20)	8	18	20	20	14	8

Table 2.4. Complete Fault Table and Initial Weights.

that this is a true minimal diagnosis test set for this problem.

It should be emphasized that in computing the test weights we should deal with the complete fault table and not the fault table since f_0 corresponds to a circuit condition to be distinguished from the other fault conditions. Chang, in his paper, did not include the $\vec{f}_0$ row in his analysis. As a result, the test set he obtained in his example was sufficient to distinguish among the faults, if it is known that a fault has actually occurred, but was not sufficient as a detection set.

	t_1	t_2	t_3	t_4	t_5	t_6	t_7
f_0	0	0	0	0	0	0	0
f_1	0	0	0	1	1	0	0
f_2	0	0	0	0	1	0	0
f_7	0	0	0	1	0	1	1
f_3	1	0	1	0	0	0	0
f_4	1	1	1	0	0	1	0
f_5	1	0	0	1	1	0	0
f_6	1	0	0	0	1	0	0
f_8	1	0	1	1	0	0	0
weight		4	6	(10)	10	7	3

Table 2.5. First Rearrangement of Complete Fault Table.

This method can be extended to allow for different degrees of diagnostic resolutions, i.e., it can be used to point out to a faulty block if any fault occurs in that block without pinpointing to the actual fault.

2.6 MINIMIZATION OF ADAPTIVE EXPERIMENTS

In general, the length of an adaptive experiment varies depending on the existing fault condition. Thus, the meaning of the term "minimal" in a "minimal adaptive

	t_1	t_4	t_2	t_3	t_5	t_6	t_7
f_0	0	0	0	0	0	0	0
f_2	0	0	0	0	1	0	0
f_1	0	1	0	0	1	0	0
f_7	0	1	0	0	0	1	1
f_3	1	0	0	1	0	0	0
f_4	1	0	1	1	0	1	0
f_6	1	0	0	0	1	0	0
f_5	1	1	0	0	1	0	0
f_8	1	1	0	1	0	0	0
weight			2	3	(5)	3	1

Table 2.6. Second Rearrangement of Complete Fault Table.

experiment" should be different from that used for preset experiments. An adaptive experiment is minimal if the expected experiment length is minimal. If prior probabilities p_i $(0 \leq i \leq n)$ for the different fault conditions are known, and if ℓ_i $(0 \leq i \leq n)$ is the experiment length if fault condition f_i exists, then the expected experiment length $\bar{\ell}$ is:

	t_1	t_4	t_5	t_2	t_3	t_6	t_7
f_0	0	0	0	0	0	0	0
f_2	0	0	1	0	0	0	0
f_7	0	1	0	0	0	1	1
f_1	0	1	1	0	0	0	0
f_3	1	0	0	0	1	0	0
f_4	1	0	0	1	1	1	0
f_6	1	0	1	0	0	0	0
f_8	1	1	0	0	1	0	0
f_5	1	1	1	0	0	0	0
weight				(1)	0	1	0

Table 2.7. Third Rearrangement of Complete Fault Table.

	t_1	t_4	t_5	t_2	t_3	t_6	t_7
f_0	0	0	0	0	0	0	0
f_2	0	0	1	0	0	0	0
f_7	0	1	0	0	0	1	1
f_1	0	1	1	0	0	0	0
f_3	1	0	0	0	1	0	0
f_4	1	0	0	1	1	1	0
f_6	1	0	1	0	0	0	0
f_8	1	1	0	0	1	0	0
f_5	1	1	1	0	0	0	0
weight					0	0	0

Table 2.8. Fourth Rearrangement of Complete Fault Table.

$$\bar{\ell} = \sum_{i=0}^{n} p_i \ell_i \qquad (2.9)$$

$\bar{\ell}$ is the function to be minimized for a minimal adaptive experiment. Two approaches to select a test set for minimal experiments are discussed: exhaustive enumeration, which leads to a true minimal solution, and the method of distinguishability criteria which results in a near-minimal experiment.

2.6.1 Exhaustive Enumeration

This approach was not practical for preset experiments. It is even worse for adaptive experiments, since we have to consider all possible permutations of test subsets. Unfortunately, this is the only known method that gives a true minimal adaptive experiment. Garey [32], in his Ph.D. thesis, developed a systematic method for the enumeration, and discussed special cases which have shorter solutions. This method is only suitable for small problems.

2.6.2 Method of Distinguishability Criteria

This method makes use of a figure of merit that is computed for every test that might be selected. At any stage in the experiment, the test that has the highest figure of merit is selected. The process is repeated until

no further diagnosis is possible. This approach results in locally optimized procedures, rather than globally optimized ones. Chang's method, discussed in subsection 2.5.4, can be easily adjusted to apply for adaptive experiments. The test weight is considered its figure of merit. Instead of adding up the number of pairs a test can distinguish in all blocks, only one block is considered.

Another figure of merit, based on information gain, has been suggested by Brule´ et al. [4]. The initial uncertainty A_0 about the fault condition is:

$$A_0 = - \sum_{i=0}^{n} p_i \log_2 p_i \ . \qquad (2.10)$$

At every stage, the test that results in maximum information gain is selected. For example, to select the first test, the information gain ΔA_i due to every test t_i is computed. If test t_i is applied, it will fail with probability q_i, narrowing down the fault condition to a smaller block. Let the uncertainty among this block be $A_{1,1}$. It is also true that if t_i is applied, it will not fail with probability $(1-q_i)$, narrowing down the fault condition to a different smaller block. Let the uncertainty among this block be $A_{1,0}$. The information gain ΔA_i due to t_i is:

$$\Delta A_i = A_0 - (q_i A_{1,1} + (1-q_i) A_{1,0}). \qquad (2.11)$$

The test with the largest information gain is selected. The probability of failure q_i can be calculated from the prior probabilities. The computations at later stages are computed in a similar fashion.

CHAPTER III

DETECTION OF INTERMITTENT FAULTS IN COMBINATIONAL CIRCUITS

Intermittent faults in digital circuits are those faults whose effects are not present all the time. A probabilistic model for intermittent faults is presented. Permanent faults are a special case in this model. Detection of intermittent faults through repeated application of tests that detect such faults, as if they were permanent, is suggested, together with a detection criterion. The detection procedure proposed is equivalent to a sequential statistical decision problem. Optimization of detection experiments is discussed later in the chapter.

Assumptions similar to those made in Section 2.2 for the permanent faults case are used here, namely:

(1) The single fault assumption; i.e., the circuit can have only one fault during the testing experiment.

(2) Irredundancy; i.e., the circuit is assumed to be irredundant, thus the effect of a fault would not be masked.

Moreover, we will assume that:

(3) The faults are well behaved, i.e., during an application of a test, the circuit behaves as if it is fault free or having a permanent fault for that period of time. That is, during the application of a test, the effect of an intermittent fault is either not present at all, present for a relatively brief interval of time that the response to the test is the same as if no failure occurred, or present for a long enough period of time so that the fault appears to be permanent for that application of this test.

(4) The faults are signal independent; i.e., the presence of a fault does not depend on the signal values existing in the circuit.

3.1 THE MODEL

The model proposed is a probabilistic one. The basic elements of the model are defined below. Assumptions about the model are indicated as we proceed. The notation used is very close to that employed in pattern recognition literature, for example, see [12, 15, 26].

State Space Ω: $\Omega = \{\omega_0, \omega_1, \ldots, \omega_n\}$

Each point in Ω represents a state of the circuit;

ω_0: denotes the state of being fault free;

ω_i: $(1 \leq i \leq n)$ denotes the state of having intermittent fault number i.

It is assumed that the state of the circuit can be described by a single element from Ω (which one, is not known); i.e., the circuit is fault free or it has only one of n possible intermittent faults (the single fault assumption). During the testing experiment, the circuit is assumed to stay in the same state. If the circuit is in state ω_i $(1 \leq i \leq n)$, it does not mean that the effect

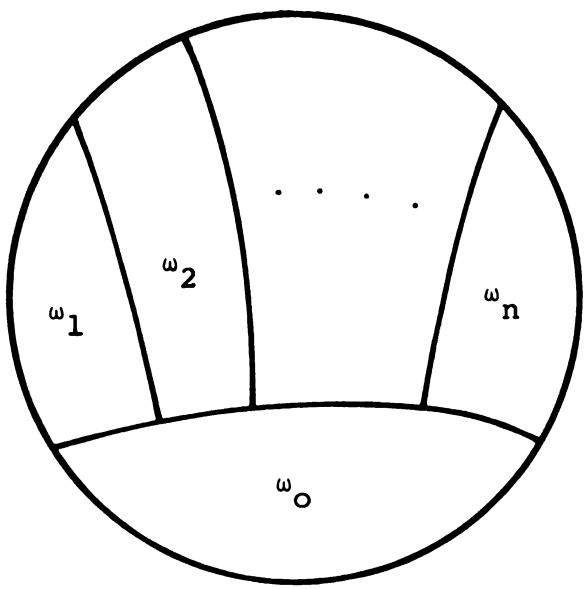


Figure 3.1. State Space Ω. State ω_i corresponds to the circuit having intermittent fault number i.

of the i-th fault on the behaviour of the circuit will be present all the time; this is due to the intermittency of the fault.

Permanent Fault Space F: $F = \{f_0, f_1, \ldots, f_n\}$

There is a one to one correspondence between F and Ω. f_i, $(1 \leq i \leq n)$, is the permanent fault that the circuit would have, if the effect of ω_i is present all the time and f_0 denotes the fault free condition.

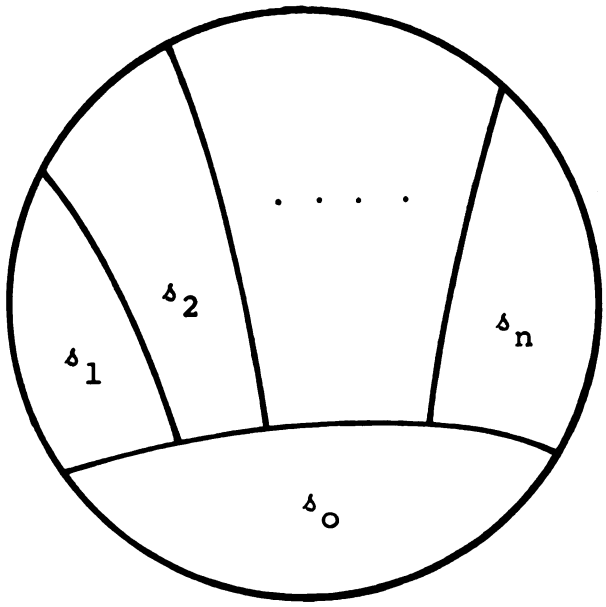


Figure 3.2. Sample Space S. Sample point δ_i corresponds to the circuit behaving as if it has permanent fault f_i.

Sample Space S: $S = \{\delta_0, \delta_1, \ldots, \delta_n\}$

Every point in S corresponds to the outcome of a random experiment. Conceptually, the random experiment can be thought of as testing all circuit components. With the single fault assumption, the outcome of such an experiment would be:

δ_o: all components are fault free, or

δ_i: $(1 \leq i \leq n)$, the component that pertains to the i-th fault is faulty.

Observing δ_0 in this conceptual random experiment, we cannot infer that the state of the circuit is ω_0; in fact, it could be any ω_i $(0 \leq i \leq n)$ due to the intermittency of the faults. However, observing δ_i $(1 \leq i \leq n)$, we could infer that the state of the circuit is ω_i for sure.

Test Set τ: $\tau = \{t_1, t_2, \ldots, t_m\}$

The test set τ is a complete test set that would detect all faults under consideration if they were permanent. This set can be generated using any of the methods discussed in Section 2.2 to detect permanent faults. In general, a test t_j will detect more than one fault. Let

$$\{f_{j_1}, f_{j_2}, \ldots, f_{j_{m_j}}\}$$

be the set of faults detected by t_j (those faults that correspond to the 1-entries in column j of the fault

table [definition 2.1]). In terms of the random experiment, test t_j can be thought of as testing the components corresponding to faults f_{j_1}, f_{j_2}, ..., and $f_{j_{m_j}}$. The test has two possible outcomes:

(a) all components tested are fault free, or

(b) one component is faulty, which one, is not known.

Test Subset τ_i: $(1 \leq i \leq n)$

A subset of τ that contains all the tests in τ that detect f_i. The elements of this subset correspond to the 1-entries in the vector $\vec{f}_i$ of the fault table.

Random Variable Set T: $T = \{T_1, T_2, \ldots, T_m\}$

Every test t_j $(1 \leq j \leq m)$ in τ defines a random variable T_j on S. If test t_j is applied to the circuit and it fails, i.e., the observed output is different from that of a normal (fault free) circuit, the value of T_j is defined to be 1. On the other hand if t_j does not fail (i.e., produces an output identical to that of a normal circuit), then the value of T_j is defined to be 0. Formally, T_j is a function from S into the set $\{0,1\}$ defined as:

$$T_j : S \rightarrow \{0,1\} \quad , \quad (1 \leq j \leq m) \quad ;$$

$$T_j(\delta_i) = \begin{cases} 1 & 1 \leq i \leq n, \text{ and } t_j \text{ detects fault } f_i. \\ 0 & \text{otherwise.} \end{cases} \qquad (3.1)$$

Random Variable Subset T_i: $(1 \leq i \leq n)$

T_i is a subset of T; it contains all the random variables that correspond to tests in τ_i.

Action Space $\Omega \times S$:

The probability measure to be used, is defined on the action space $\Omega \times S$ in order to be able to use prior information about the distribution over Ω. Since the presence of the effect of an intermittent fault corresponds to a single point in S, many of the points in $\Omega \times S$ will have zero probability, namely:

$$P(\omega_0, \delta_i) = 0 \quad , \quad 1 \leq i \leq n$$

and, $P(\omega_i, \delta_j) = 0 \quad , \quad 1 \leq i,j \leq n \quad \text{and} \quad i \neq j$:

That is, only $2n+1$ points in $\Omega \times S$ have non-zero probabilities, namely:

$$(\omega_0, \delta_0) \ , \ (\omega_i, \delta_i) \ , \ (\omega_i, \delta_0) \ , \ 1 \leq i \leq n.$$

Every point ω_i in Ω defines an event on $\Omega \times S$ as follows:

$$\omega_i = \{(\omega_i, \delta_j) \mid 0 \leq j \leq n\} \tag{3.2}$$

It follows that only one point in $\Omega \times S$ of those contained in the ω_0 event has non-zero probability, namely: (ω_0, δ_0). Similarly, only two points in $\Omega \times S$ in the $\omega_i (1 \leq i \leq n)$ event, have non-zero probabilities, namely: (ω_i, δ_0) and (ω_i, δ_i).

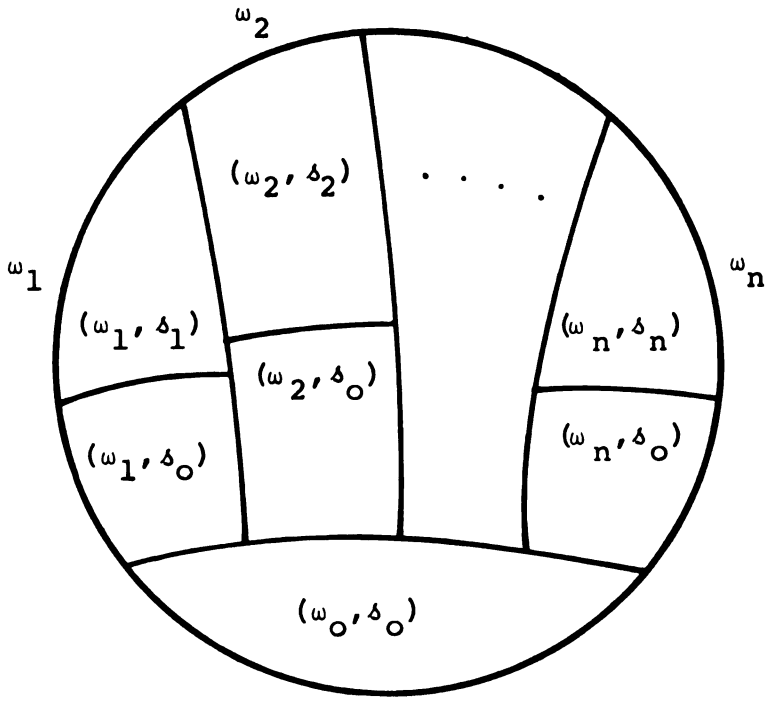


Figure 3.3. Points in Ω x S With Non-Zero Probabilities.

Similarly, every point δ_i in S defines an event on Ω x S as follows:

$$\delta_i = \{(\omega_j, \delta_i) \mid 0 \leq j \leq n\} \tag{3.3}$$

Only one point in Ω x S, of those in the event δ_i $(1 \leq i \leq n)$, has non-zero probability, namely (ω_i, δ_i), and all the $n+1$ points in δ_0 have non-zero probabilities.

k-Dimensional Random Variable Set T^k (T_i^k, $1 \leq i \leq n$):

$T^k(T_i^k)$ is the cartesian product of the set $T(T_i)$ by itself k times. An element b from $T^k(T_i^k)$ can be written as a k-tuple of random variables from $T(T_i)$, for example:

$$b = (T_{j_1}, T_{j_2}, \ldots, T_{j_k})$$

$$T_{j_r} \in T \quad (T_{j_r} \in T_i) \text{ , for } 1 \leq r \leq k \text{ .}$$

The outcome of an experiment in which k tests from $\tau(\tau_i)$ are applied to the circuit can be represented by such an element b. Its value is a k-dimensional binary vector. The notation $b = \vec{0^k}$ will be used to denote a k-dimensional binary vector, all of its entries are 0's. This corresponds to an experiment of k tests; none of them has failed.

It should be kept in mind that during the application of an experiment (a sequence of tests), the circuit will stay in the same state ω_i ($0 \leq i \leq n$). During the course of the testing experiment, the sample space point will be δ_0 all the time or changing randomly between δ_0 and δ_i (for some, but fixed i) only.

Prior Probabilities $P(\omega_i)$: ($0 \leq i \leq n$)

$p_i = P(\omega_i)$, $0 \leq i \leq n$, is the prior probability that the circuit is in state ω_i. The values of these p_i's are assumed to be known in the model. These values could

be obtained empirically, from manufacturer information, or from experience.

Conditional Probability of Malfunction $P(\delta_i/\omega_i)$: $(1 \leq i \leq n)$

$e_i = P(\delta_i/\omega_i)$, $1 \leq i \leq n$, is the probability that the effect of intermittent fault ω_i will be present knowing that the circuit already has intermittent fault ω_i. The values of the e_i's are assumed to be constants, also known in the model.

3.2 DETECTION OF INTERMITTENT FAULTS

In general, fault detection means applying tests from a certain test set to find out whether a given circuit is fault free or not. To detect permanent faults, any particular test need only be applied once. The approach proposed for detection of intermittent faults employs repeated application of tests from the test set τ. The repetition of a particular test is needed since the circuit might have an intermittent fault that could be detected by this test but the effect of such a fault is not always present when this test is applied. After applying a particular test repeatedly and obtaining outputs identical with those of a fault free circuit, it could be reasoned that the circuit might still have an intermittent failure whose effect has not yet been observed. Should the test

be repeated forever? What is needed is a "wise" stopping rule. It is assumed that testing is done by a fast machine, say a computer, so repetitions, possibly in the order of millions of times, can be done fast and fairly easy. The problem of detection of intermittent faults can then be viewed as a statistical decision problem. An appropriate decision rule is one that is sequential in nature. The one suggested makes use of the posterior probabilities.

Posterior Probabilities $P(\omega_i/T_j)$

After applying a test t_j to the circuit and observing the output, the probabilities $P(\omega_i/T_j)$, $0 \leq i \leq n$, can be calculated using Bayes' Rule:

$$P(\omega_i/T_j) = \frac{P(T_j/\omega_i)\ P(\omega_i)}{\sum_{k=0}^{n} P(T_j/\omega_k)P(\omega_k)} \tag{3.4}$$

Details and proof of Bayes' Rule are found in [11].

As the prior probabilities $P(\omega_i)$, $0 \leq i \leq n$, reflect the beliefs about the condition of the circuit before any test is applied, the posterior probabilities $P(\omega_i/T_j)$ are an updated version of these beliefs after applying test t_j and observing the output. These posterior probabilities will be used as prior probabilities next time a test is applied. The following detection procedure is suggested.

Detection Procedure:

(1) Apply an appropriate test t_j from τ. The meaning of "appropriate" will be clear after defining the decision rule. If the output of the circuit is different from that of the fault free circuit (i.e., if $T_j = 1$) decide that the circuit has an intermittent fault and stop; otherwise go to (2).

(2) At this point, the output so far is identical to that of a fault free circuit. Calculate the posterior probabilities $P(\omega_i/T_j = 0)$, $0 \leq i \leq n$, using Bayes' Rule.

(3) Using the posterior probabilities apply a decision rule, and decide one of two things:

   (a) Request the application of a particular test and repeat, i.e., go to (1), or;

   (b) Decide that the circuit is fault free and stop.

Decision Rule:

The decision rule mentioned above can be freely selected to satisfy a set of conditions that represent acceptable measures for deciding that the circuit is fault free. Moreover, the decision rule must ensure the termination of the detection procedure in a finite amount of time, i.e., ensure that we have a detection algorithm.

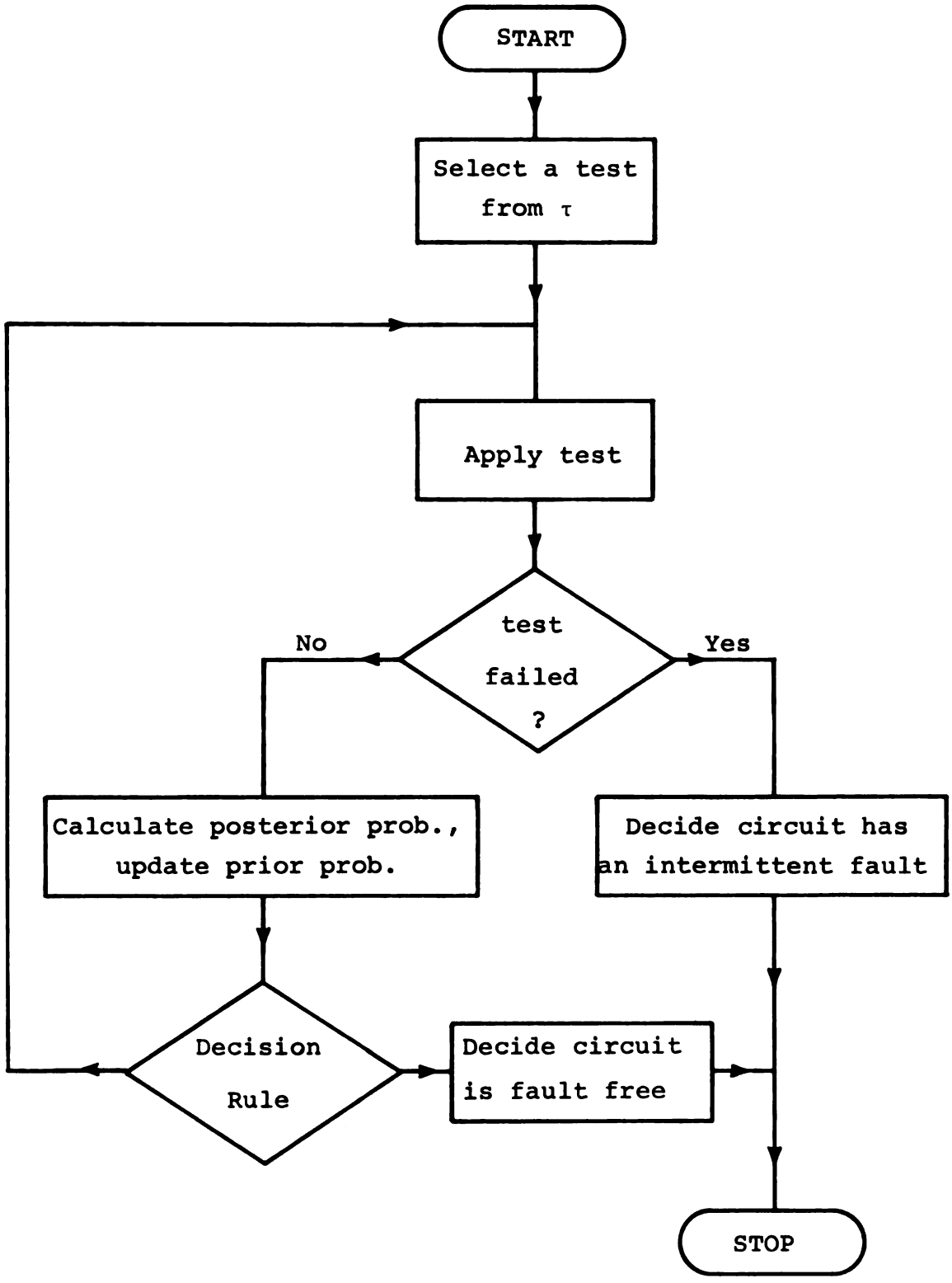


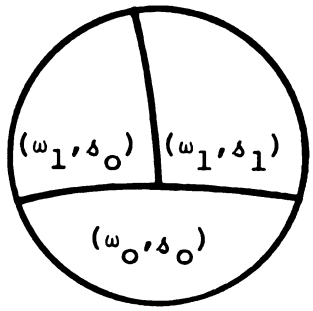
Figure 3.4. Flow Chart for Detection Procedure.

Examples of typical decision rules are presented later in this section.

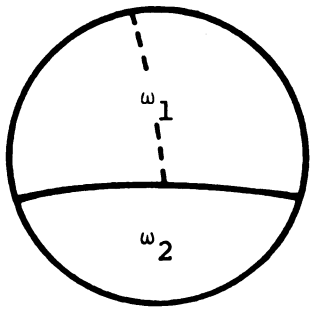
3.2.1 A Simple Case

Let $\Omega = \{\omega_0, \omega_1\}$, i.e., one particular intermittent fault could possibly exist in the circuit.

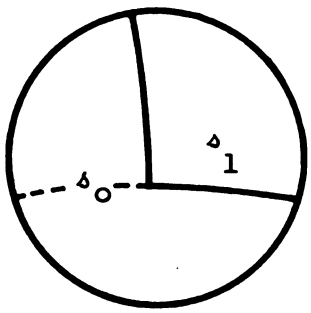
Let $\tau = \{t_1\}$. If τ is not a singleton, choose any element of τ and disregard the others, since it will function as well as any other test in τ. Obviously, any test in τ will detect f_1.



(a) Non-Zero Probability Points.



(b) State Space.



(c) Sample Space.

Figure 3.5. Ω x S Space for a Simple Case.

It follows that:

$S = \{\delta_0, \delta_1\}$

$F = \{f_0, f_1\}$

$\omega_1 = \{(\omega_1, \delta_0), (\omega_1, \delta_1)\}$

$T_1(\delta_0) = 0$

$T_1(\delta_1) = 1$

Consider the following distribution on $\Omega \times S$:

$$P(\omega_1) = p_1$$

$$P(\omega_0) = p_0 = 1 - p_1$$

$$P(\delta_1/\omega_1) = e_1$$

$$P(\omega_i/T_j) = \frac{P(T_j/\omega_i)\ P(\omega_i)}{\sum_{k=0}^{1} P(T_j/\omega_k)\ P(\omega_k)}$$

$$p_{1,1} = P(\omega_1/T_1 = 0) = \frac{P(T_1=0/\omega_1)\ P(\omega_1)}{P(T_1=0/\omega_0)\ P(\omega_0) + P(T_1=0/\omega_1)\ P(\omega_1)}$$

$$= \frac{(1-e_1)\ p_1}{1 - e_1\ p_1} \qquad (3.5)$$

$(1-e_1)$ is less than $(1-e_1\ p_1)$, therefore:

$$p_{1,1} < p_1 \qquad (3.6)$$

Similarly,

$$p_{0,1} = P(\omega_0/T_1 = 0) = \frac{p_0}{1-e_1\ p_1} \qquad (3.7)$$

it follows that:

$$p_{0,1} > p_0 \qquad (3.8)$$

$$P(\omega_1/T_1 = 1) = \frac{P(T_1 = 1/\omega_1)\ P(\omega_1)}{P(T_1 = 1/\omega_0)\ P(\omega_0) + P(T_1 = 1/\omega_1)\ P(\omega_1)} = 1 \quad (3.9)$$

Similarly,

$$P(\omega_0/T_1 = 1) = 0 \quad (3.10)$$

The interpretation of (3.6) and (3.8) is that, if t_1 is applied and the circuit produced a good output (identical with that of a fault free circuit), then our certainty about the circuit being faulty will decrease and our certainty about the circuit being fault free will increase; which is quite reasonable. On the other hand, the interpretation of (3.9) and (3.10) is that, once a bad (different from that of a fault free circuit) output is observed upon the application of t_1, then we know for sure that the circuit has an intermittent fault.

The posterior probabilities after applying t_j for $k+1$ times and observing $k+1$ good outputs are:

$$p_{1,k+1} = P(\omega_1/T_1=0, b=\vec{0^k}\ ;\ b \in T^k)$$

$$= \frac{P(T_1=0/\omega_1, b=\vec{0^k})\ P(\omega_1/b=\vec{0^k})}{P(T_1=0/\omega_0, b=\vec{0^k})\ P(\omega_0/b=\vec{0^k}) + P(T_1=0/\omega_1, b=\vec{0^k})\ P(\omega_1/b=\vec{0^k})}$$

$$= \frac{(1-e_1)\ p_{1,k}}{(1-p_{1,k}) + (1-e_1)\ p_{1,k}} = \frac{(1-e_1)\ p_{1,k}}{1-e_1\ p_{1,k}} \quad (3.11)$$

Similarly,

$$p_{0,k+1} = P(\omega_0/T_1=0, b=\vec{0}^k \; ; \; b \in T^k)$$

$$= \frac{p_{0,k}}{1 - e_1 + e_1 p_{0,k}} = \frac{p_{0,k}}{1 - e_1 (1-p_{0,k})} \cdot \qquad (3.12)$$

From (3.11) and (3.12) it follows that:

$$p_{1,k+1} < p_{1,k} \; , \qquad (3.13)$$

$$p_{0,k+1} > p_{0,k} \cdot \qquad (3.14)$$

Theorem 3.1

$$\text{(a)} \quad p_{1,k} = \frac{(1-e_1)^k p_1}{p_0 + (1-e_1)^k p_1} \qquad (3.15)$$

$$\text{(b)} \quad p_{0,k} = \frac{p_0}{p_0 + (1-e_1)^k p_1} \qquad (3.16)$$

Proof

(a) By induction.

$$k = 1 : \quad p_{1,1} = \frac{(1-e_1) p_1}{p_0 + (1-e_1) p_1} = \frac{(1-e_1) p_1}{1 - e_1 p_1}$$

which is true from (3.5).

Assume true for k: from (3.11),

$$p_{1,k+1} = \frac{(1-e_1) p_{1,k}}{1 - e_1 p_{1,k}}$$

$$= \frac{(1-e_1) (1-e_1)^k p_1}{p_0 + (1-e_1)^k p_1 - e_1 (1-e_1)^k p_1}$$

$$= \frac{(1-e_1)^{k+1} p_1}{p_1 + (1-e_1)^k (1-e_1) p_1}$$

$$= \frac{(1-e)^{k+1} p_1}{p_0 + (1-e_1)^{k+1} p_1}$$

i.e., true for k+1

Q.E.D.

(b) $p_{0,k} = 1 - p_{1,k}$

$$= \frac{p_0}{p_0 + (1-e_1)^k p_1}$$

Q.E.D.

Corollary 3.1

(a) $$\lim_{k\to\infty} p_{1,k} = 0 \qquad (3.17)$$

(b) $$\lim_{k\to\infty} p_{0,k} = 1 \qquad (3.18)$$

Proof:

$(1-e_1)$ is less than 1, therefore,

$$\lim_{k\to\infty} (1-e_1)^k = 0 \ . \qquad (3.19)$$

(a) Using (3.15) and (3.19) :

$$\lim_{k\to\infty} p_{1,k} = \frac{0}{p_0} = 0 \ .$$

Q.E.D.

(b) Using (3.16) and (3.19) :

$$\lim_{k\to\infty} p_{0,k} = \frac{p_0}{p_{0+0}} = 1 \ .$$

Q.E.D.

Definition 3.1

If the test t_1 is applied k times and good output (identical with that of a fault free circuit) was observed every time, the likelihood ratio λ_k is defined as:

$$\lambda_k = \frac{p_{1,k}}{p_{0,k}} \ .$$

From (3.15) and (3.16), it follows that:

$$\lambda_k = \frac{(1-e_1)^k \, p_1}{(1-p_1)} \qquad (3.20)$$

Corollary 3.2

$$\lambda_{k+1} < \lambda_k \qquad (3.21)$$

Proof:

from (3.20) : $\dfrac{\lambda_{k+1}}{\lambda_k} = (1-e_1) < 1$

i.e., $\lambda_{k+1} < \lambda_k$

Q.E.D.

Corollary 3.3

$$\lim_{k\to\infty} \lambda_k = 0 \qquad (3.22)$$

Proof:

From (3.19) and (3.20):

$$\lim_{k\to\infty} \lambda_k = \frac{0}{1-p_1} = 0$$

Q.E.D.

A Decision Rule

Earlier in this section it was stated that the posterior probabilities will be used in the decision rule to decide whether to continue testing or that the circuit is fault free. The following decision rule is suggested to be part of the detection procedure:

> If the posterior probability $p_{1,k}$ goes below a certain threshold s ($0 < s < 1$), decide that the circuit is fault free and stop, otherwise apply t_1 and repeat (i.e., go to step (1) of the detection procedure).

This decision rule is an acceptable one, because it guarantees termination of the detection procedure in a finite amount of time by virtue of the fact that $p_{1,k}$ is monotonically decreasing (from (3.13)), and that as k increases it can get below the threshold s (from (3.17)).

The value of the threshold s can be chosen by considering the probability of error (probability of deciding that the circuit is fault free while it is actually faulty). This of course depends on how critical the proper operation

of the circuit is. Also this affects the length of the testing experiment which is a factor that can be taken into consideration when choosing s.

To determine a least upper bound on the length of the experiment, find k for which $p_{1,k} < s$. From (3.15):

$$\frac{(1-e_1)^k p_1}{(1-p_1) + p_1 (1-e_1)^k} < s$$

$$(1-e_1)^k p_1 < s \quad [(1-p_1) + p_1 (1-e_1)^k]$$

$$(1-e_1)^k (p_1 - s p_1) < s \quad (1-p_1)$$

$$(1-e_1)^k < \frac{s(1-p_1)}{p_1 (1-s)}$$

$$k \log (1-e_1) < \log \frac{s (1-p_1)}{p_1 (1-s)}$$

or,
$$k > \frac{\log \frac{s (1-p_1)}{p_1 (1-s)}}{\log (1-e_1)} \qquad (3.23)$$

Note that $\log(1-e_1)$ is negative.

Example 3.1

Among the gates produced by a certain manufacturer, it is estimated that for about 0.01% of them, the gap between the ON and OFF voltages is smaller than some critical value. If the gap is below the critical value, the gate will malfunction 5% of the time. Find a least upper bound for the length of the testing experiment.

Solution

From the given figures, the following quantities are estimated as shown:

$$p_1 = 10^{-4}$$

$$e_1 = 0.05$$

If we choose the threshold s to be 10^{-6}, then :

$$k > \frac{\log \frac{10^{-6}(1-10^{-4})}{10^{-4}(1-10^{-6})}}{\log 0.95}$$

or $k \geq 91$

Another Decision Rule

The rule suggested here compares the likelihood ratio (which is a function of the posterior probabilities) with a threshold u $(u > 0)$ as follows:

If λ_k goes below u decide that the circuit is fault free and stop, otherwise apply t_1 and repeat (i.e., go to step (1) of the detection procedure).

This decision rule is also an acceptable one, since it guarantees termination of the detection procedure in a finite amount of time by virtue of the fact that λ_k is monotonically decreasing (from (3.21)), and that as k increases it can get below u (from (3.22)). Actually this is the optimum Bayesian decision rule (that minimizes the average loss) for the (0,1) loss function [15].

The value of the threshold u could be chosen in a fashion similar to that of selecting s.

The least upper bound on the length of the testing experiment is obtained by finding k such that $\lambda_k < u$. From (3.20):

$$\frac{(1-e_1)^k \, p_1}{(1-p_1)} < u$$

$$(1-e_1)^k < \frac{u\,(1-p_1)}{p_1}$$

$$k \log (1-e_1) < \log \frac{u\,(1-p_1)}{p_1}$$

or,
$$k > \frac{\log \dfrac{u\,(1-p_1)}{p_1}}{\log (1-e_1)} \qquad (3.24)$$

Which is similar to the results in Wald [39] for binomial samples.

3.2.2. The General Case

This is the case described by the model in Section 3.1. The basic assumptions are: (1) the circuit is irredundant, (2) the circuit can have only a single intermittent fault out of the n faults considered, (3) the faults are well behaved, and (4) the faults are signal independent.

Consider the probability distribution on $\Omega \times S$ governed by the following conditions:

$$P(\omega_i) = p_i \quad , \quad 0 \leq i \leq n$$

$$P(\delta_i/\omega_i) = e_i \quad , \quad 1 \leq i \leq n$$

Definition 3.2

The membership functions M_j $(1 \leq j \leq m)$ from the the set $\{0,1,\ldots,n\}$ into the set $\{0,1\}$ are defined as:

$$M_j(0) = 0 \quad ,$$

$$M_j(i) = \begin{cases} 1 & \text{if } t_j \text{ detects } f_i \\ 0 & \text{if } t_j \text{ does not detect } f_i \end{cases} . \qquad (1 \leq i \leq n)$$

The posterior probabilities are now calculated using Bayes' Rule (Equation 3.4)),

$$P(\omega_i/T_j) = \frac{P(T_j/\omega_i)\ P(\omega_i)}{\sum_{\ell=0}^{n} P(T_j/\omega_\ell)\ P(\omega_\ell)}$$

$$P(T_j = 1/\omega_i) = \begin{cases} e_i & \text{if } t_j \text{ detects } f_i \\ 0 & \text{if } t_j \text{ does not detect } f_i \end{cases} .$$

Similarly,

$$P(T_j = 0/\omega_i) = \begin{cases} 1-e_i & \text{if } t_j \text{ detects } f_i \\ 1 & \text{if } t_j \text{ does not detect } f_i \end{cases} .$$

In terms of the membership functions, we can write:

$$P(T_j = 0/\omega_i) = e_i\ M_j(i) \qquad (3.25)$$

$$P(T_j = 0/\omega_i) = 1 - e_i\ M_j(i) \ . \qquad (3.26)$$

Thus:

$$P(\omega_i/T_j = 1) \quad \frac{e_i \; M_j(i) \; P(\omega_i)}{\sum_{\ell=0}^{n} e_\ell \; M_j(\ell) \; P(\omega_\ell)} \tag{3.27}$$

It follows that:

$$P(\omega_0/T_j = 1) = 0 \quad . \tag{3.28}$$

If t_j does not detect f_i then:

$$P(\omega_i/T_j = 1, \; M_j(i) = 0) = 0 \tag{3.29}$$

If t_j detects f_i then:

$$P(\omega_i/T_j = 1, \; M_j(i) = 1) = \frac{e_i \; P(\omega_i)}{\sum_{\ell=0}^{n} e_\ell \; M_j(\ell) \; P(\omega_\ell)} \; . \tag{3.30}$$

Equations (3.28), (3.29) and (3.30) give the value of the posterior probabilities if the applied test t_j fails (results in an output different from that of a fault free circuit).

Similarly,

$$P(\omega_i/T_j = 0) = \frac{[1-e_i \; M_j(i)] \; P(\omega_i)}{\sum_{\ell=0}^{n} [1-e_\ell \; M_j(\ell)] \; P(\omega_\ell)} \tag{3.31}$$

Thus,

$$P(\omega_0/T_j = 0) = \frac{P(\omega_0)}{\sum_{\ell=0}^{n} [1-e_\ell M_j(\ell)] P(\omega_\ell)} \qquad (3.32)$$

Notice that:

$$\sum_{\ell=0}^{n} P(\omega_\ell) = 1$$

Thus,

$$\sum_{\ell=0}^{n} e_\ell M_j(\ell) P(\omega_\ell) < 1 , \qquad (3.33)$$

and,

$$\sum_{\ell=0}^{n} [1-e_\ell M_j(\ell)] P(\omega_\ell) < 1 \qquad (3.34)$$

From (3.32) and (3.34), it follows that:

$$P(\omega_0/T_j = 0) > P(\omega_0) \quad . \qquad (3.35)$$

If t_j does not detect f_i, then:

$$P(\omega_i/T_j = 0, M_j(i) = 0) > P(\omega_i) \qquad (3.36)$$

This is an interesting result, since it indicates that the certainty about the circuit having a particular fault increases (if the applied test does not detect that fault)

even though the applied test produced a good output (that is, identical with that of a fault free circuit).

If t_j detects f_i, $P(\omega_i/T_j = 1)$ could be less than, equal to, or even greater than $P(\omega_i)$, i.e.,

$$P(\omega_i/T_j = 0,\ M_j(i) = 1) \gtrless P(\omega_i) \tag{3.37}$$

This is even a more interesting result since the certainty about the circuit having a particular fault could increase even if the applied test detects that fault and produces a good output.

From (3.36) and (3.37), it is clear that a decision rule based on comparing the posterior probabilities with some thresholds, as was done in Subsection 3.2.1 for the simple case, is not acceptable since the posterior probabilities are not monotonically decreasing functions, thus there is no guarantee that the detection procedure will terminate in a finite amount of time.

The posterior probabilities after applying k+1 tests; the k+1-st being, say, test t_j; are (from (3.4)):

$$P(\omega_i/T_j,\ b\ ;\ b \in T^k) = \frac{P(T_j/\omega_i,b)\ P(\omega_i/b)}{\sum_{\ell=0}^{n} P(T_j/\omega_\ell,b)\ P(\omega_\ell/b)}\ .$$

Notice that $P(T_j/\omega_i,b) = P(T_j/\omega_i)$, thus:

$$P(\omega_i/T_j = 0,\ b\ ;\ b \in T^k) = \frac{[1-e_i\ M_j(i)]\ P(\omega_i/b)}{\sum_{\ell=0}^{n} P(T_j = 0/\omega_\ell)\ P(\omega_\ell/b)} \cdot \tag{3.38}$$

Definition 3.3

If $k+1$ tests from τ are applied; the $k+1$-st being, say, test t_j; and a good output was observed every time, the likelihood ratios $\lambda_{i,k+1}(1 \leq i \leq n)$ are defined as:

$$\lambda_{i,k+1} = \frac{P(\omega_i/T_j = 0,\ b = \vec{0}^k\ ;\ b \in T^k)}{P(\omega_0/T_j = 0,\ b = \vec{0}^k\ ;\ b \in T^k)} \cdot \tag{3.39}$$

From (3.38), it follows that:

$$\lambda_{i,k+1} = [1-e_i\ M_j(i)]\ \lambda_{i,k} \cdot \tag{3.40}$$

Thus
$$\lambda_{i,k+1} \leq \lambda_{i,k} \tag{3.41}$$

The equality in (3.41) holds only if the $k+1$-st test detects f_i, otherwise this relation is strict inequality.

Definition 3.4

The initial likelihood ratios λ_i $(1 \leq i \leq n)$ are defined as:

$$\lambda_i = \lambda_{i,0} = \frac{P(\omega_i)}{P(\omega_0)} \cdot \tag{3.42}$$

Theorem 3.2

If k tests from τ are applied, ℓ of them are from τ_i, and none of them failed, then:

$$\lambda_{i,k} = (1-e_i)^{\ell} \lambda_i \ . \qquad (3.43)$$

Proof

It is clear that $\ell \leq k$. Proof is by induction on k.

k = 1 : from (3.40),

$$\lambda_{i,1} = [1-e_i M_j(i)] \lambda_i ;$$

the first test being t_j.

ℓ could be 0 or 1:

If $\ell = 0$ ($t_j \notin \tau_i$), then $M_j(i) = 0$, i.e., $\lambda_{i,1} = \lambda_i$, which satisfies (3.43) for $\ell = 0$.

If $\ell = 1$ ($t_j \in \tau_i$), then $M_j(i) = 1$, i.e., $\lambda_{i,1} = (1-e_i)\lambda_i$, which satisfies (3.43) for $\ell = 1$.

Assume true for k: from (3.40),

$$\lambda_{i,k+1} = [1-e_i M_j(i)] \lambda_{i,k} ;$$

the k+1-st test being t_j.

If $t_j \notin \tau_i$, then $M_j(i) = 0$, thus,

$$\lambda_{i,k+1} = \lambda_{i,k} = (1-e_i)^{\ell} \lambda_i \ . \qquad (3.44)$$

$t_j \notin \tau_i$ also means that the number of tests from τ_i in the first k+1 tests is ℓ. Therefore, (3.44) indicates that the theorem is true for k+1.

If $t_j \in \tau_i$, then $M_j(i) = 1$, thus;

$$\lambda_{i,k+1} = (1-e_i)\ \lambda_{i,k} = (1-e_i)^{\ell+1}\ \lambda_i \ . \qquad (3.45)$$

$t_j \in \tau_i$ also means that the number of tests from τ_i in the first k+1 tests is $\ell+1$. Therefore, (3.45) indicates that the theorem is true for k+1.

Q.E.D.

Corollary 3.4

$$\lim_{\ell\to\infty} \lambda_{i,k} = 0 \qquad (3.46)$$

Proof:

$$(1-e_i) < 1 \ ,$$

therefore, $\lim_{\ell\to\infty} \lambda_{i,k} = \lim_{\ell\to\infty} (1-e_i)^{\ell}\lambda_i = 0$.

Q.E.D.

The Decision Rule

The decision rule suggested, compares the likelihood ratios λ_i, $(1 \leq i \leq n)$, with thresholds u_i ($u_i > 0$ for all i) as follows:

If $\lambda_i \leq u_i$ for all i, decide that the circuit is fault free and stop.

If $\lambda_i > u_i$ for some i, select a test from τ_i and repeat, (i.e., go to step (1) of the detection procedure).

From (3.41) we see that the likelihood ratios are monotonically non-increasing functions. Any likelihood

ratio $\lambda_{i,k}$ will strictly decrease if we apply a test from τ_i (and of course, that test produces a good output). Thus, from (3.46), the likelihood ratios could go below the specified thresholds, and the detection procedure is guaranteed to terminate in a finite amount of time using this decision rule. Hence, it is an acceptable rule.

Theorem 3.2 can be used to determine a least upper bound on the number of tests k_i from τ_i ($1 \leq i \leq n$) that are needed, as follows:

$$(1-e_i)^{k_i} \lambda_i < u_i$$

$$(1-e_i)^{k_i} < \frac{u_i}{\lambda_i} \quad .$$

$$k_i \log(1-e_i) < \log \frac{u_i}{\lambda_i}$$

or,
$$k_i > \frac{\log \frac{u_i}{\lambda_i}}{\log(1-e_i)}$$

or,
$$k_i > \frac{\log \frac{u_i\, p_0}{p_i}}{\log(1-e_i)} \quad . \qquad (3.47)$$

3.3 OPTIMUM DETECTION EXPERIMENT

In Section 3.2.2, least upper bounds k_i's on the number of tests that detect f_i that are needed were obtained (3.47). However, a given test usually detects more than one fault; so, a method is needed to determine

how many times every test should be repeated so that (3.47) is satisfied for all i ($1 \leq i \leq n$), and that the overall experiment length is minimum. If t_j ($1 \leq j \leq m$) is repeated x_j times, then, (3.47), in terms of the fault table matrix A (definition 2.1), yields,

$$\sum_{j=1}^{m} a_{ij} x_j \geq k_i \ . \tag{3.48}$$

The experiment length ℓ is:

$$\ell = \sum_{j=1}^{m} x_j \tag{3.49}$$

ℓ is the function to be minimized.

The quantities at hand satisfy the following conditions:

1. $x_j \geq 0$
2. all x_j's are integers

This is an all integer-integer programming problem, the solution of which determines the optimum number of repetitions for each test. For details and solutions of integer programming problems, see [19, 20].

It should be noted that for the permanent faults case, every k_i ($1 \leq i \leq n$) will be equal to 1; thus, similar to the problem of minimizing boolean functions. Integer

programming treatments for this special case are found in [9, 10, 19].

Example 3.2

Consider the following fault table matrix:

	t_1	t_2	t_3	t_4	t_5
f_1	1	1	0	1	0
f_2	1	1	1	0	1
f_3	0	1	1	1	1
f_4	0	1	0	0	1
f_5	0	0	0	1	1
f_6	1	0	1	0	0

The corresponding integer programming problem is:

$$
\begin{aligned}
x_1 + x_2 \quad\quad + x_4 \quad\quad &\geq k_1 \\
x_1 + x_2 + x_3 \quad\quad + x_5 &\geq k_2 \\
x_2 + x_3 + x_4 + x_5 &\geq k_3 \\
x_2 \quad\quad\quad + x_5 &\geq k_4 \\
x_4 + x_5 &\geq k_5 \\
x_1 \quad + x_3 \quad\quad &\geq k_6
\end{aligned}
$$

Find integer x_j's that minimize:

$$x_1 + x_2 + x_3 + x_4 + x_5.$$

3.3.1 A Suboptimal Solution

If the values of the k_j's are relatively large (e.g., > 10), the integer programming problem presented by

(3.48) and (3.49) can be solved as a linear programming problem (actually as a transportation problem since the coefficients are 0's and 1's). The solutions are then rounded up (the smallest integers greater than or equal to the obtained solutions are used). This is generally a faster solution since solving a linear programming problem is easier than solving an integer programming one. Linear programming methods will result in a very little deviation from the optimal solution if the values of k_j's are large enough, since the function to be minimized is just the sum of the x_j's.

3.3.2 Reduction of the Fault Table Matrix A:

The size of the matrix A can easily get to be huge for a large size circuit. The amount of memory and number of computations needed to solve an integer programming problem (or a linear programming one) can be greatly reduced if the size of matrix A is reduced. In this subsection, an attempt is made to transform the problem into an equivalent one, but with a smaller matrix A*. As A is transformed into A*, the values of k_j's will be adjusted. The obtained solution, i.e., x_j's, for the reduced problem will also be adjusted by adding appropriate biases b_j's to obtain the solution for the original problem. The initial bias values are zeros.

The following operations are suggested for the reduction of the fault table matrix A to A*:

(a) If only one 1 occurs in row i, say, $a_{ij}=1$ and $a_{ik}=0$ for $k \neq j$, delete row i from the matrix, increment, b_j by k_i, and, for every row q for which $a_{qj}=1$, replace k_q by k_q-k_i, if this difference is zero or negative, delete row q.

(b) If row i contains 1's wherever row j does ($i \neq j$), that is, for each k, $a_{jk} = 1$ implies $a_{ik}=1$; (row i dominates row j) and if $k_j \geq k_i$, delete row i.

(c) If column i contains 1's wherever column j does ($i \neq j$), that is, for each k, $a_{kj}=1$ implies $a_{ki}=1$, (column i dominates column j) then delete column j.

These operations may be carried out in any order starting with A and continuing until a matrix A* is obtained on which none of them can be applied. The solution for the reduced problem (with A*) is then obtained. Every x_j in the solution of the original problem is obtained by adding the bias b_j to the corresponding x_j obtained in the solution with A*.

Theorem 3.3

If A* is the matrix obtained from the fault table matrix A by a succession of operations of the (a), (b) and (c) types suggested above, then the solution to the original problem (presented by (3.48) and (3.49) is the same as that of the reduced problem adjusted by the biases b_j's ($1 \leq j \leq m$).

Proof

It is sufficient to prove that any operation will result in a problem with an equivalent solution.

(1) Assume operation (a) is performed. This means that one of the constraints in (3.48) is

$$x_j \geq k_i \ ,$$

i.e., test t_j has to be repeated at least k_i times. If we adjust the corresponding bias b_j by k_i, this constraint can be removed (i.e., delete row i) since the bias will make sure that it is satisfied.

Every constraint of the form:

$$\ldots + x_j + \ldots \qquad \geq k_q \ ,$$

can be written as

$$\ldots + (x_j - k_i) + \ldots \quad \geq k_q - k_i \ .$$

Let $x_j' = x_j - k_i$. The set of constraints can now be written in terms of x_j' (the reduced problem). If $k_q - k_i$ is zero or negative, then this constraint is automatically satisfied by $x_j > k_i$ thus row q can be eliminated. If $k_i < k_q$ we solve the reduced problem. Obviously, a solution to the reduced problem is a solution to the original problem if we adjust the obtained value for x_j' by k_i, i.e., by the amount of bias adjustment. Thus, operation (a) results in an equivalent solution.

(2) Assume operation (b) is performed. Let the number of 1-entries in rows i and j be q_i and q_j respectively. Without loss of generality, assume that the 1-entries in row i are the first q_i entries in that row. Since the condition for operation (b) is satisfied, we can also assume that the 1-entries of row j are the first q_j entries in that row. It is clear that $q_i \geq q_j$. The following constraints must now exist:

$$x_1 + x_2 + \ldots + x_{q_j} + \ldots + x_{q_i} \geq k_i$$
$$x_1 + x_2 + \ldots + x_{q_j} \geq k_j \ .$$

The second constraint implies:

$$x_1 + x_2 + \ldots + x_{q_j} + \ldots + x_{q_i} \geq k_j$$

If $k_j \geq k_i$, then the first constraint is automatically satisfied by satisfying the second. Thus, elimination of row i will not change the solution.

(3) Assume operation (c) is performed. In this case, it is clear that $x_j=0$ does not result in any contradiction to the constraints of (3.48), since for every x_j appearance, x_i also appears; so, x_i can be set large enough to satisfy the constraint. We need to prove that $x_j=0$ is part of a solution that minimizes : $\sum_{q=1}^{m} x_q$. Assume

that a solution minimizing this sum exists with $x_j > 0$. From this solution, substitute $x_i + x_j$ for x_i and 0 for x_j in (3.48). No contradiction results since column i dominates column j. The same substitution will result in the same sum $\sum_{q=1}^{m} x_q$, i.e., minimality is maintained. Thus we found a solution with $x_j=0$. Thus column j can be eliminated.

Q.E.D.

CHAPTER IV

DIAGNOSIS OF INTERMITTENT FAULTS IN COMBINATIONAL CIRCUITS

In Chapter III, a probabilistic model for intermittent faults was introduced, also an approach for the detection of these faults. The detection procedure proposed relied on the repeated application of tests that would detect these faults had their effect been permanent. Several methods for test generation were discussed in Chapter II. In this chapter, we present an approach for the diagnosis of intermittent faults in combinational circuits, also, employing the repetition of tests that detect permanent faults.

4.1 GENERAL ASSUMPTIONS

There are three basic assumptions for the diagnosis methodology to be presented, namely:

(1) The probabilistic model, introduced in Section 3.1 for intermittent faults, will be used here.

All the assumptions previously made in the model are also carried here. Of major importance is the single fault assumption.

(2) A detection experiment, similar to that proposed in Section 3.2, is assumed to have been run and resulted in the decision that the circuit has an intermittent fault, i.e., a test in that experiment has failed. This assumption assures that a fault exists in the circuit before we start the diagnosis experiment.

(3) The posterior probabilities of the states of the circuit at the end of the detection experiment are assumed to be known to the experimenter. These will be used as prior probabilities in the diagnosis experiment.

A fourth assumption, that is helpful even though not essential, will also be assumed:

(4) The test that failed in the detection experiment, say test t_j, is assumed to be known to the experiment. This assumption tends to reduce the length of the diagnosis experiment since it will start with less possible faults (those faults that correspond to the 1-entries of column j of the fault table) than the total fault set.

4.2 DIAGNOSIS OF INTERMITTENT FAULTS

The approach suggested for the diagnosis of intermittent faults is through the repeated application of tests from the test set τ. A subset of τ is selected and its tests are repeatedly applied until a failure occurs. This narrows down the possible faults that the circuit might have. The fault table is then reduced and another subset of τ is selected and the process is repeated until enough failures occur to diagnose the fault in the circuit.

Let the set of possible intermittent faults, at any stage in the experiment, be Ω_p. Let F_p be the subset of F that corresponds to Ω_p. Let Ω_p (also F_p) contain η elements. Initially F_p contains the faults that have 1-entries in column j of the fault table (assuming that t_j was the test that failed in the detection experiment). Obviously, Ω_p does not contain ω_0 since in the diagnosis phase, we know that the circuit is faulty for sure. The posterior probabilities of the states of the circuit at the end of the detection experiment are used as prior probabilities for the diagnosis experiment.

Fault Table Reduction. Whenever a failure occurs during the diagnosis experiment, F_p is narrowed down, and consequently the fault table is reduced. The following reduction steps are applied until none of them can be applied any more:

(a) Eliminate all rows that correspond to faults not contained in F_p.

(b) Eliminate all redundant columns, i.e., eliminate all but one of every group of identical columns.

(c) Eliminate all 0-columns.

(d) Eliminate all the 1-columns. The failure of a test, whose corresponding column in the fault table has all 1-entries, does not contribute any information to the diagnosis of the existing fault. Thus this test is eliminated.

Diagnosis Procedure. The following diagnosis procedure is now suggested (a flow chart for this procedure is given in Figure 4.1):

(1) From the outcome of the detection experiment, compute F_p , Ω_p. Each of these sets has η elements.

(2) If F_p is a singleton (i.e., if $\eta = 1$) go to (9), otherwise go to (3).

(3) Obtain a reduced fault table using steps (a) through (d) indicated above.

(4) Select a subset τ_p of τ that covers F_p, i.e., τ_p is a detection test set for F_p. Let τ_p contain μ tests. Without loss of generality, we can assume that the tests in τ_p are ordered, so we can speak of the i-th test in τ_p ($1 \leq i \leq \mu$). If no such τ_p is found, go to (9).

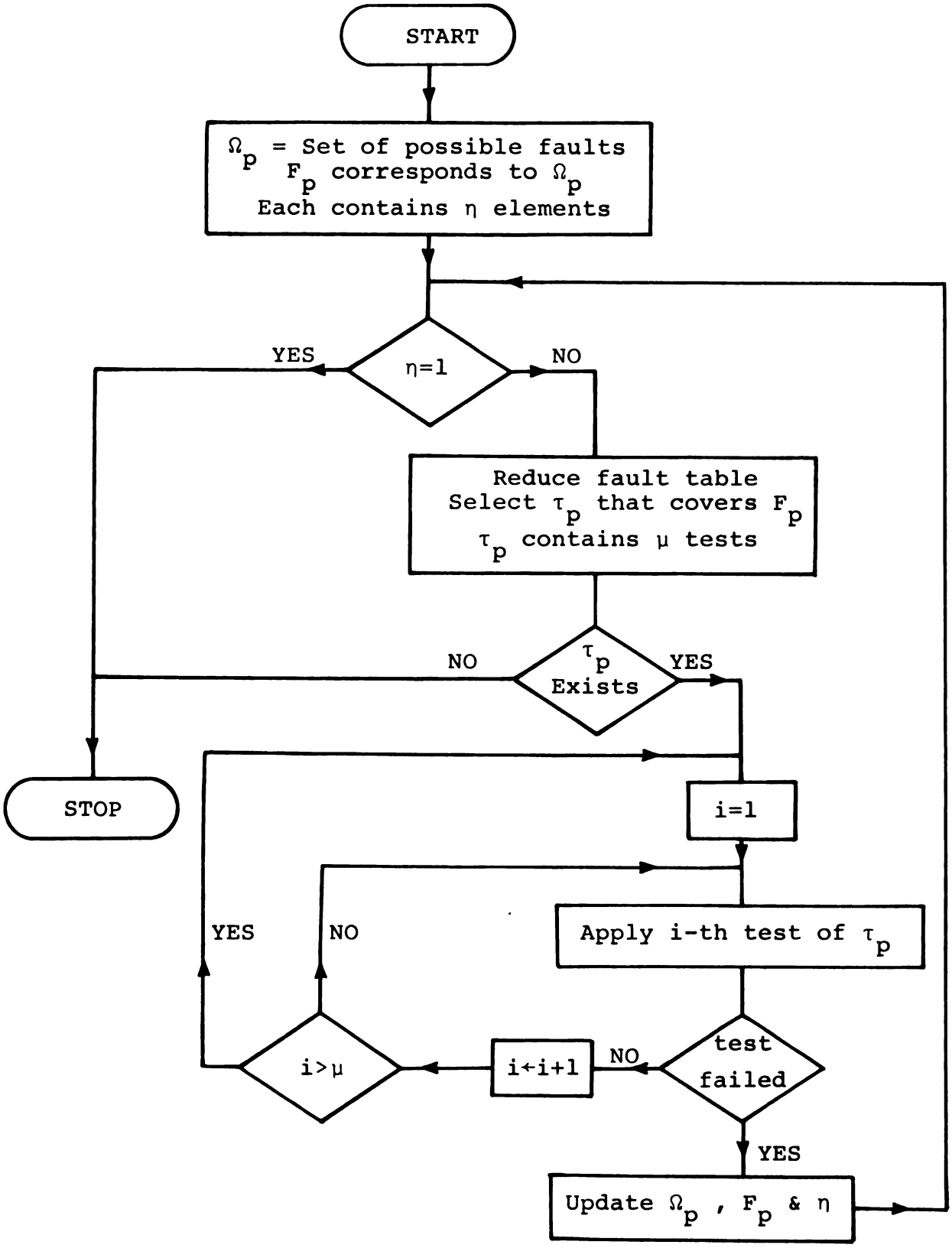


Figure 4.1. Flow Chart for the Diagnosis Procedure.

(5) $i = 1$.

(6) Apply the i-th test of τ_p. If this test fails, go to (8), otherwise go to (7).

(7) Increment i by 1. If it exceeds μ go to (5), otherwise go to (6).

(8) At this point, a test in τ_p has failed, this narrows down the set of possible faults Ω_p (also F_p) to a smaller set. This smaller set is the one that corresponds to faults in F_p with 1-entries in the column corresponding to the failing test in the fault table. Replace F_p and Ω_p by the smaller sets indicated above. Update η. Go to (2).

(9) Stop. Diagnosis experiment is complete. Diagnostic resolution is determined by F_p. If F_p is a singleton, complete diagnosis is obtained.

Notice that when selecting the test set τ_p, no test that failed earlier in the diagnosis experiment will be chosen, since such a test corresponds to a column, all of its entries are 1's, in the reduced fault table. Such a column will be eliminated by operation (d) of the fault table reduction procedure.

The diagnosis procedure terminates only if one of two conditions arises:

(1) F_p becomes a singleton; in which case complete diagnosis is obtained, or

(2) no τ_p that covers F_p is found; in which case complete diagnosis is not obtained, the diagnosis resolution being determined by F_p.

Diagnosis Tree. The diagnosis procedure can be represented by a tree. The nodes correspond to the F_p's, with the root being the initial F_p obtained from the detection experiment. The edges out of a node correspond to the tests of the appropriate τ_p, i.e., every node has μ (the corresponding μ) edges out of it. An edge, corresponding to a test t_j out of a node α, goes to node β that corresponds to the subset of the faults of node α that are detected by t_j. At any stage in the diagnosis procedure, if a different τ_p is chosen, a different diagnosis experiment and consequently a different diagnosis tree will result.

A complete diagnosis tree is a diagnosis tree that contains all the subtrees corresponding to all the possible outcomes of the diagnosis experiments. The leaves correspond to the maximum diagnostic resolution possibly obtained.

Definition 4.1

The portion of the diagnosis procedure that consists of applying the tests of τ_p repeatedly until a failure occurs is called a subexperiment.

In terms of the diagnosis procedure, a subexperiment consists of iterations of the loop defined by steps (5),

(6) and (7) until a failure occurs to exit from this loop. Notice that the failure of a test is the only way to exit from this loop. In order for the diagnosis experiment to be finite, the expected length of every subexperiment in it should be finite. The condition that τ_p covers F_p guarantees that, this will be proved later in Section 4.3.

During the diagnosis procedure, it is desirable to calculate the posterior probabilities of the different states of the circuit after observing the outcome of every test applied. These posterior probabilities can be employed to select a τ_p, at the beginning of a subexperiment, that tends to make the diagnosis experiment shorter. It should be noted that the posterior probabilities for any ω_i that is not in Ω_p is zero. This follows directly from (3.29) since such an ω_i corresponds to a permanent fault f_i that is not detected by a test that failed earlier.

Example 4.1

A detection experiment was run. It resulted in the decision that the circuit has an intermittent fault. Observing the failing test ruled out some possibilities for the fault condition. Fault table reduction, as suggested above, resulted in the reduced fault table given in Table 4.1. A diagnosis experiment is to be run.

	t_1	t_2	t_3	t_4	t_5	t_6	t_7
f_1	0	0	0	1	1	0	1
f_2	0	1	0	1	0	1	0
f_3	0	1	0	0	1	1	1
f_4	1	0	1	0	1	0	0
f_5	1	1	1	0	0	0	1
f_6	0	1	0	0	0	1	1
f_7	0	0	1	1	0	1	1
f_8	1	1	0	0	0	1	0

Table 4.1. Reduced Fault Table After Detection Experiment.

The initial Ω_p and F_p are:

$$\Omega_p = \{\omega_1,\omega_2,\ldots,\omega_8\}$$
$$F_p = \{f_1,f_2,\ldots,f_8\}$$

Notice that ω_0 and f_0 are not included in Ω_p or F_p. The test set $\{t_1,t_2,t_7\}$ covers F_p; select it as τ_p. Apply $t_1,t_2,t_7,t_1,t_2,t_7,\ldots$ until a failure occurs. Assume that this subexperiment results in the failure of t_7. The set of possible faults is narrowed down to those faults with 1's in column t_7 of Table 4.1 (ω_2,ω_4, and ω_8 are ruled out). Thus:

$$\Omega_p = \{\omega_1,\omega_3,\omega_5,\omega_6,\omega_7\}.$$
$$F_p = \{f_1,f_3,f_5,f_6,f_7\}.$$

The corresponding reduced fault table is shown in Table 4.2.

Select a new τ_p that covers F_p. The test set $\{t_2,t_5,t_6\}$ is selected. Apply $t_2,t_5,t_6,t_2,t_5,t_6,\ldots$ until a failure occurs. Assume that t_2 failed in this subexperiment. This rules out ω_1 and ω_7. Thus, we have:

$$\Omega_p = \{\omega_3,\omega_5,\omega_6\}$$
$$F_p = \{f_3,f_5,f_5\}$$

Reducing Table 4.2 in correspondence with the outcome of this subexperiment, we obtain Table 4.3. The test set $\{t_1,t_6\}$ is a suitable τ_p since it covers F_p. Apply $t_1,t_6,t_1,t_6,\ldots$ until a failure occurs. Say t_1 fails. This rules out ω_3 and ω_6. The corresponding F_p and Ω_p are:

	t_1	t_2	t_3	t_4	t_5	t_6
f_1	0	0	0	1	1	0
f_3	0	1	0	0	1	1
f_5	1	1	1	0	0	0
f_6	0	1	0	0	0	1
f_7	0	0	1	1	0	1

Table 4.2. Reduced Fault Table After First Subexperiment.

	t_1	t_5	t_6
f_3	0	1	1
f_5	1	0	0
f_6	0	0	1

Table 4.3. Reduced Fault Table After Second Subexperiment.

$$\Omega_p = \{\omega_5\}$$
$$F_p = \{f_5\}$$

F_p is a singleton. This terminates the diagnosis experiment. The circuit is diagnosed as having intermittent fault ω_5.

If test t_6 is the one that failed in this subexperiment and not t_1, then ω_5 is ruled out. The corresponding F_p and Ω_p are:

$$\Omega_p = \{\omega_3, \omega_6\}$$
$$F_p = \{f_3, f_6\}$$

The corresponding reduced fault table is given in Table 4.4. In this case there is no τ_p that covers both f_3 and f_6. This terminates the diagnosis experiment with the diagnosis resolution being defined by F_p which contains f_3 and f_6.

	t_5
f_3	1
f_6	0

Table 4.4. Reduced Fault Table when t_6 Fails in Third Subexperiment.

That is, the result of the diagnosis experiment is: the circuit has either intermittent fault ω_3 or intermittent fault ω_6. Thus, the components that pertain to these two faults should be replaced for repair. A diagnosis tree for this example is shown in Figure 4.2. The tree shown is not a complete diagnosis tree.

In this example, we did not make use of the posterior probabilities since we were not concerned about comparing τ_p's or designing a shortest diagnosis experiment.

The last result of Example 4.1 should be compared with the diagnosis of permanent faults. If the faults of this example were permanent, maximum diagnostic is possible since the rows of the fault table are distinguishable (Theorem 2.2). However, in case of intermittent faults, and using the suggested diagnosis procedure, we were unable to obtain complete diagnosis. Hence, there are fundamental

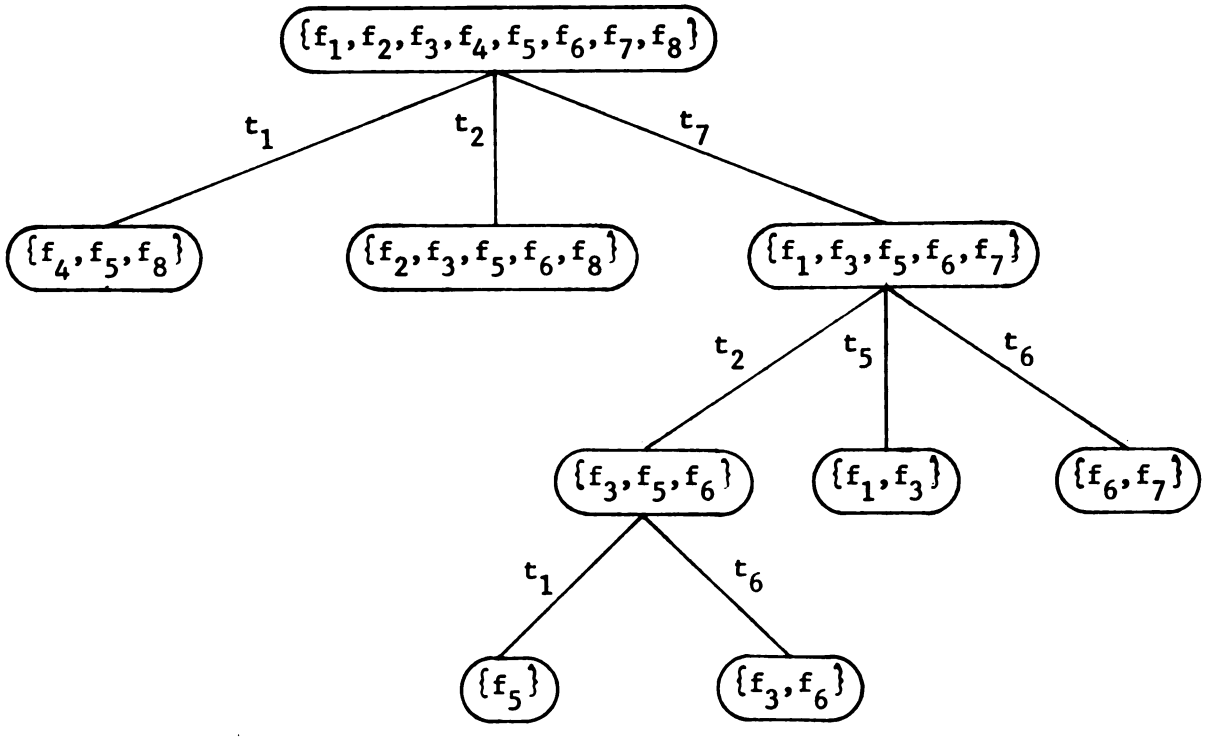


Figure 4.2. Diagnosis Tree for Example 4.1.

differences between the diagnosis of intermittent faults and the diagnosis of permanent ones that are more subtle than mere repetition of tests. Fault table properties that are relevant to the diagnosis of intermittent faults are explored later in Section 4.4.

4.3 EXPECTED LENGTH OF SUBEXPERIMENT

In this section, we try to compute the expected length of a subexperiment (definition 4.1) of the diagnosis procedure. In a subexperiment, tests of an appropriate τ_p are repeatedly applied in sequence until a failure occurs. τ_p contains μ tests. Let the set of corresponding random variables be T_p. The set of possible faults Ω_p contains η states. Without loss of generality, we can label the elements of τ_p and T_p as: $1,2,\ldots,\mu$, and the elements of Ω_p as: $1,2,\ldots,\eta$. We assume that the posterior probabilities for the states of the circuit were computed after observing the outcome of every test in the diagnosis experiment. The posterior probabilities computed thus far (to the beginning of the subexperiment) will be used as prior probabilities in the subexperiment. The problem is to find the expected number of times the tests of τ_p will be applied until the first failure.

Notation. The following notation will be used in this section:

$T_p = \vec{0}^{\mu}$ will mean $T_1=0$, $T_2=0,\ldots,$ and $T_\mu=0$

This corresponds to the case where every test in τ_p has been applied once and one of them failed.

$T_p \neq \vec{0}^{\mu}$ will mean the complement of the above

condition. That is, every test of τ_p was applied once and at least one of them failed.

$T_p^k = 0^{\vec{\mu k}}$ will mean $T_p=0^{\vec{\mu}}$, $T_p=0^{\vec{\mu}}$,... k times.

This corresponds to the case where the test set τ_p was applied k times and no failure occurred.

The ω_i's $(1 \leq i \leq \eta)$ define disjoint events over the probability space $(\Omega \times S)^k$ (the cartesian product of $(\Omega \times S)$ by itself k times). The probabilities of these events (the prior probabilities of the subexperiment) add to 1 due to the single fault assumption. Thus we can write:

$$P(T_p=0^{\vec{\mu}}) = P(T_1=T_2=\ldots=T_\mu=0)$$

$$= \sum_{i=1}^{\eta} P(T_p=0^{\vec{\mu}}/\omega_i)\ P(\omega_i) \qquad (4.1)$$

$$P(\omega_i) = p_i \qquad (4.2)$$

If ω_i is covered k_i times by τ_p (i.e., k_i tests of τ_p detect f_i), then:

$$P(T_p=0^{\vec{\mu}}/\omega_i) = (1-e_i)^{k_i} \qquad (4.3)$$

The conditional probability of at least one failure during the application of τ_p is:

$$P(T_p\neq 0^{\vec{\mu}}/\omega_i) = 1 - (1-e_i)^{k_i} \qquad (4.4)$$

If τ_p is applied k times and no test failed, then:

$$P(T_p^k = 0^{\vec{\mu k}}/\omega_i) = (1-e_i)^{k_i k} \tag{4.5}$$

The probability of a first failure on the k+1-st application of τ_p is:

$$P(T_p^{k+1} ; T_p^{k+1} = T_p^k T_p , T_p^k = 0^{\vec{\mu k}} , T_p \neq 0^{\vec{\mu}})$$

$$= \sum_{i=1}^{\eta} P(T_p^{k+1} ; T_p^{k+1} = T_p^k T_p, T_p^k = 0^{\vec{\mu k}}, T_p \neq 0^{\vec{\mu}}/\omega_0) \; p_i$$

$$= \sum_{i=1}^{\eta} (1-e_i)^{k_i k} \left(1 - (1-e_i)^{k_i}\right) p_i \tag{4.6}$$

This is the probability that the subexperiment will have k+1 applications of τ_p.

Thus, the expected number of applications of τ_p is:†

$$\bar{k} = E(k)$$

$$= \sum_{k=0}^{\infty} (k+1) \sum_{i=1}^{\eta} (1-e_i)^{k_i k} \left(1-(1-e_i)^{k_i}\right) p_i$$

$$= \sum_{i=1}^{\eta} p_i \sum_{k=0}^{\infty} (1-e_i)^{k_i k} \left(1-(1-e_i)^{k_i}\right) (k+1)$$

---

†E(k) denotes the expected value of k.

The inner infinite series is the expansion of a negative binomial,* thus

$$\bar{k} = \sum_{i=1}^{\eta} \frac{p_i}{1-(1-e_i)^{k_i}} \tag{4.7}$$

The expected subexperiment length $\bar{\ell}$ is:

$$\bar{\ell} = \mu\bar{k} \tag{4.8}$$

The expected subexperiment length given by (4.8) is approximate since the last application of τ_p (when a failure occurs) will probably be incomplete. However, this is fairly accurate if $\bar{k}$ is relatively large.
The conditional expectation of the number of applications of τ_p is given by:

$$E(k/\omega_i) = \frac{1}{1-(1-e_i)^{k_i}} \tag{4.9}$$

From (4.7) and (4.8), if $k_i > 0$, then $\bar{\ell}$ is finite, and if $k_i = 0$, for some $i(1 \leq i \leq \eta)$, then $\bar{\ell} = \infty$. This explains why τ_p was required to cover Ω_p in the diagnosis procedure: to insure that the diagnosis procedure will eventually terminate.

---

*If $0<x<1$, then $(1+x)^{-2} = \sum_{i=0}^{\infty} (1+i)x^i$.

4.4 FAULT TABLE

In this section, we study some fault table properties that are pertinent to the diagnosis of intermittent faults. Our discussion will deal only with the fault table and not with the complete fault table, i.e., $\vec{f}_0$ will be excluded. The number of rows is assumed to be n, and the number of columns is assumed to be m.

Definition 4.2

A fault table will be called locally symmetric if for every pair of rows, f_i and f_j, there is a pair of columns, t_p and t_q, whose entries for these two rows have one of the following forms:

	t_p	t_q
f_i	1	0
f_j	0	1

or

	t_p	t_q
f_i	0	1
f_j	1	0

Theorem 4.1

The local symmetry property is not affected by the fault table reduction operations (operations (a), (b), (c) and (d) of Section 4.2) of the diagnosis procedure.

Proof:

It sufficies to show that the application of any of these operations maintains the local symmetry property in the reduced table.

(a) The elimination of a row, obviously, does not change the local symmetry for the rest of the rows as long as two or more rows are left. In the diagnosis procedure, table reduction is performed only if the new F_p is not a singleton, i.e., the fault table will have at least two rows after the reduction.

(b) The elimination of a redundant column, clearly, maintains the property for the remaining rows.

(c) The elimination of an all zeros column does not change the property since it does not contribute anything to it.

(d) Similarly, the elimination of an all ones column does not change the property since it does not contribute anything to it.

Q.E.D.

Corollary 4.1

If the original fault table is locally symmetric, then any row in a reduced fault table, during the course of the diagnosis experiment, has at least one 1-entry. This follows directly from Theorem 4.1.

Theorem 4.2

A necessary and sufficient condition for maximum diagnostic resolution (i.e., every fault is diagnosable), using the diagnosis procedure of Section 4.2, is that the fault table be locally symmetric.

Proof:

Sufficiency. If the fault table is locally symmetric, then by Theorem 4.1, this property will be maintained in the reduced fault tables. By Corollary 4.1, every row in a reduced fault table will contain at least one 1-entry. Thus, we can always find a τ_p that covers F_p. Thus the only way the diagnosis procedure can terminate is when F_p is a singleton. That is, we have maximum diagnostic resolution.

Necessity. Assume that every fault is diagnosable.

Let ω_i be the fault that the circuit possesses. Thus, during the diagnosis experiment, a row corresponding to f_i will always exist in the reduced fault table since ω_i will never be ruled out (i.e., no test that does not detect f_i will fail). Diagnosis is accomplished through observing enough failures to narrow down the possibilities of the fault condition as much as possible. Since ω_i is diagnosable, then considering the fault ω_j, a test that detects f_i and does not detect f_j must eventually fail in the experiment (in order to rule out ω_j). That is, there must exist a test, t_p, whose entries for f_i and f_j are of the form:

	t_p
f_i	1
f_j	0

Let ω_j be the fault that the circuit possesses. Since every fault is diagnosable, then by a similar argument, there must exist a test, say t_q, whose entries for f_i and f_j are of the form:

	t_q
f_i	0
f_j	1

That is, there exists a pair of tests, say t_p and t_q, whose entries for f_i and f_j are in the one of the forms:

	t_q	t_p
f_i	0	1
f_j	1	0

or

	t_p	t_q
f_i	1	0
f_j	0	1

The argument can be repeated for any other pair of faults. It follows that maximum diagnostic resolution implies the local symmetry property.

Q.E.D.

Definition 4.3

The complement of an n x m fault table is another n x m fault table with the 1-entries of the original table replaced by 0's and vice versa for the 0-entries.

Theorem 4.3

If a fault table is locally symmetric, then its complement is.

The proof follows directly from the definition of the local symmetry property.

Theorem 4.4

A fault table that has no identical rows and that is its own complement (with the columns rearranged) is locally symmetric.

Proof:

Since the rows are distinguishable, then for every pair of rows, say $\vec{f}_i$ and $\vec{f}_j$, there is a test that has different entries for these rows. Let the entries of such a test for $\vec{f}_i$ be 1 and for $\vec{f}_j$ be 0. Since the table is its own complement, then there is a test whose entries are the complement of the entries of the above test, i.e., has entries of 0 for $\vec{f}_i$ and 1 for $\vec{f}_j$. That is, the table is locally symmetric.

Q.E.D.

Corollary 4.2

An n x m fault table with no identical rows will become a locally symmetric n x 2m fault table if its complement is appended to it.

The proof follows directly from Theorem 4.4.

Corollary 4.2 and Theorem 2.3 indicate that the least number of tests m that satisfies the local symmetry

property for n faults is greater than $\log_2 n$ and less than or equal to $2\lceil \log_2 n \rceil$.†

4.5 OPTIMIZATION OF DIAGNOSIS EXPERIMENT

In this section we discuss the problem of optimizing the diagnosis experiment. The diagnosis procedure suggested in Section 4.2 is adaptive in nature, namely, the subexperiment to be adopted at any stage depends on which test failed in the previous subexperiment. The length of a particular subexperiment is not deterministic, i.e., it varies when the same subexperiment is run over again under the same circuit conditions, due to the intermittency of the faults. The key to an optimum experiment lies in the choice of an appropriate τ_p for every subexperiment. The usual aim of optimization is to minimize the expected length of the subexperiment. Three approaches that tend to minimize the length of the diagnosis experiment are presented: exhaustive enumeration, local optimization and the method of maximum resolution. The latter two approaches are heuristic in nature.

4.5.1 Exhaustive Enumeration

In this method, all possible complete diagnosis trees are constructed. The expected lengths of the

---

† $\lceil x \rceil$ means the smallest integer greater than or equal to x.

experiments corresponding to these trees are computed. The one with the shortest expected length is the optimal experiment. This method guarantees an optimum solution, but is not feasible even for small problems. This is due to the fact that the number of possible trees becomes unwieldy since at every subexperiment we have to construct trees corresponding to every possible τ_p. The number of trees to be enumerated is much larger than the case of permanent faults (which, by itself, is impossibly large as indicated in Subsection 2.6.1). The reasons are that the trees we have here are not binary, and that the tests of a subexperiment do not divide the set of possible faults into disjoint subsets, due to the fact that a possible fault may be detected by more than one test in the subexperiment. This latter reason also adds complexity to estimating the expected length of an experiment since it amounts to more than one leaf, in the complete diagnosis tree, having the same label.

4.5.2 Local Optimization

This method attempts to minimize the length of the diagnosis experiment by selecting the best (according to some criterion) τ_p for every subexperiment during the course of the diagnosis procedure. The criterion suggested here is one that takes into effect the expected length of the subexperiment, as well as the resolution that might result at the end of such subexperiment.

Definition 4.4

Let the μ fault subsets that are fault possibilities corresponding to the failures of the μ tests of τ_p be $F_{p_1}, F_{p_2}, \ldots, F_{p_\mu}$. Let the number of elements in F_{p_i} $(1 \leq i \leq \mu)$ be η_i. The resolution figure of merit r for this subexperiment is defined as:

$$r = \sum_{i=1}^{\mu} \binom{\eta_i}{2} \tag{4.10}$$

$\binom{\eta_i}{2}$ is the number of pairs of faults that we have to distinguish if the i-th test of τ_p fails in this subexperiment.

A figure of merit W, that is a function of $\bar{\ell}$ (the expected length of the corresponding subexperiment as defined by (4.8)) and r, is computed for every τ_p. The τ_p with the lowest W is selected for the subexperiment. We suggest the following simple function for W:

$$W = \bar{\ell}.r \tag{4.11}$$

Any other suitable function of W, in terms of $\bar{\ell}$ and r, may be chosen as long as it is monotonically increasing in $\bar{\ell}$ and also in r.

This method requires the enumeration of all possible subexperiments rather than all possible experiments as in

the previous method. However, it does not guarantee an optimum result.

4.5.3 The Method of Maximum Resolution

The local optimization method requires the enumeration of all possible subexperiments, a task that could be quite lengthy especially in the earlier stages of the experiment. In this section, we suggest a method not as good as local optimization but one that does not require the enumeration of all possible subexperiments. The idea is similar to local optimization. However, we select the figure of merit to be r itself and try to minimize it. The problem of selecting the τ_p with the lowest r is similar to the covering problem of switching theory with costs assigned to the prime implicants. The prime implicants are analogous to the tests of the reduced fault table and the minterms to be covered are analogous to the faults of the reduced fault table. The cost of a prime implicant being $\binom{n_i}{2}$. n_i is the number of 1's in the column of the corresponding test.

4.6 SPECIAL CASES

In this section we will discuss the diagnosis problem for special cases of the fault table, namely, the simple fault table and the simple elimination fault table.

4.6.1 The Simple Fault Table Case

A fault table is called simple, if every column has exactly one 1-entry and every row has exactly one 1-entry. It follows that the number of rows is equal to the number of columns and the fault table matrix is the identity matrix with some of its columns permuted. This special fault table corresponds to the case where every test detects one and only one fault.

If the original fault table is simple, then there is no need for a diagnosis experiment. The test that fails in the detection experiment identifies the fault that the circuit possesses.

If one of the reduced fault tables is simple, then τ_p of the corresponding subexperiment will contain all the tests of this table. This will be the last subexperiment in the diagnosis procedure since it will result in fault identification.

4.6.2 The Simple Elimination Fault Table

In a simple elimination fault table every column has exactly one 0-entry and every row has exactly one 0-entry. It follows that the fault table matrix is a square matrix. This table corresponds to the case where every test detects all but one of the faults. Without loss of generality we can assume the tests are labeled such that

test t_i does not detect fault f_i, i.e., the fault table matrix is the complement of the identity matrix.

In this case any two tests will suffice as a τ_p since any two tests cover all the faults. It is easy to see that after the first subexperiment the reduced fault table will also be a simple elimination fault table with the number of rows (or columns) being one less than that of the previous subexperiment.

If we are to minimize the diagnosis experiment for this case using the method of maximum resolution, we notice that selecting any two tests will result in the same value for the resolution figure of merit r. Notice that selecting more than two tests will result in a larger r. Thus, any two tests minimize r.

CHAPTER V

SUMMARY, CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK

5.1 THESIS SUMMARY

Digital computers are being relied upon as integral parts of an increasing number of systems handling all aspects of our life. The proper operation of computers is vital to the functioning of these computerized systems. One of the major approaches to the proper operation of computers is fault diagnosis plus repair. This thesis lends itself to one aspect of this approach, namely: diagnosis of intermittent faults in combinational circuits.

Chapter I presents a general discussion of fault-tolerant computing together with some of the physical failures that can occur in digital circuits. The logical representations of many of these failures are also presented.

Chapter II deals with failures that are permanent in nature. The idea of applying proper inputs as tests detecting certain faults is discussed. In Section 2.2

several methods for the generation of tests that detect permanent faults were surveyed, most notably the path sensitizing and the D-algorith methods. The notion of representing the data obtained from the test generation phase by a fault table is explained in Section 2.3. Theorems 2.1 and 2.3 in that section give bounds on the number of tests of a fault table having n faults. Theorem 2.2 gives a necessary and sufficient condition to be able to diagnose every fault, i.e., to have maximum diagnostic resolution. The representation of diagnosis experiments as trees is elucidated in Section 2.4. Sections 2.5 and 2.6 cover the problem of optimizing detection and diagnosis experiments for permanent faults. Unfortunately, the methods that give true optimal experiments are impossibly lengthy for any practical size problem. Suboptimal methods that have been used are surveyed. Most of them locally optimize the testing experiment, i.e., at every stage a decision is made to select the testing strategy that optimizes only that stage.

Chapter III introduces a mathematical model for intermittent faults and deals with the detection of these faults. The model, which is a probabilistic one, is detailed in Section 3.1. Many of the notations used in this model are similar to those employed in pattern recognition literature. Every intermittent fault is thought of as a pattern class. The tests and their outcomes are thought of as random variables. In Section 3.2

a detection procedure, through the repeated application of tests that would detect these faults had their effect been permanent, is proposed. The procedure suggested is analogous to a sequential statistical decision procedure. A decision rule for that procedure must guarantee the termination of this procedure in a finite amount of time. Two decision rules were suggested in Subsection 3.2.1 for the simple case where the circuit can have only one possible fault. One rule compared the posterior probability of the fault with a threshold; the other compared the likelihood ratio (definition 3.1) for that fault with a threshold. It was proved, in that subsection, that these are acceptable rules. The general problem, of having one of n possible faults was tackled in Subsection 3.2.2. A decision rule for this case, based on comparing the posterior probabilities with thresholds, was not acceptable since it did not guarantee a finite length for the procedure. However, a decision rule that compares the likelihood ratios with thresholds was proved to be acceptable. Section 3.3 dealt with optimizing the detection experiment. Finding an optimal solution was shown to be equivalent to an integer programming problem. Theorem 3.3 proves the validity of some suggestions made to reduce the size of such an integer programming problem.

Chapter IV lends itself to the problem of diagnosing intermittent faults. The model, introduced in Chapter III is also employed in this chapter. A diagnosis procedure

employing the repetition of tests is proposed in Section 4.2. The representation of such a procedure by a tree is also discussed in that section. In Section 4.3, the expected length of a subexperiment in that procedure is computed. This made it possible to show that the requirements of the diagnosis procedure guarantees that the expected length of the procedures be finite. Properties of the fault table that are relevant to the diagnosis of intermittent faults are explored in Section 4.4. Theorem 4.2 gives the necessary and sufficient conditions for maximum diagnostic resolution. The conditions in this theorem are much stricter than those needed for the permament fault case. Section 4.5 deals with optimizing the diagnosis experiment. As expected, the method that yields a true optimum solution is impossibly lengthy to solve. Two heuristic approaches are suggested that result in local optimization (optimization of subexperiments only). One relies on maximizing the resolution at the end of the subexperiment, and the other takes into account his resolution together with the expected length of the subexperiment. In Section 4.6, the diagnosis procedures for two special cases are discussed.

5.2 CONCLUSIONS

Intermittent faults constitute a respectable portion of the failures that occur in digital systems.

Ad-hoc methods have been used to coupe with these faults in practice, while formal treatment has been completely ignored despite the need for such a tool. This thesis represents, to the best of the author's knowledge, the first major attempt to treat intermittent faults formally. Detection and diagnosis procedures for these faults, that can be employed in practice, were developed. Some fundamental differences between these procedures and those for permanent faults were pointed out. A great number of problems still remain to be solved. It is hoped that this thesis paves the way for working on these problems.

5.3 SUGGESTIONS FOR FUTURE WORK

Several interesting problems related to the diagnosis of intermittent faults remain to be solved.

The probabilistic model presented in Section 3.1 assumes constant probability for the presence of the effect of an intermittent faults. It will be interesting to study the detection and diagnosis procedures, proposed in this thesis, using different models. The employment of models, that are of the Markov chain type or that have time as a parameter in the probability distributions, as alternatives, deserves exploration.

Another problem is finding other suitable decision rules for the detection procedure suggested in Section 3.2.

More efficient suboptimal solutions for minimizing the detection experiment are also needed.

Variations of the diagnosis procedure suggested in Section 4.2 open a fertile research area. Diagnosis procedures of limited length (or time) are of practical importance.

It will be interesting to find the least number of tests m that satisfies the local symmetry property for n faults.

Other approaches for designing suboptimal diagnosis experiments are also needed. Another interesting problem is that of optimizing the detection and diagnosis experiments when tests costs are taken into account.

An important problem is the detection and diagnosis of intermittent faults in sequential circuits. Unfortunately, despite the importance of this problem, it is expected that it will be quite difficult since the problem of diagnosis of permanent faults in sequential circuits has not been solved to satisfaction yet.

Another important problem is the design of easily diagnosable circuits. For example, desired properties in the fault table could be taken into account when designing digital circuits.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Avizienis, A., "Design of Fault-Tolerant Computers," in 1967 Fall Joint Computer Conf., AFIPS Conf. Proc., pp. 733-743.

[2] Avizienis, A.; Gilley, G. C.; Mathur, F. P.; Rennels, D. A.; Rohr, J. A.; and Rubin, D. K., "The STAR (Self-Testing And Repairing) Computer: An Investigation of the Theory and Practice of Fault-Tolerant Computer Design," in Digest 1971 International Symp. on Fault-Tolerant Computing, pp. 92-96.

[3] Breuer, M. A., "Generation of Fault Detection Tests for Intermittent Faults in Sequential Circuits," in Digest 1972 International Symp. on Fault-Tolerant Computing, pp. 53-57.

[4] Brule, J. D.; Johnson, R. A.; and Kletsky, E. J., "Diagnosis of Equipment Failures," IRE Trans. Rel. Qual. Contr., Vol. RQC-9, pp. 23-24, April 1960.

[5] Carroll, J. J., "Examination of Sequential Circuits : A Model and a Method," Ph.D. thesis, Illinois Inst. of Tech., 1972.

[6] Chang, H. Y., "An Algorithm for Selecting an Optimum Set of Diagnostic Tests," IEEE Trans. Electron. Comput., Vol. EC-14, pp. 706-711, Oct. 1965.

[7] Chang, H. Y.; Manning, E.; and Metze G., Fault Diagnosis of Digital Systems. New York: Wiley-Interscience, 1970.

[8] Clegg, F. W., "Algebraic Properties of Faults in Logic Networks," Ph.D. thesis, Stanford Univ., 1970.

[9] Cobhan, A.; Fridshal, R.; and North, J. H., "An Application of Linear Programming to the Minimization of Boolean Functions," in Proc. 2nd Annu. Symp. Switching Theory and Logical Design, pp. 3-9, 1961.

[10] Cobhan, A., and North, J. H., "Extensions of the Integer Programming Approach to the Minimization of Boolean Functions," IBM Res. Report, RC-915, April 1963.

[11] Dubes, R. C., The Theory of Applied Probability. Englewood Cliffs, N. J.: Prentice-Hall, 1968.

[12] _____, "Pattern Recognition," notes for CPS 817, Dept. of Computer Sci., Mich. State Univ., 1971.

[13] Friedman, A. D., "Fault Detection in Redundant Circuits," IEEE Trans. Electron. Comput., Vol. EC-16, pp. 99-100, Feb. 1967.

[14] Friedman, A. D., and Menon, P. R., Fault Detection in Digital Circuits. Englewood Cliffs, N. J.: Prentice-Hall, 1971.

[15] Fu, K. S., Sequential Methods in Pattern Recognition and Machine Learning. New York: Academic Press, 1968.

[16] Galey, J. M.; Norby, R. E.; and Roth, J. P., "Techniques for the Diagnosis of Switching Failures," IEEE Trans. Commun. Electron., Vol. CE-83, pp. 509-514, Sept. 1964.

[17] Garey, M. R., "Optimal Binary Decision Trees for Diagnostic Identification Problems," Ph.D. thesis, the Univ. of Wisconsin at Madison, 1970.

[18] Gill, A., "Comparison of Finite State Models," IRE Trans. Circuit Theory, Vol. CT-7, pp. 178-179, June 1960.

[19] Gomory, R. E., "All-Integer Integer Programming Algorithm," IBM Res. Report, RC-189, Jan. 1960.

[20] Hillier, F. S., and Lieberman, G. J., Introduction to Operations Research. San Francisco: Holden-Day, Inc., 1967.

[21] Kohavi, I., "Fault Diagnosis of Logical Circuits," in Proc. 10th Annu. Symp. Switching and Automata Theory, pp. 166-173, 1969.

[22] Lyons, R. E., and Vanderkulk, W., "The Use of Triple-Modular Redundancy to Improve Computer Reliability," IBM J. Res. Dev., Vol. 6, pp. 200-209, April 1962.

[23] McCluskey, E. J., Jr., "Minimization of Boolean Functions," Bell System Tech. J., Vol. 35, pp. 1417-1444, Nov. 1956.

[24] McCluskey, E. J., Jr. Introduction to the Theory of Switching Circuits. New York: McGraw-Hill, 1965.

[25] Mealy, G. H., "A Method for Synthesizing Sequential Circuits," Bell System Tech. J., Vol. 34, pp. 1045-1080, Sept. 1955.

[26] Mendel, J. M., and Fu, K. S., Adaptive, Learning and Pattern Recognition Systems. New York: Academic Press, 1970.

[27] Moore, E. F., "Gedanken Experiments on Sequential Machines," in Automata Studies, Annals of Mathematics Studies No. 34, Shannon, C. E., and McCarthy, J., eds., pp. 129-153. Princeton, N. J.: Princeton Univ. Press, 1956.

[28] Moore, E. F., and Shannon, C. E., "Reliable Circuits Using Less Reliable Relays," J. Franklin Inst., Vol. 262, pp. 191-208, Sept. 1956, and pp. 281-297, Oct. 1956.

[29] Poage, J. F., "Derivation of Optimal Tests to Detect Faults in Combinational Circuits," in Proc. Symp. on Mathematical Theory of Automata, Polytechnic Inst. of Brooklyn, pp. 483-528, 1963.

[30] Ramamoorthy, C. V., "Fault-Tolerant Computing : An Introduction and an Overview," IEEE Trans. Comput., Vol. C-20, pp. 1241-1244, Nov. 1971.

[31] Roth, J. P., "Diagnosis of Automata Failures : A Calculus and a Method," IBM J. Res. Dev., Vol. 10, pp. 278-291, July 1966.

[32] Schneider, P. R., "On the Necessity to Examine D-Chains in Diagnostic Test Generation-An Example," IBM J. Res. Dev., Vol. 11, p. 114, Jan. 1967.

[33] Schertz, D. R., and Metze, G. A., "On the Distinguishability of Faults in Digital Systems," in Proc. 6th Annu. Allerton Conference on Circuit and System Theory, pp. 752-760, 1968.

[34] Schnable, G. L.; Ewald, H. J.; and Schlegel, E. S., "MOS Integrated Circuit Reliability," IEEE Trans. Rel., Vol. R-21, pp. 12-19, Feb. 1972.

[35] Sellers, F. F., Hsiao, M. Y., and Bearnson, L. W., "Analyzing Errors with the Boolean Difference, IEEE Trans. Comput., Vol. C-17, pp. 676-683, July 1968.

[36] Szygenda, S. A., "A Simulator for Digital Design Verification and Diagnosis," Proc. Lehigh Univ. Workshop on Reliability and Maintainability, Dec. 1971.

[37] Tryon, J. G., "Quaded Logic," in Redundancy Techniques for Computing Systems, Wilcox, R. H., and Mann, W. C., eds., pp. 205-228. Washington, D.C.: Spartan Books.

[38] Von Neumann, J., "Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components," in Automata Studies, Annals of Mathematics Studies No. 34, Shannon, C. E., and McCarthy, J., eds., pp. 43-98. Princeton, N. J.: Princeton Univ. Press, 1956.

[39] Wald, A., Sequential Analysis. New York: Wiley, 1947.

[40] Yen, Y. T., "Intermittent Failure Problems of Four-Phase MOS Circuits," IEEE J. Solid State Circuits, Vol. SC-3, June 1969.