Example 7.1.1

Consider the circuit of Fig. 7.1.3. The output function of this circuit is $z = f(x_1, x_2, x_3) = x_1x_2 + x_2x_3$. The fault table for detection and location of all the single faults of the circuit is shown in Table 7.1.1. In this table, f_0 denotes the output function of the faultless version of the circuit, and f_{ij} represents the output function of the circuit with its ith line stuck at 0 ($j = 0$) or 1 ($j = 1$). It is observed that all the faults are detectable, and four groups of faults $\{f_{10}, f_{20}, f_{50}\}$, $\{f_{11}, f_{41}\}$, $\{f_{30}, f_{40}, f_{60}\}$, and $\{f_{51}, f_{61}\}$ in this table are indistinguishable faults. We combine them, choose one function from each group, and delete the rest of them from the

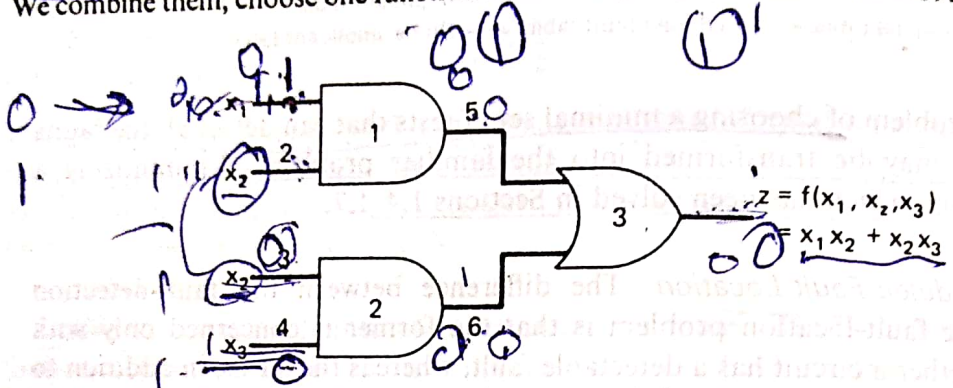
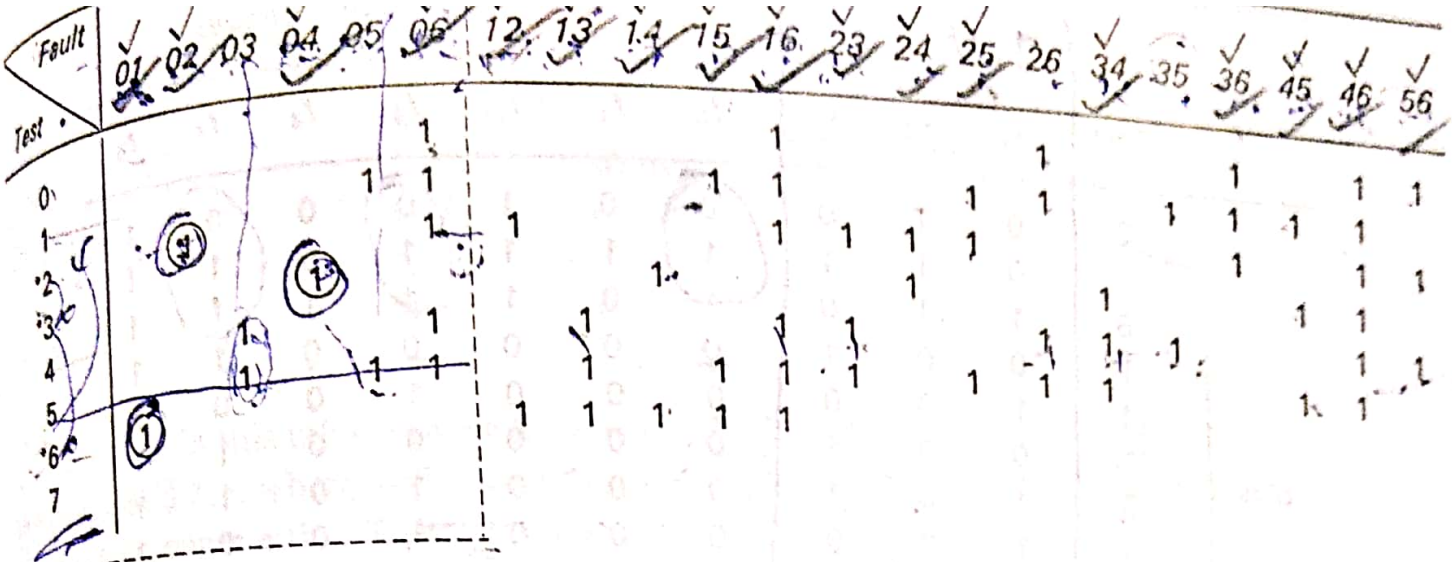


Fig. 7.1.3 Example to illustrate fault location using the fault table.

TABLE 7.1.1 Fault Table for Detection and Location of all Single Faults of the Circuit of Fig. 7.1.2

t	x_1	x_2	x_3	f_0	f_{10}	f_{11}	f_{20}	f_{21}	f_{30}	f_{31}	f_{40}	f_{41}	f_{50}	f_{51}	f_{60}	f_{61}
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
1	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	1
2	0	1	0	0	0	1	0	0	0	0	0	1	0	1	0	1
3	0	1	1	1	1	1	1	1	0	1	0	1	1	1	0	1
4	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1
5	1	0	1	0	0	0	0	1	0	1	0	0	0	1	0	1
6	1	1	0	1	0	1	0	1	1	1	1	1	0	1	1	1
7	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
				f_0	f_1	f_2		f_3	f_4		f_5		f_6			

table. Name the six distinct output functions which represent six distinguishable faults to be $f_1, \ldots, f_6$, as indicated at the bottom the table. From the functions $f_0, \ldots, f_6$, we form the fault-location table G_L, which is shown in Table 7.1.2. The column label kl represents the functions f_k and f_l, from which the column is constructed ($f_k \oplus f_l$), where $0 \le k < l \le m$, and m denotes the total number of distinguishable faults. So there are $C_2^m = m(m-1)/2$ columns in G_L. Note that it is not necessary to show all the 0 entries of this table, and that the subtable G_D of this table, composed of the first six columns, is, in fact, the fault-detection table needed to detect the above-designated faults. From this subtable it is seen that tests 2, 3, and 6, which cover columns 01, 02, 04, and 06, are essential, and columns 03 and 05 can be covered by test 5. Thus the minimal complete test set for detection of all single faults



Fault-detection table G_D

of this circuit is {2, 3, 5, 6}. Note that in this example we only need four out of a total of eight tests to completely check all the single faults of the circuit, which shows a 50% savings of testing time over the test time using the complete truth table. A close examination of the faults that are covered by each of these four tests shows that

Test	x_1	x_2	x_3	
3	0	1	1	test all the s-a-0 faults
6	1	1	0	
2	0	1	0	test all the s-a-1 faults
5	1	0	1	

Now let us examine the fault-location table, which is the entire table of Table 7.1.2. It is observed that tests 2, 3, and 6 are essential tests which cover all the columns except the four columns 03, 05, 26, and 35. With these columns and the nonessential rows (tests), we construct the fault-location table:

Test \ Fault	03	05	26	35

Response of Faulty Circuits

f_0	f_1	f_2	f_3	f_4	f_5	f_6
0	0	1	0	0	0	1
1	1	1	1	0	1	1
1	0	1	1	1	1	1
0	0	0	0	0	1	1
0	0	0	1	0	0	1
0	0	0	0	0	1	1
0	0	0	1	0	1	1
0	0	0	1	0	0	1
0	0	0	1	0	1	1

aults that they detect and locate. This is illustrated
taining only three tests, 2, 3, and 6, cannot detect
est set consisting of these three sets plus one addi-
gle faults. But this additional test must be test 5
3, 6, 1} cannot detect f_3, and test set {2, 3, 6, 4}
consisting of five tests, tests 2, 3, and 6, plus two
ocate all the six distinguishable single faults. The
f the circuit under no fault and a single (distin-
e 7.1.3.

ion and Fault-Location Tests
Circuit of Fig. 7.1.3 with No
Distinguishable Fault

guishable faults $f_1, \ldots, f_6$, we find that the following relation exists between the individual faults and the three gates:

Faults		*Faulty Gate*
f_1, f_3	indicate	gate 1
f_4, f_5	indicate	gate 2
f_2	indicates	gate 1 or gate 2
f_6	indicates	gate 3

The third row in this table demands an explanation. Fault f_2 represents two indistinguishable faults, f_{11} and f_{41}, which indicate a fault in gate 1 and gate 2, respectively. In other words, fault f_2 may come either from f_{11}, which indicates that gate 1 is faulty, or from f_{41}, which indicates that gate 2 is faulty. Thus the presence of f_2 may indicate either gate 1 or gate 2 being faulty (but not gate 3, of course). The G_M for this case is the same as G_L of Table 7.1.2, except that certain columns representing pairs of faults (i.e., 13 and 45) within the same module (gate) need not be included. It can easily be shown that the same test set for locating each individual single fault is required for detecting and locating a faulty gate of the circuit.

Suppose that the circuit is constructed by two modules, module 1, containing the two AND gates, and module 2, containing the OR gate. In this case the relation between the individual single fault and the faulty modules indicated by them is

Faults		*Faulty Modules*
f_1, f_2, f_3, f_4, f_5	indicate	module 1 (gates 1 and 2)
f_6	indicates	module 2 (gate 3)

By deleting columns 12, 13, 14, 15, 23, . . . , 45, from the G_L of Table 7.1.2 and finding the minimal cover for it, we find that the fault-detection test set {2, 3, 6, 5} obtained previously will locate any single fault within the two modules.

Let N_D, N_L, and N_M be the number of test inputs in the minimal experiments for fault detection, fault location, and fault location-to-within-modules. It should be remarked that for detecting or locating a set of faults of a circuit, the following relation always holds:

$$N_D \leq N_M \leq N_L$$

For example, in the example above, $N_D = 4$ and $N_L = 5$. $N_M = 5$ if each gate is considered as a module, and $N_M = 4$ if the two AND gates and the OR gate are considered as two modules of the circuit.

We have been discussing fixed-scheduled fault detection and location. In the following we will discuss the same problems but the test schedule will be adaptive. It will be shown that the use of an adaptive test schedule will be of no benefit in fault detection, but in fault location and fault location-to-within-modules, the possible

reductions in test-schedule length are substantial. A heuristic procedure is given which is easy to carry out and reasonably effective, although it does not necessarily lead to a test whose length is absolutely minimal.

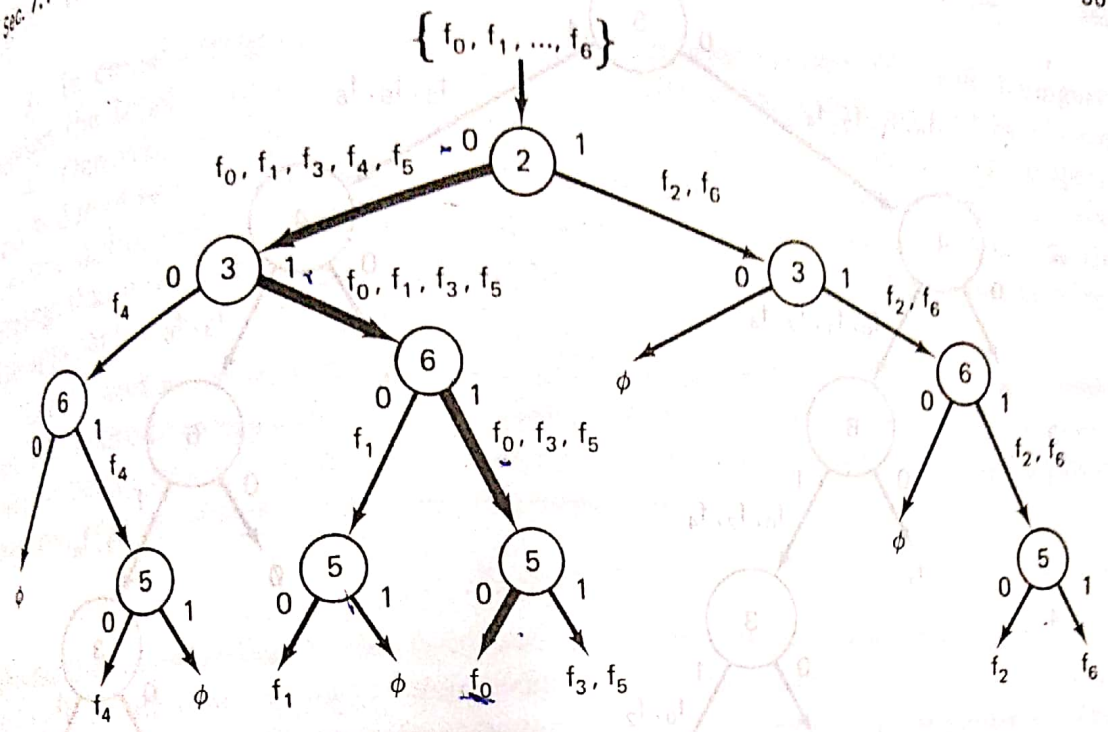
2 *Adaptive-Scheduled Fault Detection and Location* It may be observed that the choice of test schedules derived above is completely independent of the outcome of the individual tests in the sequence; moreover, the length of the schedule is independent of the order in which the tests are performed. It is quite conceivable, however, that after the first test input of a schedule has been applied and the output noted, the residual test schedule, which is minimal with respect to a 0 output, is not the same as that which is minimal with respect to a 1 output. Similarly, after two test inputs have been applied, the four partial test schedules that should follow may be all different in content and length, and so on for successive test inputs. We consider here the economies that can be achieved by choosing each test input to be applied to the circuit on the basis of the outcomes of all previous tests in the schedule. A convenient way to present such a sequence of tests and their outcomes is to use a diagnosing tree, which is defined as follows:

DEFINITION 7.1.5

A *diagnosing tree* is a directed graph whose nodes are tests. The outgoing branches from a node represent the different outcomes of the particular test.

It is easy to show that adaptive-scheduled fault detection requires the same test length as that of fixed-scheduled fault detection. Take the fault detection of the circuit of Fig. 7.1.3, for example. A diagnosing tree of the fixed-scheduled fault-detection experiment {2, 3, 5, 6} for the circuit may be constructed from the fault table (Table 7.1.1), as shown in Fig. 7.1.4(a). From this diagnosing tree, we see that the circuit is fault-free if and only if the output sequence to the sequence (2, 3, 6, 5) is (0, 1, 1, 0), as indicated by the path in dark lines. Thus this tree can be simplified to the one shown in Fig. 7.1.4(b). Applying tests 2, 3, 5, 6 in any order will not shorten the length of the experiment. Another diagnosing tree for test set {2, 3, 5, 6} is shown in Fig. 7.1.4(c), which still requires all four tests. Since the adaptive-scheduled fault detection has no advantage over the fixed-scheduled fault detection, we can completely ignore it.

Unlike fault detection, adaptive-scheduled fault location in general yields a shorter experiment. For example, the fault location of the circuit of Fig. 7.1.3 using the fixed-scheduled experiment, requires



ϕ denotes the empty set

(a)

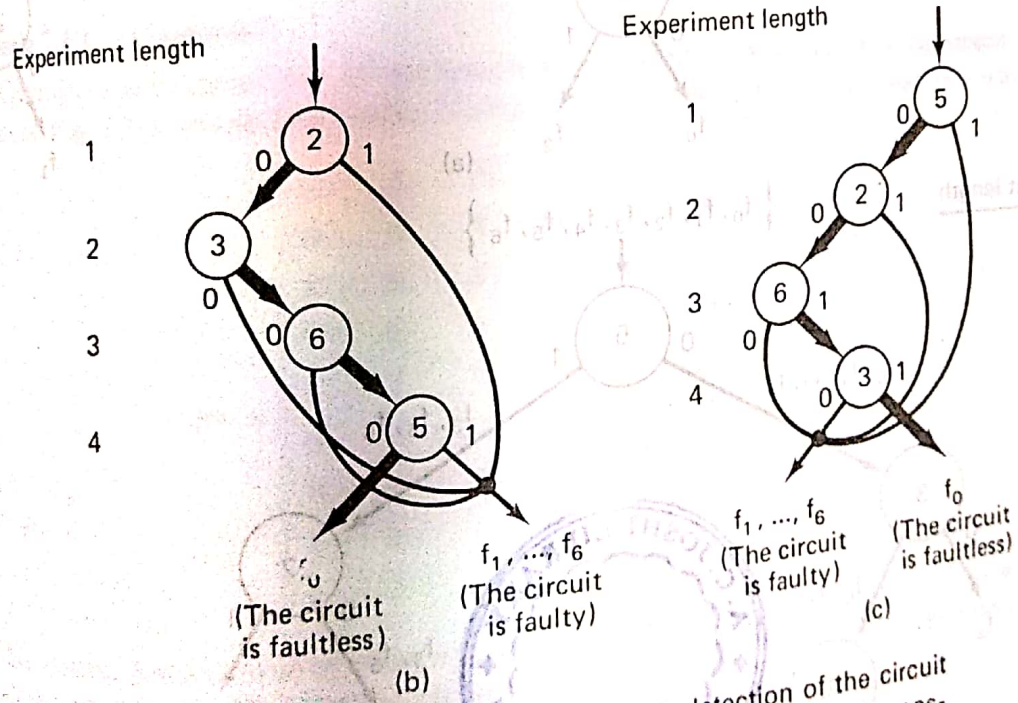
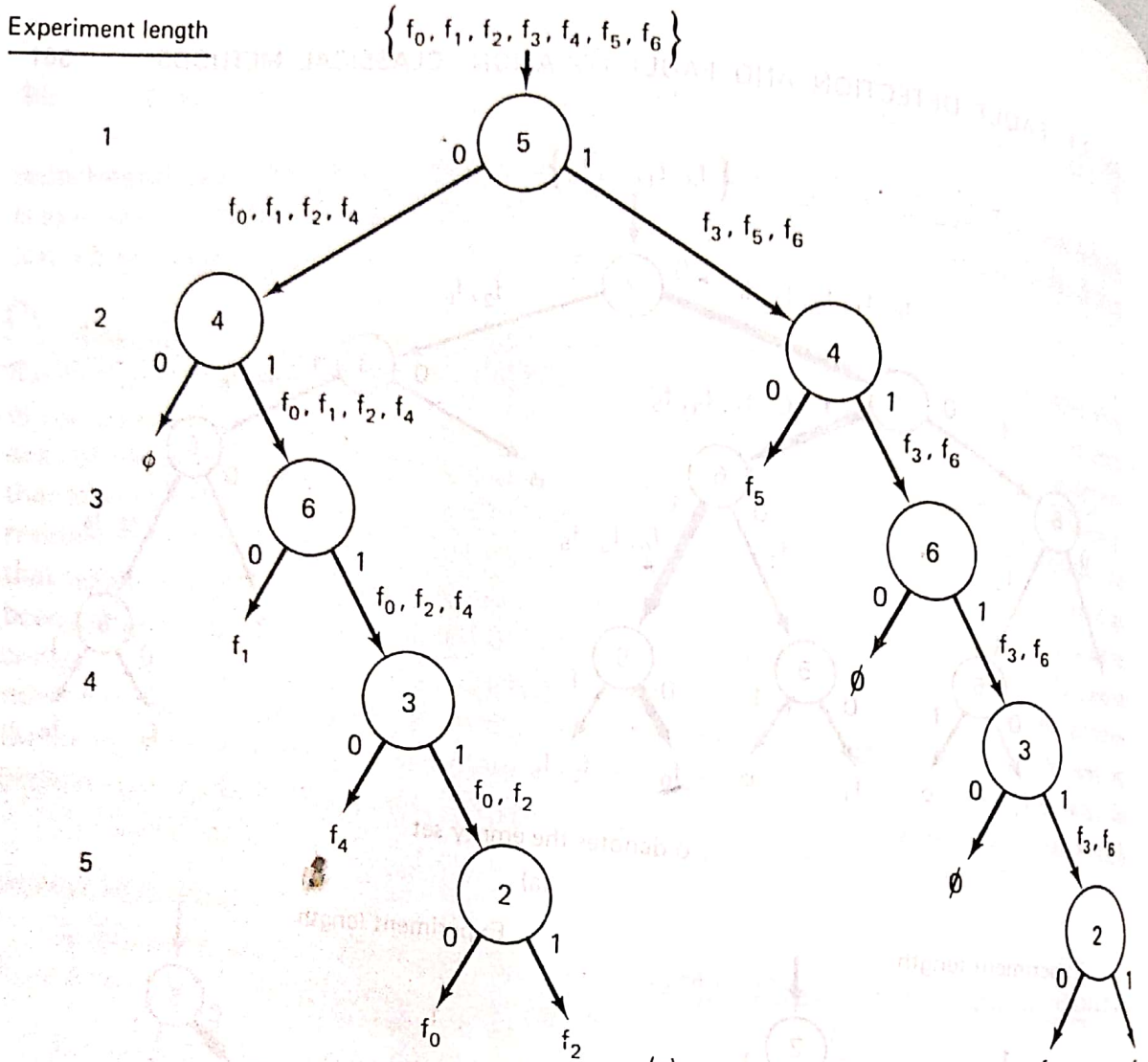


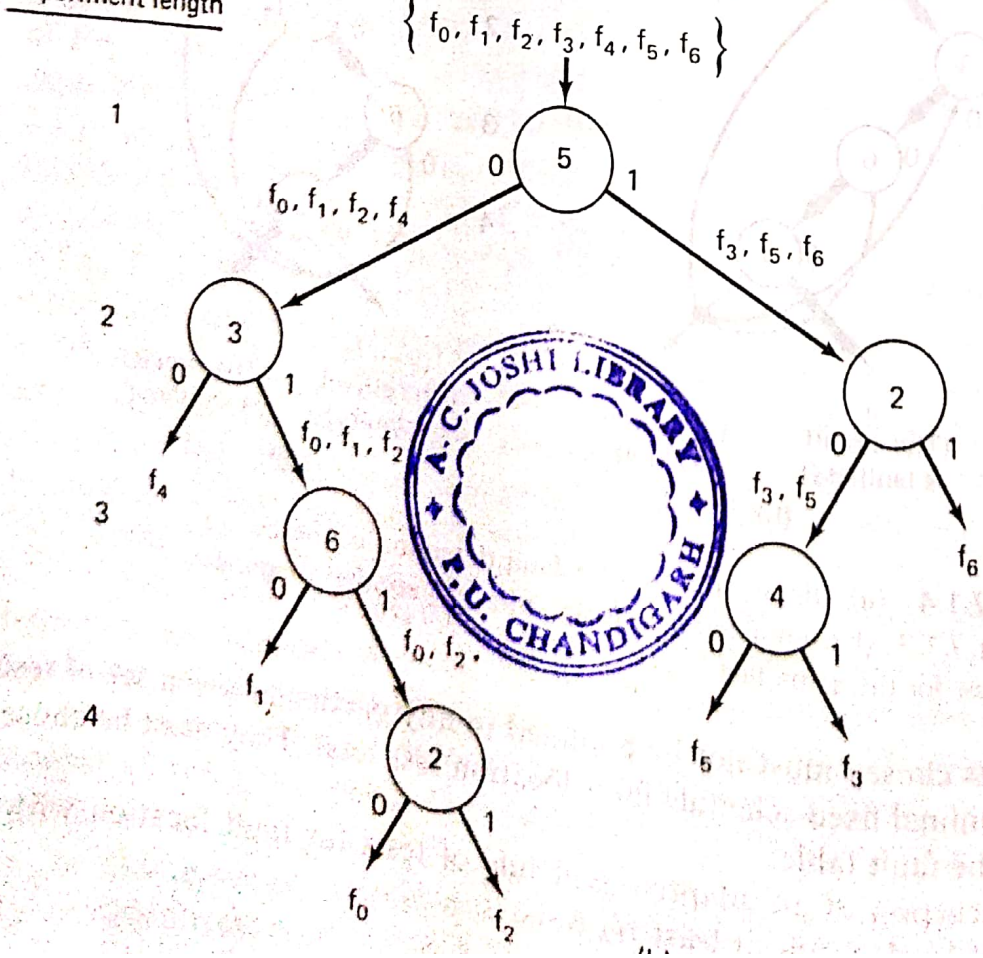
Fig. 7.1.4 (a) Diagnosing tree for fault detection of the circuit of Fig. 7.1.3; (b) simplified diagnosing tree; (c) another diagnosing tree for the same test set {2, 3, 5, 6}.

an experiment, tests chosen must not be confined to any particular given set of tests (e.g., any of the minimal fixed-schedule fault-location test sets). They must be chosen from the rows of the fault table.

2. The construction of an adaptive schedule of tests for fault location with a minimal number of levels needs at least N_L tests.



(a)



3. In constructing this experiment, at each step the test that will distinguish between the largest number of faults not already distinguished should be chosen.
4. One way to select the appropriate row (test) having the desired property described in observation 3 at each step of the procedure is to try all possible remaining rows (tests). For even a small fault table, however, the table number of possible graph labelings that must be tried to determine the minimal number of levels is astronomical. This approach is therefore impractical.

Let w_{i0} and w_{i1} denote the numbers of 0's and 1's in row i, respectively. A simple heuristic method for finding a nearly minimal adaptive-scheduled fault-location experiment is: Select row i, which maximizes the number of (0, 1) pairs between digits in that row, that is, which maximizes the exprssion

$$R_i = w_{i0}w_{i1} \tag{7.1.3}$$

This number is optimized if the row that has the most nearly equal distribution of 0's and 1's, [i.e., the row (or one of the subset of rows) for which $|w_{i0} - w_{i1}|$ is minimal] is selected. The use of this criterion appears to work very well for many problems. This method is illustrated by the following example.

Example 7.1.1 (Continued)

Consider the construction of an adaptive-scheduled fault-location experiment for the circuit of Fig. 7.1.3 whose fault table (Table 7.1.1) may be presented in matrix form as

$$F^* = \begin{array}{c} \begin{matrix} f_0 & f_1 & f_2 & f_3 & f_4 & f_5 & f_6 \end{matrix} \\ \left[\begin{array}{ccccccc|c} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 2 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 3 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 5 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 6 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 7 \end{array}\right] \end{array} \tag{7.1.4}$$

For this matrix, row 5 should be chosen first, since it has 4 0's and 3 1's and $|w_{50} - w_{51}| = 1$, and all the other rows of this matrix whose $|w_{i0} - w_{i1}|$ are greater than 1. Selection of row 5 yields the two submatrices

$$F_0^*(5) = \begin{array}{c} \begin{matrix} f_0 & f_1 & f_2 & f_4 \end{matrix} \\ \left[\begin{array}{cccc|c} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 2 \\ 1 & 1 & 1 & 0 & 3, \\ 0 & 0 & 0 & 0 & 4 \\ & & & & 6 \end{array}\right] \end{array} \qquad F_1^*(5) = \begin{array}{c} \begin{matrix} f_3 & f_5 & f_6 \end{matrix} \\ \left[\begin{array}{ccc|c} 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 2 \\ 1 & 1 & 1 & 3 \\ 1 & 0 & 1 & 4 \\ 1 & 1 & 1 & 6 \\ 1 & 1 & & 7 \\ 1 & & & \end{array}\right] \end{array} \tag{7.1.5}$$

The selection of subsequent rows for the two submatrices should be considered separately. For $F_0^*(5)$, there are three rows, rows 2, 3, and 6, whose $|w_{i0} - w_{i1}| = 2$, and those of the rest of the rows are 4. Thus any of the three rows may be chosen. Say that we choose row 3. Now consider $F_1^*(5)$. It is obvious that any rows of this matrix may be chosen except rows 3, 6, and 7. Suppose that we choose row 2. Then the two submatrices are further partitioned into four smaller submatrices:

$$F_{00}^*(5,3) = \begin{array}{c} f_4 \\ \left[\begin{array}{c} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{array}\right. \end{array} \begin{array}{c} 0 \\ 1 \\ 2 \\ 4 \\ 6 \\ \left.\begin{array}{c} 7 \end{array}\right] \end{array}, \quad F_{01}^*(5,3) = \begin{array}{c} f_0\ f_1\ f_2 \\ \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array}\right. \end{array} \begin{array}{c} 0 \\ 1 \\ 2 \\ 4 \\ 6 \\ \left.\begin{array}{c} 7 \end{array}\right] \end{array}, \tag{7.1.6}$$

$$F_{10}^*(5,2) = \begin{array}{c} f_3\ f_5 \\ \left[\begin{array}{cc} 0 & 0 \\ 0 & 1 \\ 1 & 1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{array}\right. \end{array} \begin{array}{c} 0 \\ 1 \\ 3 \\ 4 \\ 6 \\ \left.\begin{array}{c} 7 \end{array}\right] \end{array}, \quad F_{11}^*(5,2) = \begin{array}{c} f_6 \\ \left[\begin{array}{c} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{array}\right. \end{array} \begin{array}{c} 0 \\ 1 \\ 3 \\ 4 \\ 6 \\ \left.\begin{array}{c} 7 \end{array}\right] \end{array}$$

Now we have two submatrices, $F_{00}^*(5,3)$ and $F_{11}^*(5,2)$, which have only one column. This means that the faults denoted by f_4 and f_6 are already located by the sequences of tests (5, 3) and (5, 2), respectively. Continue this process until all the submatrices form a single-column matrix. One such adaptive experiment having the minimal number of levels obtained by this procedure is shown in Fig. 7.1.5(b).

The method for deriving the shortest test sequence to locate any individual faults just described applies with little change to fault location-to-within-modules. We now modify the criterion of row acceptance to count not all (0, 1) pairs, but only those pairs in each row in which the 0 and the 1 fall in different module classes. This quantity is most easily calculated by subtracting the sum of the number of (0, 1) pairs that fall entirely within the individual classes from the total number of (0, 1) pairs:

$$R_i = w_{i0}w_{i1} - \sum_{j=1}^{p} w_{ij0}w_{ij1} \tag{7.1.7}$$

where w_{ij0} and w_{ij1} are the number of 0's and 1's, respectively, in the jth module class in row i. The row to be selected is the one with the largest row count R_i.

Example 7.1.1 (Continued)

As an example for adaptive-scheduled fault location-to-within-modules, suppose that in the above F^* matrix, faults f_1, f_2, and f_4 are associated with a single module, as are faults

f_5, and f_6. The row counts on the eight rows of F^* are

0: $6 - 0 - 2 = 4$
1: $10 - 0 - 2 = 8$
2: $10 - 2 - 2 = 6$
3: $6 - 2 - 0 = 4$
4: $10 - 0 - 2 = 8$
5: 12
6: $6 - 2 - 0 = 4$
7: 0

Clearly, row 5 should be selected first, and the same two submatrices $F_0^*(5)$ and $F_1^*(5)$ that appeared in Eq. (7.1.5) arise here. $F_1^*(5)$ falls entirely within (and in fact covers exactly) module class (f_3, f_5, and f_6); therefore, it need not be reduced further. For the rows of $F_0^*(5)$, the row-count values are

0: 0
1: 0
2: $3 - 2 = 1$
3: $3 - 2 = 1$
4: 0
6: $3 - 2 = 1$
7: 0

Any of 2, 3, or 6 should be chosen. Selecting 2, only the $F_{00}^*(5, 2)$ need be further reduced, and any of the nonconstant rows 3 or 6 can be used. Figure 7.1.6(b) shows the resulting diagnosing tree. The diagnosing tree for fixed-scheduled fault location-to-within-modules for the same modules is shown in Fig. 7.1.6(a). It is seen that for locating faults f_3, f_5, and f_6 within the module, one test (test 5) is sufficient if the adaptive-scheduled experiment is used.

From the above discussion, the following bounds can readily be established.

$$1 \le l_D = N_D \le m$$
$$1 + \lceil \log_2 m \rceil \le l_L \le N_L \le m$$
$$1 + \lceil \log_2 p \rceil \le l_M \le N_M \le m$$

where m and p denote the number of distinguishable faults under investigation and the number of module classes, respectively; l denotes the number of levels in adaptive test schedules. For most problems, of course, we can expect l_L to be much smaller than N_L.

The heuristic adaptive-scheduled fault-location method provides a good but not necessarily optimal solution. The procedure is quite straightforward and easy to apply.

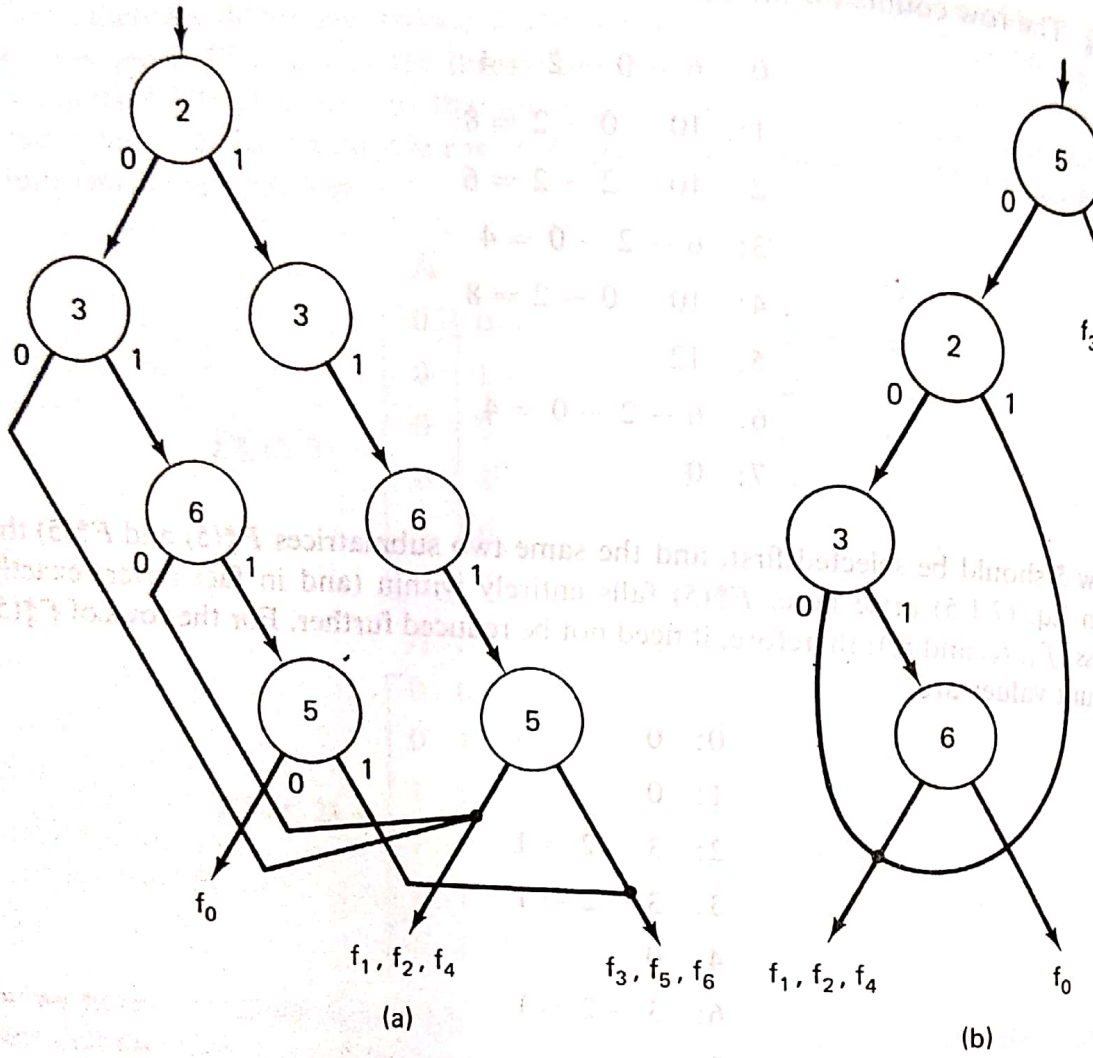


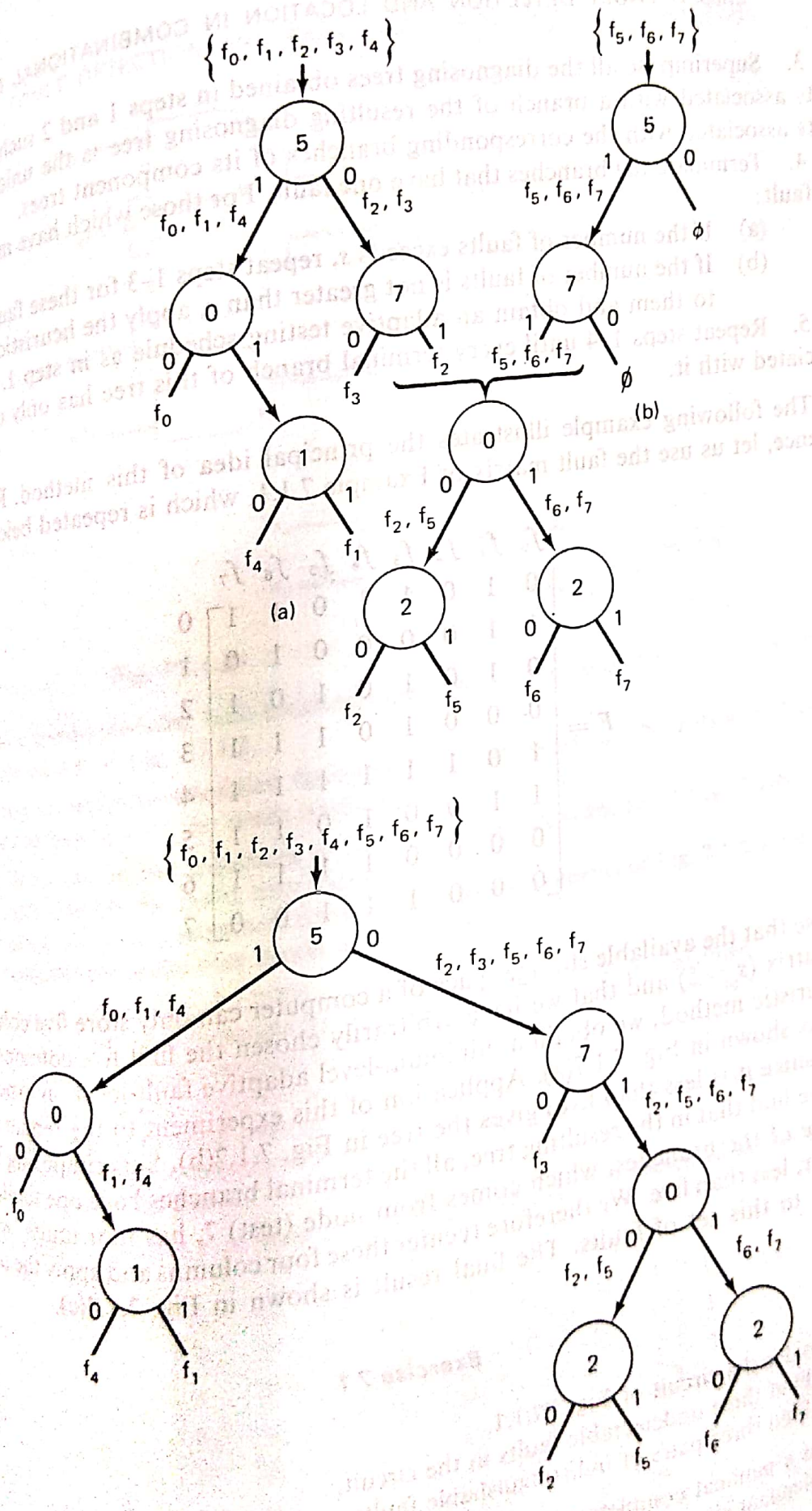
Fig. 7.1.6 Fault location-to-within-modules of the circuit of Fig. 7.1.3. (a) Diagnosing tree of the fixed-scheduled experiment {2, 3, 6, 5}. (b) Diagnosing tree of the adaptive-scheduled experiment {5, 2, 3, 6}.

The drawback of this method is that it requires a large amount of computer storage space to store the fault table. When the size of the fault table exceeds the amount available, we modify the above method to the *hybrid method*, which is described below.

Let n be the number of primary inputs of a circuit and m be the number of single faults of the circuit under consideration. Then the size of the fault table will be $2^n(m+1)$. Suppose that the available memory of a computer can take only s columns of the $m+1$ columns at a time. The hybrid method consists of the following steps:

1. Enter the memory with s arbitrary columns of the fault table and obtain an adaptive-scheduled experiment for it by the heuristic method described above. Record this experiment and then clean up the memory.

2. Enter the rest columns into the computer, s columns at a time, and apply the obtained adaptive-scheduled experiment to each of these $2_n \times s$ submatrices. Note that the application of the adaptive-scheduled experiment to these sets of faults is fixed-scheduled.



Example to illustrate the hybrid method.

3. Superimpose all the diagnosing trees obtained in steps 1 and 2 such that the faults associated with a branch of the resulting diagnosing tree is the union of the faults associated with the corresponding branches of its component trees.

4. Terminate the branches that have one fault. For those which have more than one fault:
   (a) If the number of faults exceeds s, repeat steps 1–3 for these faults.
   (b) If the number of faults is not greater than s, apply the heuristic method to them and obtain an adaptive testing schedule as in step 1.

5. Repeat steps 1–4 until every terminal branch of this tree has only one fault associated with it.

The following example illustrates the principal idea of this method. For convenience, let us use the fault matrix of Example 7.1.1, which is repeated below.

$$F = \begin{array}{c} \begin{matrix} f_0 & f_1 & f_2 & f_3 & f_4 & f_5 & f_6 & f_7 \end{matrix} \\ \left[\begin{array}{cccccccc} 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{array}\right] \end{array} \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{matrix}$$

Suppose that the available storage space of a computer can only store five columns of this matrix ($s = 5$) and that we have arbitrarily chosen the first five columns. From the heuristic method, we obtain a minimum-level adaptive fault-location experiment, which is shown in Fig. 7.1.7(a). Application of this experiment to the remaining columns (since it is less than five) gives the tree in Fig. 7.1.7(b). Superimposing the two trees, we find that in the resulting tree, all the terminal branches have one fault, except that one of the branches, which comes from node (test) 7, has four faults, which is, however, less than five. We therefore reenter these four columns and apply the heuristic method to this set of faults. The final result is shown in Fig. 7.1.7(c).

Exercise 7.1

1. Consider the circuit of Fig. P7.1.1.
   (a) Find three undetectable faults in the circuit.
   (b) Find three pairs of indistinguishable faults in the circuit.
2. Find a minimal complete test set for detecting all distinguishable single faults in the irredundant circuit of Fig. P7.1.2 by the fault-table method.

Show that a complete fixed-scheduled fault-location experiment of this circuit requires four tests, whereas a complete adaptive-scheduled fault-location experiment requires only three levels.

7.2 Path-Sensitizing Method

The fault-table method presented in the previous section requires construction of the fault table, which in general is not practical. For example, if a circuit has, say, 20 inputs and if there are 30 lines within the circuit, then there will be over 30,000,000 entries in the associated fault table. In this section we shall show that a fault-detection test may be found by examining the paths of transmission from the location of an assumed fault to one of its primary outputs. This is the principal idea behind the path-sensitizing method.

DEFINITION 7.2.1

In a logic circuit, a *primary output* is a line whose signal output is accessible to the exterior of the circuit, and a *primary input* is a line that is not fed by any other line in the circuit.

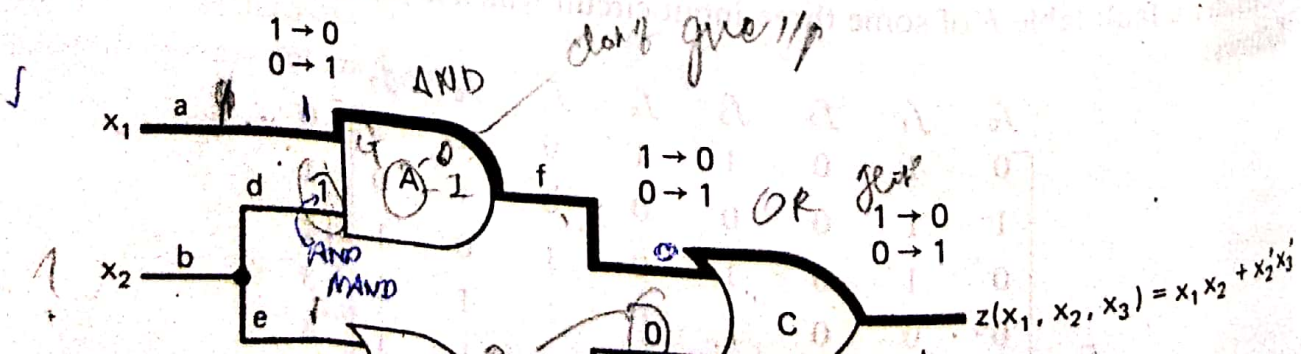
DEFINITION 7.2.2

A *transmission path*, or simply path of a combinational circuit, is a connected, directed graph containing no loops from a primary input or internal line to one of its primary outputs.

DEFINITION 7.2.3

A path is said to be *sensitized* if 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 of the path.

For example, consider the irredundant circuit of Fig. 7.1.1(b), which, for convenience, is repeated in Fig. 7.2.1. Each line of the circuit is labeled by a letter, *a*, *b*,



Value Change on Line a	Corresponding Change in Value on Line f	Corresponding Change in Value on Line h
$0 \longrightarrow 1$	$0 \longrightarrow 1$	$0 \longrightarrow 1$
$1 \longrightarrow 0$	$1 \longrightarrow 0$	$1 \longrightarrow 0$

The actual changes in values along path *afh* upon occurrence of the fault at line *a* are indicated in Fig. 7.2.1. Path *afh* is thus said to be *sensitized*. From the value of line $g = 0$ it is required that c or e or both equal to 1. If we choose to make $e = 1$ and $c = 0$ or 1 (don't care), then the values of d and e are 1 and thus are *consistent*; hence a test exists for detecting the fault a_0 using this method. Moreover, we observe that

Path	Input Variable Values Required to Sensitize the Path	Value of the Input Variable of the Path	Response
afh	$x_2 = 1$, $x_3 = 0$ or 1 (don't care)	$x_1 = 1$	1, if the circuit is faultless 0, if the circuit has fault a_0
		$x_1 = 0$	0, if the circuit is faultless 1, if the circuit has fault a_1

Furthermore, it is apparent that a test which detects fault a_0 also detects faults f_0 and h_0 and that a test which detects fault a_1 also detects faults f_1 and h_1. Thus the above two tests together would detect *all* the s-a-0 and s-a-1 faults on this path.)

From the above simple example, a systematic procedure for detecting a single fault by path sensitizing may be described as follows:

Step 1 Choose a path from the faulty line to one of the primary outputs.

Step 2 Assign the faulty line a value 0 (1) if the fault is a s-a-0 (s-a-1) fault.

Step 3 Along the chosen path, except the lines of the path, assign a value 0 to each input to the OR and NOR gates in the path and a value 1 to each input to the AND and NAND

Step 4 Trace back from gates along the sensitized path toward the circuit inputs. If a consistent input combination (a test) exists, the procedure is terminated. If, on the other hand, a contradiction is encountered, choose another path which starts at the faulty line, and repeat the above procedure.

To appreciate the advantage of this procedure over the fault-table method, an example of deriving a complete test set using the path-sensitizing method is given below.

Example 7.2.1

Consider the same circuit of Example 7.1.1 that was used to illustrate the fault-table method. By applying the above procedure to sensitize the four paths of the circuit, *afh*, *bdfh*, *begh*, and *cgh*, the required input-variable values, the assigned value of the input variable of the paths, and the faults detected by the tests are shown in Table 7.2.1. It is seen that the faults listed in the last column of this table include all the single faults of the circuit. From this table it is an easy matter to obtain a complete test set, which is {0, 2, 5, 7}. This set is one of the four minimal complete test sets obtained in Example 7.1.1, but the effort as well as the information storage space required here for obtaining it is much less compared to that required by the fault-table method.

TABLE 7.2.1 Construction of the Complete Test Set of the Circuit of Fig. 7.2.1 by Sensitizing Its Four Paths

Path Number	Path	Input-Variable Values Required by Sensitizing the Path	Assigned Value of the Input Variable of the Path	Test	Faults Detected by the Test
1	*afh*	$x_2 = 1$ and $x_3 = 0$ or 1	$x_1 = 1$	6 or 7	a_0, f_0, h_0
			$x_3 = 0$	2 or 3	a_1, f_1, h_1
2	*bdfh*	$x_1 = 1$ and $x_3 = 0$ or 1	$x_2 = 1$	6 or 7	b_0, d_0, f_0, h_0
		$x_1 = 1$ and $x_3 = 1$	$x_2 = 0$	5	b_1, d_1, f_1, h_1
3	*begh*	$x_1 = 0$ and $x_3 = 0$	$x_2 = 1$	2	b_0, e_0, g_1, h_1
		$x_1 = 0$ or 1 and $x_3 = 0$	$x_2 = 0$	0 or 4	b_1, e_1, g_0, h_0
4	*cgh*	$x_1 = 0$ or 1 and $x_2 = 0$	$x_3 = 1$	1 or 5	c_0, g_1, h_1
			$x_3 = 0$	0 or 4	c_1, g_0, h_0

An important question arises: Will this method always yield a test for detecting a fault? The answer is no. However, it always yields a test for a class of combinational circuits, known as tree-like circuits, which is defined as follows.

DEFINITION 7.2.4

A *tree-like circuit* is defined as a circuit in which (1) each input is an independent input line to the circuit and (2) the fan-out† of every gate is 1.

The circuits of Fig. 7.2.2(a) and (b) are examples of tree-like circuits. The reason that the path-sensitizing method always yields a test for detecting any single fault in a tree-like circuit is because that a consistent input combination always exists for any

†The definition of the fan-out of

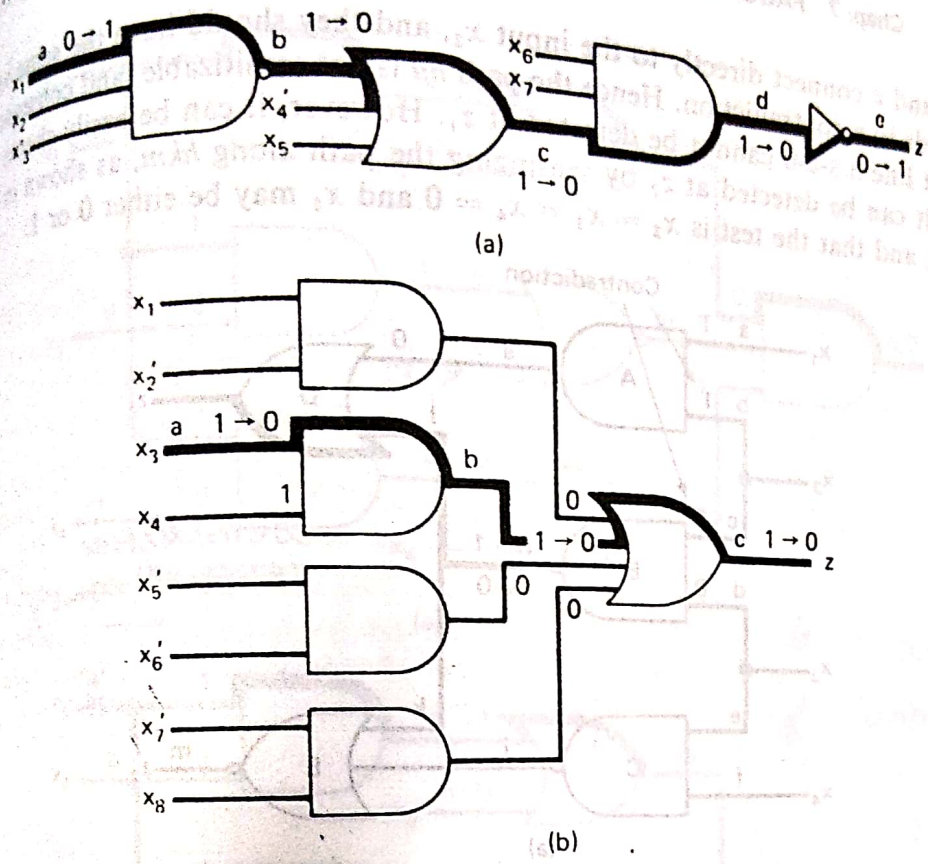


Fig. 7.2.2 Two tree-like circuits.

sensitized path in the circuit. For example, the faults a_1, b_0, c_0, d_0, and e_1 of the circuit of Fig. 7.2.2(a) can be detected by the test $x_1 = 0, x_2 = 1, x_3' = 1, x_4' = 0, x_5 = 0, x_6 = 1, x_7 = 1$, and the faults a_0, b_0, and c_0 of the circuit of Fig. 7.2.2(b) can be detected by the test $x_1 = x_2' = x_5' = x_6' = x_7' = x_8 = 0$ and $x_3 = x_4 = 1$.

From the above discussions, we have:

THEOREM 7.2.1

For any tree-like circuit, a complete test set can always be found by the path-sensitizing method.

Proof: Since every path of a tree-like circuit is sensitizable, we can find a complete test set for detecting any fault for every path. The union of the complete test sets for all the paths of the circuit is a complete test set for detecting any fault in the circuit. The determination of a minimal complete test set from this test set is similar to step 3 of the fault-table method for fault detection. ∎

For general combinational circuits, a contradiction of line value may occur at the gates, which are points of reconvergence for two or more fan-out paths from some preceding gate. For example, if we want to test whether the output of the NOR gate B of the circuit in Fig. 7.2.3(a) is s-a-0, in order to detect the s-a-0 fault, we require $h = 1$, which in turn requires $c = d = 0$. There are two paths from line h to the outputs z_1 and z_2, paths hjl and hkm. Let us first consider the path hjl. According to step 3, line g should be set to logic value 0, which requires $a = b = 1$. But both

lines b and c connect directly to the input x_2, and they should have the same value. This leads to a contradiction. Hence the path hjl is not sensitizable, and consequently the fault line h s-a-0 cannot be detected at z_1. However, it can be easily shown that this fault can be detected at z_2 by sensitizing the path along hkm, as shown in Fig. 7.2.3(b), and that the test is $x_2 = x_3 = x_4 = 0$ and x_1 may be either 0 or 1.

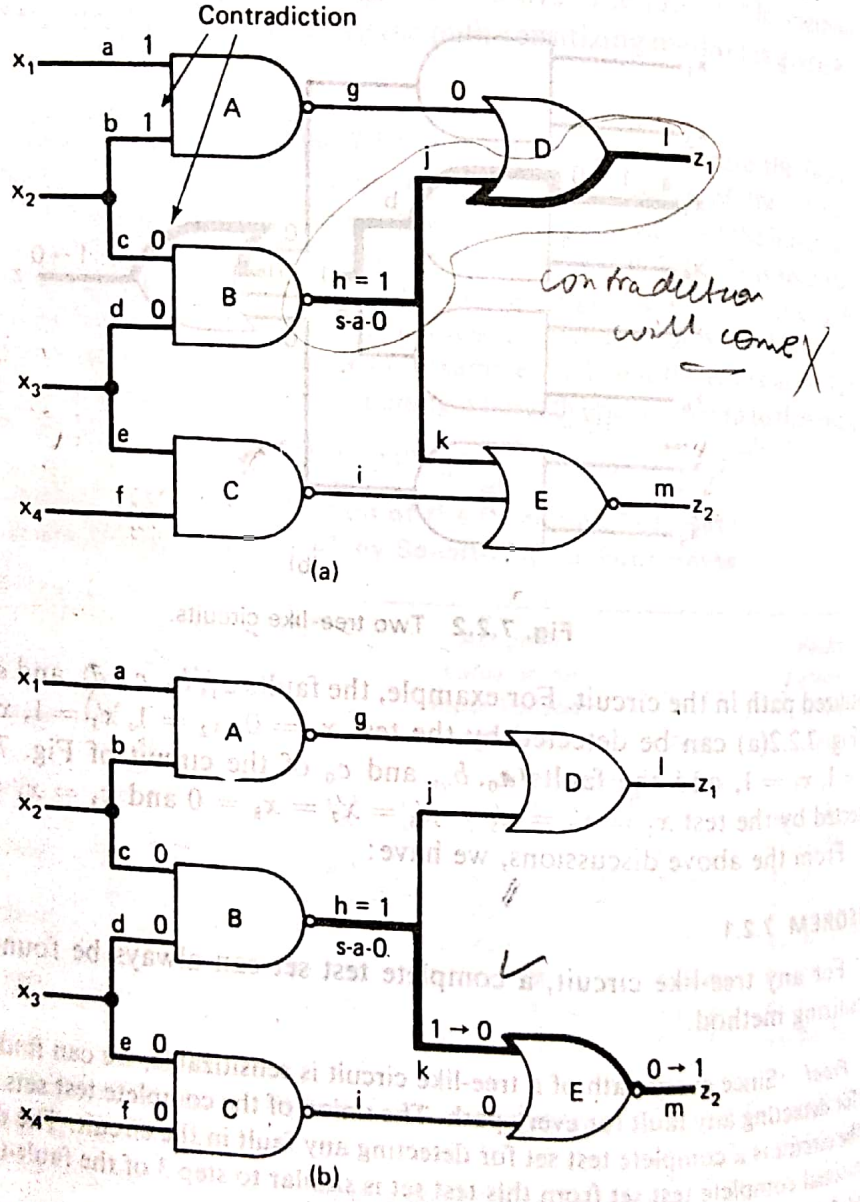


Fig. 7.2.3 Circuit of Example 7.2.2. (a) unsensitizable path hjl; (b) sensitizable path hkm.

It should be pointed out that there are cases where none of the individual paths of a circuit is sensitizable, but if we sensitize a group of them simultaneously, they become sensitizable. For example, none of the individual paths of the circuit of Fig. 7.2.4(a) from x_1 to z for detecting the fault line a s-a-0 is sensitizable, but if we sensitize *all the three paths* from x_1 to z as shown in Fig. 7.2.4(b) simultaneously, they become sensitizable.

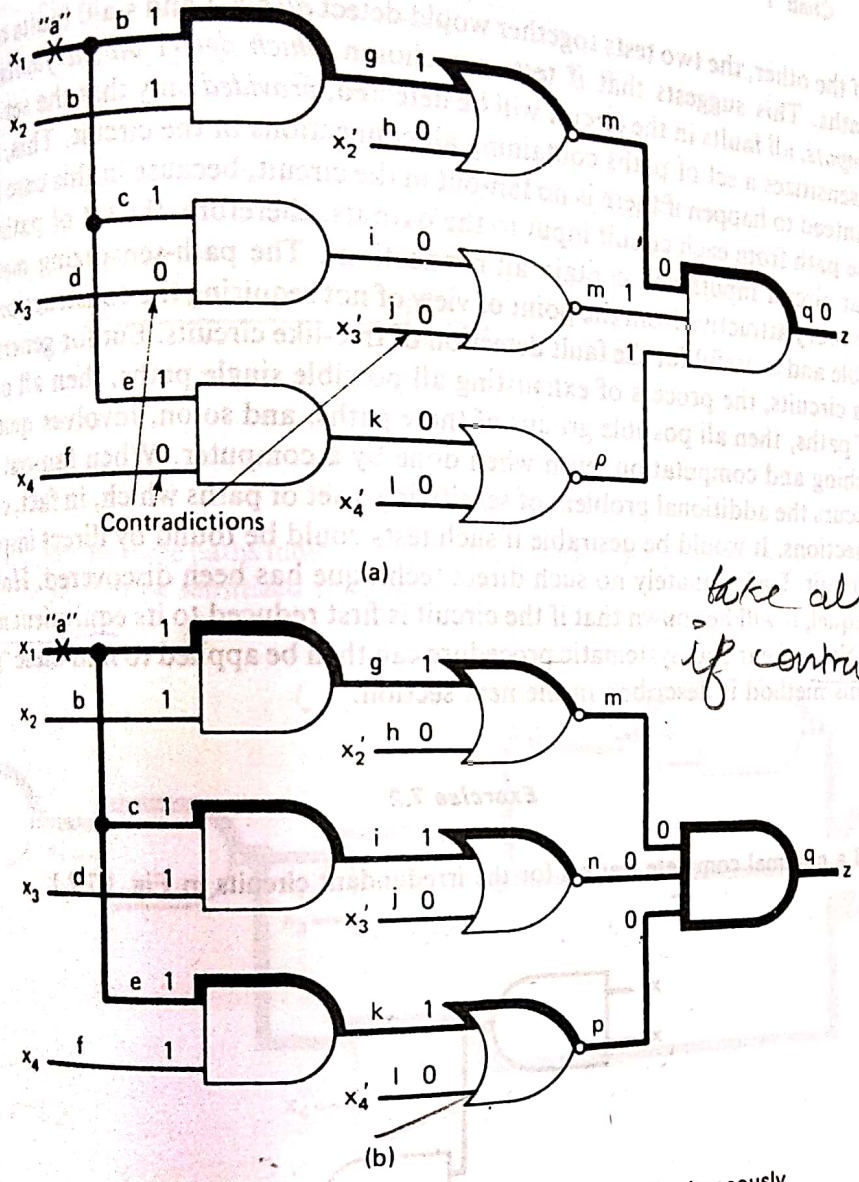


Fig. 7.2.4 Example of sensitizing several paths simultaneously.

This example shows that in searching out the sensitizable path for a particular fault, all possible combinations of single paths sensitized simultaneously must be included. We therefore add one additional step to the above procedure to make it complete.

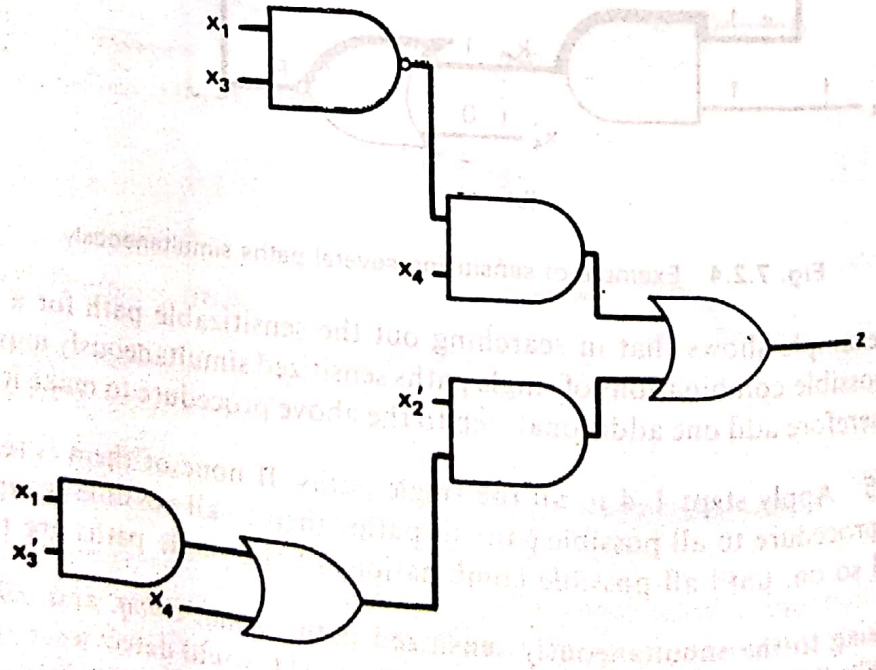
Step 5 Apply steps 1–4 to all the single paths. If none of them is sensitizable, apply the procedure to all possible pairs of paths, then to all possible groups of three paths, and so on, until all possible combinations of the single paths are tried.

Referring to the simultaneously sensitized paths *agmq*, *acinq*, and *aekpq* of the circuit of Fig. 7.2.4(b), it is easily seen that the test 1111 would detect a set of s-a-0 and s-a-1 faults on the lines along the paths, and the test 0111 would detect a set of s-a-0 and s-a-1 faults on the lines along the paths. Since the two tests detect the two sets of faults on the lines along the paths, and one is the complementary set of faults to the

faults of the other, the two tests together would detect *all* s-a-1 and s-a-0 faults on these three paths. This suggests that if tests are chosen *which detect all the faults on the circuit inputs*, all faults in the circuit will be detected, *provided* only that the set of tests chosen sensitizes a set of paths containing all connections in the circuit. This, in fact, is guaranteed to happen if there is no fan-out in the circuit, because in this case there is only one path from each circuit input to the outputs; therefore, the set of paths originating at circuit inputs will contain all connections. The path-sensitizing method is therefore very attractive from the point of view of not requiring the construction of the fault-table and is useful for the fault detection of tree-like circuits. But for general non-tree-like circuits, the process of exhausting all possible single paths, then all possible pairs of paths, then all possible groups of three paths, and so on, involves quite a lot of searching and computation, even when done by a computer. When fan-out exists, there occurs the additional problem of sensitizing a set of paths which, in fact, contain all connections. It would be desirable if such tests could be found by direct inspection of the circuit. Unfortunately no such direct technique has been discovered. However, in the sequel, it will be shown that if the circuit is first reduced to its equivalent normal form (ENF), a heuristic, systematic procedure can then be applied to find these "good" tests. This method is described in the next section.

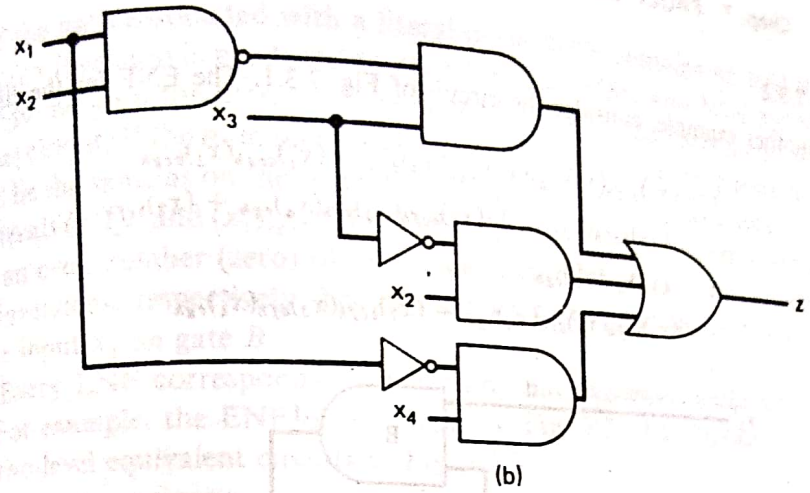
Exercise 7.2

1. Find a minimal complete test set for the irredundant circuits in Fig. P7.2.1.



(a)

Fig. P7.2.1



(b)

Fig. P7.2.1 (continued)

2. Show that the three paths indicated in the circuit of Fig. P7.2.2 cannot be sensitized individually but can be sensitized simultaneously.

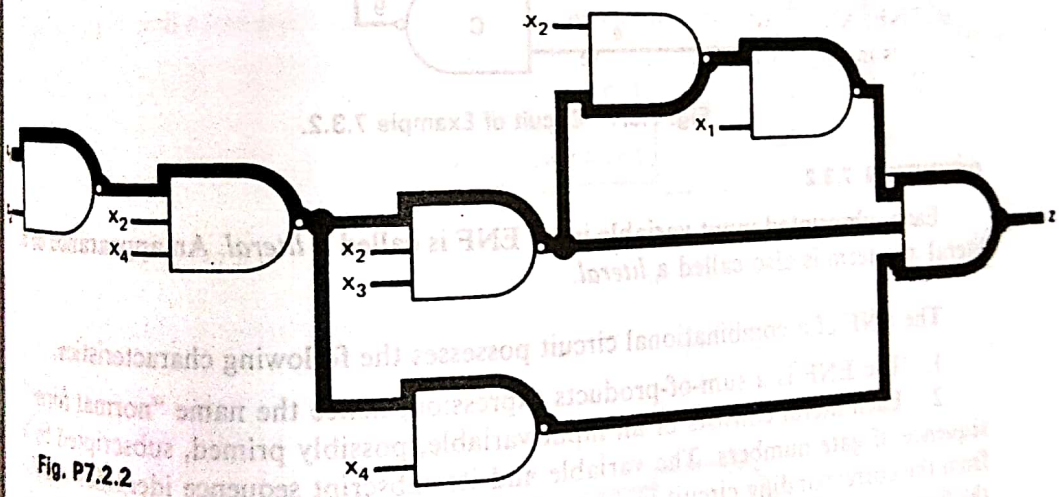


Fig. P7.2.2

7.3 Equivalent-Normal-Form Method

The equivalent normal form of a combinational circuit is defined below.

DEFINITION 7.3.1

The *equivalent normal form* (ENF) of a circuit is obtained by expressing the output of each gate as a sum-of-products expression of its inputs and preserving the identity of each gate by a suitable subscript.

Example 7.3.1 (7.3.1)

The ENF of the circuit of Fig. 7.2.1 is

$$h = (f + g)_h = (ad)_{fh} + (e'c')_{gh}$$
$$= (x_1)_{af h}(x_2)_{bdf h} + (x_2')_{beg h}(x_3')_{cg h}$$