

## An efficient method of determining functional equivalence classes of a network with reconvergent fan-out

JAYAHREE DATTA GUPTA† and GERALD MUSGRAVE‡

This paper is a study of the effects of the faults on the functional operation of a combinational logic circuit. The conditions whereby two different faults can produce the same functional output are investigated. In this approach two fault graphs of the circuits are drawn. By manipulating these fault graphs the faults which are functionally equivalent can be obtained. An algorithm for determining the functionally equivalent classes of faults in a combinational circuit is presented. The unique feature of the algorithm is that it produces the true functional equivalence (not structural equivalence) even for the circuit with reconvergent fan-out with unequal parity.

### 1. Introduction

The indistinguishability among faults in a circuit have been studied by many researchers (McClusky and Clegg 1971, Schertz 1969, Boute 1971). Faults that are indistinguishable are said to be equivalent. There are various types of equivalent relations. Most of the earlier work has been done on the structural equivalence relation McClusky and Clegg 1971, Schertz 1969). None of the existing methods were able to find the true functional equivalence in all types of circuits. While it is true that any pair of faults that are structurally equivalent are always functionally equivalent, the converse is not true. It is possible to have functionally equivalent faults that are not structurally equivalent. This paper presents a graphical representation of the faults in a circuit from which functional equivalence can be obtained. An algorithm is presented for finding the functional equivalence classes present in a network.

In this paper the fault model used to represent circuit failures will be a line stuck-at-1, line stuck-at-0 model. If a line  $i$  is stuck at 1 (0), it is considered to be permanently tied to the logical 1 (0) level. A shorthand notation is used for the fault model, the fault 'line  $i$  s-a-1' is written as  $i^1$ , and the fault  $i$  s-a-0 is written as  $i^0$ .

Faults that are functionally equivalent can always be detected by the same test. All faults that are functionally equivalent can be grouped together and can be regarded as a class often called equivalent class. Thus faults that may occur in a network are partitioned into equivalent classes by equivalence relations. This leads to a deeper understanding of the mechanism by which faults affect logic circuits. This has an immediate application in fault testing and diagnosis. Any testing procedure that has access to a network only through its input and output and which is capable of detecting a fault in the network can be guaranteed to also detect the presence of any fault that is

---

Received 30 June 1977.

† Electronics Unit, Indian Statistical Institute, 203 Barrackpore Trunk Road, Calcutta 700035, India.

‡ Electrical Engineering Department, Brunel University, England.

functionally equivalent to  $F$ . Moreover, after finding the functionally equivalent classes in a circuit, if separate tests can be derived for each class, then the fault location within an equivalence class can also be obtained by applying these tests.

## 2. Properties of faults and their definition

Let  $G$  be a combinational circuit whose output function is denoted by  $Z(G)$ . The output function of the same circuit under fault  $F$  is denoted by  $Z_F(G)$ .

### 2.1. Fault equivalence (McClusky and Clegg 1971)

Two faults  $F_1$  and  $F_2$  in a circuit are functionally equivalent if

$$Z_{F_1}(G) = Z_{F_2}(G)$$

This is denoted as  $F_1 \approx F_2$ .

Any faults that are functionally equivalent are detected by the same tests.

### 2.2. Completely distinguishable faults (Dattagupta 1976)

Two faults  $F_1$  and  $F_2$  are said to be completely distinguishable if  $Z_{F_1}(G, T) \neq Z_{F_2}(G, T)$ , where  $T$  is any possible input combination. This is denoted by  $F_1 d F_2$ .

Any faults that are completely distinguishable from one another should be detected by the disjoint sets of the test (5).

### 2.3. Dominance (Mei 1970)

A fault  $F_1$  dominates another fault  $F_2$  if any test  $t$  that detects  $F_2$  by observation on a primary output line, also detects  $F_1$  on the same output line. This is denoted by  $F_1 > F_2$ .

If  $T_1$  is the test set for detecting  $F_1$  and  $T_2$  is the test set for detecting  $F_2$ , then  $T_1 \cap T_2 = T_2$ .

An equivalence relation automatically gives a dominance relation, but the converse is not true.

## 3. Network graph

In the method to be described in this paper, the circuit diagram is first transformed into two network graphs. Each line in the actual circuit is represented by a node in each of these graphs. If  $l$  is a line in the actual circuit, if one graph contains a node  $l^1$ , then the other graph must contain a node  $l^0$ . These two graphs together contain all single fault nodes and can be applied to find equivalence classes.

### 3.1. Basic rules of construction of the graph

Circuits consisting of AND/OR/NAND/NOT gates are considered here. First the method of drawing fault graphs for each of these gates is described and then the method of drawing the graphs for a complete circuit utilizing these basic gates will be described.

## Fault graphs for an AND gate

Let  $Z$  be the output node and  $I_1, I_2, I_3$  be the input nodes of an AND gate. For an AND gate

$$Z^0 \approx I_1^0 \approx I_2^0 \approx I_3^0 \quad (1)$$

This can be represented graphically as a line, as shown in Fig. 1 (a).

For the same gate

$$I_1^1 d I_2^1 d I_3^1 \quad (2)$$

$$Z^1 > I_1^1$$

$$> I_2^1$$

$$> I_3^1 \quad (3)$$

Relations (2) and (3) can be combined together and can be represented graphically as a fork, as shown in Fig. 1 (b). This shows that a separate test is required for detecting each input line stuck-at-1 and each of them also detects the output stuck-at-1.

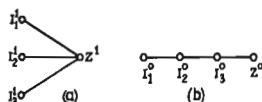


Figure 1. Fault graph for an AND gate. (a) '1' fault graph; (b) '0' fault graph.

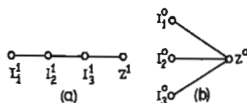


Figure 2. Fault graph for an OR gate. (a) '1' fault graph; (b) '0' fault graph.

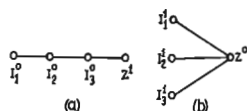


Figure 3. Fault graph for a NAND gate. (a) '1' fault graph; (b) '0' fault graph.

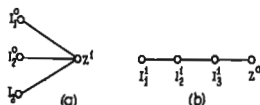


Figure 4. Fault graphs for a NOR gate. (a) '1' fault graph; (b) '0' fault graph.

In a similar way the fault graphs for an OR gate is drawn, as shown in Figs. 2 (a) and 2 (b). Fault graphs for a NAND gate are shown in Figs. 3 (a) and 3 (b). Fault graphs for a NOR gate are shown in Figs. 4 (a) and 4 (b).

### 3.2. Fault graphs for the complete circuit

While drawing the fault graphs for a complete circuit one should start from any of the primary output lines of the circuit and trace backwards until all the different paths starting from the primary output lines and reaching the different primary input lines are covered.

From the nature of the graphs for each type of basic gate it has been found that only two types of connections are possible between the input and output nodes. One type of connection is fork type and the other is of line type.

When drawing the graph for a complete circuit if a line connection is obtained, input nodes are not actually connected to the succeeding input nodes in the first stage. But they point towards the next input node, that is  $I_1$  pointing to  $I_2$ ,  $I_2$  pointing to  $I_3$ , etc. Only the last input node  $I_n$  is connected to the output node. In the next stage one proceeds with the last input node and tests whether it is a primary input node or not.

If it is a primary input node, then this node is connected to the preceding node which is pointing towards it, that is  $I_n$  will be connected to  $I_{n-1}$ , etc. But if  $I_n$  is not a primary input node, then it must be an intermediate point and so one traces farther backwards until one reaches the primary inputs connected to these intermediate nodes. Let  $J_1$  and  $J_2$  be the primary nodes connected to  $I_n$ , then  $I_{n-1}$  will be connected to these primary nodes  $J_1$  and  $J_2$  instead of  $I_n$  and  $J_1$  and  $J_2$  will in turn be connected to  $I_n$ . This will be best illustrated with an example.



#### Example

For the circuit shown in Fig. 5 the actual drawing of the graphs is shown in Fig. 6 from step 1 to step 6. The complete graph is obtained in step 6.

Once the complete graphs are obtained, the next step will be to find equivalent classes from these two graphs. By manipulating these graphs the fault equivalence of a network can be obtained.

Before describing the actual algorithm, some definitions of the theorems regarding the graphs will be discussed.

**A path.** A path in a graph is a line which passes through some fault nodes and starts with some definite starting-point and end-points.

Thus  is a path  but EFGHE is not a path.

**A complete path.** A complete path in a graph is defined to be that path which starts from one end of the graph and ends in the other end.

**Primary nodes.** Primary nodes in a path are defined to be those nodes which lie on the main path but not on any branch or parallel path.

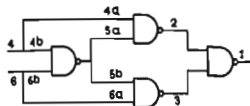


Figure 5. The circuit diagram.

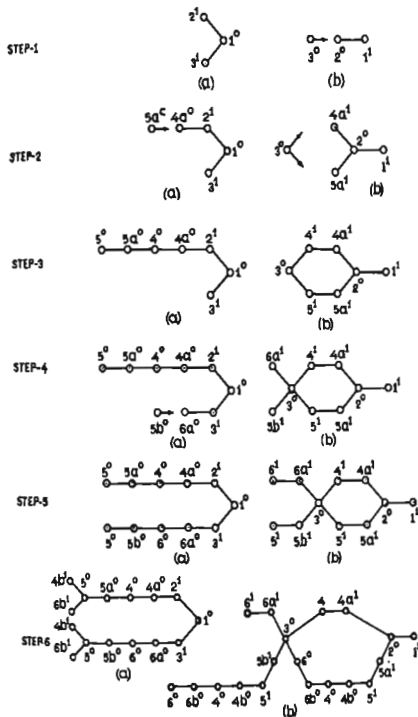


Figure 6. Fault graphs for the circuit of Fig. 5. (a) '0' fault graph; (b) '1' fault graph.

**Secondary nodes.** Secondary nodes in a path are defined to be those nodes which lie on the branch portions of the path.

In the graph shown in Fig. 7, the complete path from node 1 is 1→(2, 3)→4→5. Of these, 1, 4, 5 are primary nodes and (2, 3) are secondary nodes.

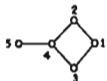


Figure 7. The graph.

*Theorem 1*

In any path, if a primary node  $m^v$  or  $n^v$  exists, then in the same path primary node  $m^{\bar{v}}$  or  $n^{\bar{v}}$  cannot exist, where  $m$  and  $n$  are fan-out branches from the same point and  $v = \{0, 1\}$ .

*Proof*

Let, in one path, both nodes  $m^0$  and  $n^1$  exist, then for detecting those faults the signals present at the fan-out stem will be 1 and 0 at the same time, which is not possible.

If in any path both  $m^{\bar{v}}$  and  $n^{\bar{v}}$  exist, then that line is not a path.

*Lemma 1*

If in a path a primary node  $m^v$  exists and in some branch path a secondary node  $n^{\bar{v}}$  exists, then that branch line on which secondary node  $n^{\bar{v}}$  exists will be detected from that path.

*Proof*

The proof follows from theorem 1.

*Theorem 2*

If  $l$  is a fan-out stem and  $l_a$  and  $l_b$  are fan-out branches from that stem, if  $l_a^v \approx l_b^v$  where  $v \in \{0, 1\}$ , then  $l^v \approx l_a^v \approx l_b^v$ .

*Proof*

The fan-out stem stuck-at-fault makes both fan-out branches faulty. And if both fan-out branches are stuck-at-0 (1), they produce the same effect as the fan-out stem stuck-at-fault 0 (1):

$$l^v \approx l_a^v \cdot l_b^v$$

Now if

$$l_a^v \approx l_b^v$$

then

$$l^v \approx l_a^v \approx l_b^v$$

**4. Description of the algorithm**

Utilizing these definitions and theorems the functional equivalence in a circuit from these graphs will be obtained as shown below.

*Step 1*

Start with the primary output node.

*Step 2*

From this node try to find a complete path.

*Step 3*

Check whether any branch lines are non-existent. If not go to step 5.

*Step 4*

If any branch line is non-existent, delete that branch and check whether only one branch line exists in the reduced path. If a single branch line exists, then consider it as a main line in the path. Go to step 5.

*Step 5*

All primary nodes along this path excluding the fan-out stem nodes and inputs feeding that fan-out stem points are grouped together and are equivalent.

*Step 6*

The fan-out stem node and its input feeding nodes are grouped together and are equivalent.

*Step 7*

Start with the preceding node that is not yet covered by any equivalence group and go to step 2.

*Step 8*

Continue step 1 to step 7 until all nodes in the graphs are covered. The algorithm will be best illustrated with an example.

*Example*

From the example 1 the graphs are obtained as in Fig. 6.

Starting with one of these graphs, say 6 (a), the equivalent classes for this graph are obtained as shown below.

From output node  $1^0$  the complete path is obtained but no other primary nodes exist on this path.

So node  $1^0$  is not equivalent to any other fault and so is a single item in class 1.

$$C_1 = \{1^0\}$$

Next proceeding with the preceding node  $2^1$  a complete path  $2^1 \rightarrow 4a^0 \rightarrow 4^0 \rightarrow 5a^0 \rightarrow 5^0 \rightarrow (4b^1, 6b^1)$  is obtained. Of this,  $4b^1, 6b^1$  are secondary nodes, the rest of them are primary nodes. But of the branch  $(4b^1, 6b^1)$  the branch  $4b^1$  is non-existent for this path by lemma 1, as  $4a^0$  lies on the main path. Then  $6b^1, 5^0, 5a^0, 4^0, 4a^0, 2^1$  are all primary nodes. Of this,  $4^0, 5^0$  are faults on the fan-out stem and so are excluded.

So

$$C_2 = \{6b^1, 5a^0, 4a^0, 2^1\}$$

$$C_3 = \{5^0\}$$

$$C_4 = \{4^0\}$$

Similarly, starting with the node  $3^1$ , a complete path is obtained, and after deleting the non-existent branch paths we get the nodes (primary)

$$3^1 \rightarrow 6a^1 \rightarrow 6^0 \rightarrow 5b^0 \rightarrow 5^0 \rightarrow 4b^1$$

Of these,  $5^0$ ,  $6^0$  are fan-out stems.

So

$$C_6 = \{3^1, 6a^0, 5b^0, 4b^1\}$$

$$C_6 = \{6^0\}$$

All the nodes in the graph 6 (a) are covered.

Starting with the graph 6 (b) a complete path is obtained through node  $1^1$ . Of these,  $1^1$ ,  $2^0$ ,  $3^0$  are primary nodes :

$$C_7 = \{1^1, 2^0, 3^0\}$$

Next proceed with node  $4a^1$  and a complete path  $4a^1 \rightarrow 4^1 \rightarrow 3^0 \rightarrow ((6a^1 \rightarrow 6^1), (5b^1 \rightarrow 4b^0 \rightarrow 6^0))$  is obtained. Of these, the branch  $5b^1 \rightarrow 5^1 \rightarrow 4b^0 \rightarrow 6^0$  is non-existent by lemma 1.

So  $4a^1$ ,  $4^1$ ,  $6a^1$ ,  $6^1$  are all primary nodes. Of these,  $4^1$ ,  $6^1$  are excluded, as they are fan-out stem faults :

$$C_8 = \{4a^1, 6a^1\}$$

$$C_8 = \{4^1\}$$

$$C_{10} = \{6^1\}$$

Similarly with node  $5a^1$ ,  $5a^1 \rightarrow 5^1 \rightarrow 4b^0 \rightarrow 6^0 \rightarrow 5b^1 \rightarrow 5^1 \rightarrow 4b^0 \rightarrow 6^0$  is a path. Of these,  $\{5^1, 4b^0, 6b^0\}$  are excluded. So  $C_{11} = \{5a^1, 5b^1\}$ .

Since  $5a^1 \approx 5b^1$ , by theorem 2,  $5^1 \approx 5a^1 \approx 5b^1$  :

$$C_{11} = \{4b^0, 6b^0, 5^1, 5a^1, 5b^1\}$$

So there are 11 classes of equivalent faults existing in the circuit.

Thus any single fault can affect the output function in only 11 different ways.

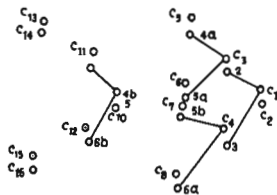


Figure 8. Diagram for finding equivalence class.

The same example with Schertz's method generates 15 equivalent classes, as shown in Fig. 8. But in the actual case, if the output functions are generated for all single faults, it will produce 11 different output functions.



The reason for having more equivalence classes in Schertz's method is due to the fact that it generates structural equivalence classes not functionally equivalent classes. The existence of functionally equivalent faults which are not structurally equivalent has been shown to depend directly on the presence of reconvergent fan-out paths in the network (Boute 1971).

#### 5. Conclusion and discussion

The basic motivation of this work is to find equivalence classes of faults present in a circuit. It was possible to find equivalent classes due to various types of a structural equivalence relation in a circuit. But so far for networks with a reconvergent fan-out an efficient procedure for calculating functional equivalence classes was not known.

This paper presents an algorithm which generates functional equivalent classes of faults present in a circuit. It can handle circuits with reconvergent fan-outs with unequal inversion parity. The algorithm presented here is straightforward and can be easily programmed on a digital computer. The algorithm can detect those faults which are functionally equivalent. Though the work presented here is mainly theoretical, it does have some direct application both to testing and diagnosis. Most procedures for developing tests for combinational circuits start with a list of faults to be tested; it follows directly from the definition of functional equivalence class. Thus the list of faults need contain only one representative from each such equivalence class rather than each individual fault. Fault equivalence is also relevant to diagnosis, since it is not possible to distinguish the faults of an equivalence class by observing only the network outputs.

#### REFERENCES

- BOUTE, R., 1971, Technical Note 9, Digital Systems Laboratory, Stanford University, Stanford, California.
- CLUGG, F. W., and McCUSKY, E. J., 1970, Rep. FU-SEL-69, p. 149, Stanford University, Stanford, California.
- DATTAGUPTA, J., 1976, M.Phil. Thesis, Brunel University, England.
- McCUSKY, E. J., and CLUGG, F. W., 1971, *I.E.E.E. Trans. Comput.*, **20**, 1285.
- SCHERTZ, D. R., 1969, Report R-418, Coordinated Science Laboratory, University of Illinois, Urbana, U.S.A.
- MI, K. C. Y., 1970, Technical Note 2, Digital Systems Laboratory, Stanford University, Stanford, California.