Test calculation for logic and short-circuit faults in digital circuits

József Sziray
Department of Informatics,
Széchenyi University, Győr, Hungary
sziray@sze.hu

Abstract – In the first part, the paper presents a test calculation principle which serves for producing tests of logic faults in digital circuits. The name of the principle is composite justification. The considered fault model includes stuck-at-0/1 logic faults. Both single and multiple faults are included. In this paper only combinational logic is taken into consideration. The computations are performed at the gate level. In the second part of the paper, the composite justification is extended to other fault class, namely, short-circuit faults. A short circuit is an erroneous galvanic coupling between two circuit lines. The calculation principle is comparatively simple. It is based only on successive line-value justification, and it yields an opportunity to be realized by an efficient computer program.

I. INTRODUCTION

Due to the ever increasing complexity of digital integrated circuits, associated with the rapid development in their manufacturing processes, the importance of testing for correct functioning is steadily increasing. All this implies new fault models which require new methods in test design [1]. The fulfillment of this requirement is especially important in the field of VLSI CMOS circuits which are widely used in the modern hardware construction.

At first, the paper gives an overview of a general test-calculation principle that is suitable for producing tests for digital circuits which are modeled at the gate level [2]. It is assumed that the circuits are of combinational type. The principle is comparatively simple, and it yields an opportunity to be realized by an efficient computer program. It should also be added that the calculation process can be extended to sequential circuits in a straightforward way [2].

The considered fault model includes stuck-at-0/1 logic faults on the primary inputs and the gate outputs. Both single and multiple faults are included.

The second part of the paper deals with another fault class, namely, short-circuit or bridging faults [1]. A short circuit is an erroneous galvanic connection between two circuit lines. Here the composite justification is extended to handle this type of faults, as well.

II. THE BASIC PRINCIPLE

First the test calculation principle in general is presented below.

Let the vector of primary input and output variables for a general logic network be \( \mathbf{x} = (x_1, x_2, \ldots, x_n) \) and \( \mathbf{z} = (z_1, z_2, \ldots, z_m) \), respectively. Let the set of possible logic values in the network be \( V = \{v_1, v_2, \ldots, v_s\} \). In addition to the elements of \( V \), the indifferent or don't care value \( d \) will be applied.

The principle of test calculation is based on the formerly elaborated composite justification algorithm that has originally been intended for various permanent logic faults, such as stuck-at, and functional faults [2]-[4]. At first, the overall concept of composite justification will be summarized.

Suppose that a sequence of primary input vectors \( \mathbf{X}(t) = \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t \) detects \( q \geq 1 \) simultaneous logic faults at a primary output \( z_j \). Now the task of calculating \( \mathbf{X}(t) \) can be stated in the following way. Find a sequence of input patterns which implies \( z_j = \alpha \) in the fault-free network, and \( z_j = \beta \) in the faulty network, where \( \alpha \in V, \beta \in V, \) and \( \alpha \neq \beta \).

To reach this goal, we associate the logic values \( \alpha \) and \( \beta \) with \( z_j \), and attempt to find a sequence of input patterns which equally justifies

1) the normal value of \( z_j \) for the normal (fault-free) network, and
2) the faulty value of \( z_j \) for the faulty network.

In the first case, \( \mathbf{X}(t) \) justifies \( z_j = \alpha \) through the values of all the necessary network lines in the usual manner. In the second case, however, since the faulty values are self-dependent, they need not be justified by \( \mathbf{X}(t) \). Thus, in the faulty network, \( \mathbf{X}(t) \) and the faulty values justify \( z_j = \beta \) jointly.

The test sequence can be derived by applying the line-value justification concept. As known, line-value justification is a procedure with the aim of successively assigning input values to the logic
elements in such a way that they are consistent with each previously assigned value. (This concept is an auxiliary calculation process for justifying an initial set of logic values in a network, first applied in the D-algorithm for two-valued logic [1], [5].)

In our approach, the computations are carried out simultaneously in the normal and faulty network, i.e., in the normal and the faulty domain. Logic values simultaneously representing signal values in both the normal and the faulty networks are called composite values. Line justification performed in terms of composite values is referred to as composite justification [2]-[4]. The two components of a composite value will be separated by a slash, with the normal component preceding the faulty one. The actual logic value of the i-th line in the network will be denoted by \( v(i) \). Then, for example, a composite value of line \( i \) is

\[
v(i) = v_a / v_b\]

In the composite justification the computational costs can be greatly reduced by the following consideration [2]. The signal values in the normal and faulty networks cannot differ at the lines that do not carry any signals propagating from the sites of the faults. These are called inactive lines, for which \( v_a ≠ v_b \) would represent inconsistency if they had the composite value of \( v_a / v_b \). It should be realized that for our purposes it is sufficient to determine which lines (called potentially active lines or PAL’s) carry signals from the faulty lines to \( z_j \). This holds true, since all the other lines in the network are either inactive or are not involved in the justification process. The set of potentially active lines can be generated by intersecting the set of the lines that are reached from the faulty lines with the set of the lines from which \( z_j \) can be reached. The two sets are very easy to obtain by topologically tracing out the signal connections. This is done by starting from the faulty lines and proceeding forward, then starting from \( z_j \), and proceeding backward. If a line is encountered in both the forward and backward tracing then it is potentially active. At the end of the calculations, those lines which actually carry the fault signal to \( z_j \) will be the real active lines. These lines have the so-called active composite values which are equivalent with the fault signals.

The other consideration relates to the initial values associated with line \( z_j \). It is not known in advance which normal and faulty values are to be assumed. Therefore we make an arbitrary choice. However, in the case of single faults, there is no need to repeat the justification process with interchanged values for \( z_j \), even if the initial choice has failed. Whenever the last in a series of active values along a path between the fault site \( i \) and \( z_j \) encounters contradiction at \( i \), we have to interchange the components of each composite value, then proceed with the calculations in the same way as before.

The implementation of the above principle for a synchronous sequential network may require the justification process to be performed through different storage states of the network, which results in a test sequence \( X(t) \) of length \( t \). The detailed principle of doing it for stuck-at-0 / 1 faults is described in [2]-[4].

### III. Extension to Short-Circuit Faults

In the following we are going to deal with an other fault class, and show how the composite justification can be extended in a straightforward way to calculating tests for this class [4]. Here galvanic coupling between circuit lines, i.e., short-circuit faults or bridging faults [1] will be considered.

A short-circuit fault is defined for two signals. This type of fault occurs when a signal erroneously takes up the normal logic value of another signal. In this case the two signals are said to be bridged (coupled). Only single occurrence is stipulated here for the faults. The other assumption is that no feedback coupling occurs between any two lines, which would cause a combinational network to become a sequential one.

Let the dominant logic value of two bridged lines be \( \alpha \). It means that the value \( \omega \) will appear on both lines, instead of the normal differing values. If the bridged lines are \( b_1 \) and \( b_2 \), then the initial values for the composite justification will be as follows:

\[
z_j = \alpha / \beta \text{ for a selected primary output, where}
\]

\[
\alpha \in V, \quad \beta \in V, \quad \text{and} \quad \alpha ≠ \beta.
\]

on the other hand,

\[
v(b_1) = v(b_2) = d / \omega.
\]

In the justification process, the following measures have to be taken for the two bridged lines:

1. If either of them is reached by a determined logic value (i.e., a value other than \( d \)) in the normal domain, then the value of the other bridged line must be set to \( \omega / \omega \). (Note that a completely determined composite value first appearing at a bridged line has necessarily different values, i.e., it is a possible fault signal.)

2. If in a backtrack process, \( b_1 \) or \( b_2 \) is reached, then the initial value \( d / \omega \) has to be assigned to both lines again, as the solution when canceling the former decisions, i.e., when canceling the former calculated logic values.
As an example, we take the network shown in Figure 1. Here, the coupled pair of lines are $b_1 = 9$ and $b_2 = 10$. Now let the dominant logic value be $\omega = 1$.

Figure 1. Test calculation for a bridging fault.

It is easy to see that a short circuit between two lines $b_1$ and $b_2$ manifests itself in a multiple-fault situation, involving two fault sites, namely $b_1$ and $b_2$. From the viewpoint of fault propagation, $b_1$ and $b_2$ are the starting elements of the potentially active lines. In our example, the united set of PAL’s is the following:

$$\text{PAL}(9–12) \cup \text{PAL}(10–12) = \{9, 10, 11, 12\}.$$  

The initial values are $v(9) = v(10) = d/1$, and $z_1 = v(12) = 0/1$.

The process of test calculation is as follows:

**Step 1:** $v(10) = 1/0$, $v(11) = d/0$ represent a contradiction.
$v(10) = d/0$, $v(11) = 1/0$ are also a contradiction

So, next we select the opposite composite value, i.e., $z_1 = v(12) = 1/0$.

**Step 2:** $v(10) = 0/1$, $v(11) = 0/d$.
Due to the determined normal component of $v(10)$,
$v(9) = \omega/\omega = 1/1$.

**Step 3:** $v(7) = 1$, $v(8) = d$.

**Step 4:** $v(8) = 1$. (Justification for gate 11.)

**Step 5:** $x_1 = 0$, $x_3 = d$.

**Step 6:** $x_3 = 0$, $x_4 = d$.

**Step 7:** $x_5 = 0$.

**Step 8:** $x_6 = d$.

Now, the found test vector is:

$$\overrightarrow{x_1} = (0, d, 0, d, 0, d).$$

It can be seen that in the fault-free network, $z_1 = 1$, while in the presence of the short circuit between lines 9 and 10, $z_1 = 0$ will occur, which is the criterion of fault detection.

**IV. CONCLUSIONS**

This paper has been meant for showing how the test calculation algorithm first published in [2] can be generalized for treating short-circuit (bridging) faults in digital circuit models. The efficiency of the composite justification is based on the fact that it establishes the minimal necessary and sufficient set of logic values which yield the test conditions for the faults. As seen, the tests are obtained by justifying the initial logic conditions. The salient advantage of composite justification is the total absence of the fault propagation phase. This feature makes the approach extremely flexible in
terms of circuit modeling and fault classes. The same applies to the use of an HDL. As known, the fault propagation phase, i.e., D-propagation is an inherent part of the wide-spread D-algorithm, where this phase implies serious difficulties for functional-level models and multiple faults. Functional algorithms for constructing computational tools of complex logic modules have been presented in [6]. This paper clearly illustrates the problems encountered in this topic.

When multiple faults are considered, the complete procedure of the D-algorithm has to be repeated $2^q - 1$ times in worst case, for one primary output, where q is the multiplicity of faults. In this case, attempts are made to propagate different combinations of the individual faults. On the other hand, composite justification requires only 2 iterations in worst case, also for one primary output.

An other promising approach is the use of Boolean proof engines, where the so-called Satisfiability (SAT) problem [7] is converted to the problem of test generation [8]. Here, a Boolean proof engine serves for solving Boolean equations, thus yielding tests for digital circuits. However, this approach requires having the Boolean descriptions, and is used only for single faults. It should also be added that the generation of Boolean functions is a necessary task of this approach, which has to be performed with knowledge of the gate-level description related to an actual circuit. This task is to be performed with knowledge of the circuit schematic, and it requires an extremely time consuming procedure.

As far as the computer implementation is concerned, only line justification is to be accomplished in the presented principle, which is also an advantage. In order to perform the line-value justification, the inverse models of the building elements in the network are required. An inverse model defines the set of possible input patterns which result in a specific state or an output pattern [6], [9]. For this purpose, high level hardware-description languages, such as VHDL can also be applied [10]-[11].

The computerized implementation of the proposed principle will handle multi-valued logic, where the number of logic values corresponds to the number of the possible signal values in the HDL applied for modeling.

Finally, as far as the generation of tests for short-circuit faults is concerned, all the advantages of composite justification apply also to solving that different computation task. The principle presented in this paper can also be extended to handling bridging faults in logic networks that are modeled at the functional level.

V. REFERENCES


