Test Pattern Generation for Path Delay Faults in Synchronous Sequential Circuits Using Multiple Fast Clocks and Multiple Observation Times

Prasanti Uppaluri, Irith Pomeranz and Sudhakar M. Reddy +
Electrical and Computer Engineering Department
University of Iowa
Iowa City, IA 52242
U.S.A

Abstract

In this paper, the problem of test generation for path delay faults in synchronous sequential circuits is addressed. In existing testing methods, a single fast clock cycle is used to activate path delay faults and a fault is said to be detected only if the fault free response is different from the faulty response at a single output and at a specified time unit in the test sequence. We refer to these methods as single fast clock cycle and single observation time testing methods. In this paper, we show that testable faults may exist, which are untestable using a single fast clock cycle and a single observation time. Such faults are testable when multiple fast clock cycles and/or multiple observation times are used. A test generation procedure is given that uses multiple fast clock cycles and multiple observation times. Experimental results are presented on MCNC synthesis benchmarks to demonstrate the effectiveness of the proposed strategy in increasing the fault coverage and reducing the test length.

1. Introduction

Correct operation of logic circuits requires proper logic and timing behaviour. Manufacturing defects and random variations in process parameters may cause propagation delays in the circuit to exceed their specification. Such defects are modeled as delay faults. Two delay fault models have been proposed in the literature : the gate delay fault model [11] and the path delay fault model [7]. Gate delay faults model defects that cause gate delays to be outside their specified range. Path delay faults model defects that cause cumulative propagation delays along circuit paths to exceed their specification. Two types of tests have been proposed : robust and non-robust [8,9].

This paper addresses the issue of test generation for path delay faults in synchronous sequential circuits. Several procedures for path delay fault testing in sequential circuits have been presented in [1-6]. With the exception of [5], all the procedures rely on the use of two clocks: (1) The normal system clock, also referred to as a *fast clock*, and (2) a clock slower than the normal system clock or a *slow clock*. The slow clock period is chosen to ensure that all signal values stabilize in the circuit within one clock period, even in the presence of delay faults. All approaches, with the exception of [5,6], use the following testing method, first proposed in [1].

Test application is done in three steps : initialization, fault activation and fault propagation. The goal of initialization is to bring the circuit to the appropriate initial state in which the fault can be activated. This is performed by applying an initializing sequence using a slow clock. Fault activation consists of a single clock cycle and it is performed using a fast clock. Depending on whether a delay fault exists or not in the circuit, signal-transitions begun at this time unit may or may not reach the primary outputs and/or flip-flops by the end of the fast clock cycle. At the end of this clock cycle, the effects of activated faults can be observed on the primary outputs and/or they are latched in the flip-flops. The goal of fault propagation is to bring fault effects latched in the flip-flops to the primary outputs. Fault propagation is performed by applying a propagation sequence using a slow clock.

In the testing strategy of [1-4], a fault is activated using a single fast clock cycle and it is said to be detected only if the fault free response is different from the faulty response at a single output and at a predetermined time unit along the test sequence. This strategy is referred to here as the single fast clock, single observation time strategy (SFCSO). It imposes two restrictions on the test sequence, that result in loss of fault coverage and increased test length, namely, the use of a single fast clock cycle and a single observation time. These restrictions are removed by the testing strategy proposed here, that uses *multiple fast clock cycles* and *multiple observation times* (MFCMO).

The use of multiple observation times was first proposed in [10] for detecting stuck-at faults in synchronous sequential circuits with no reset, or where reset may fail.

+ Research supported in part by NSF Grant No. MIP-9220549 and in part by NSF Grant No. MIP-9357581.

It was shown that significant improvements in fault coverage are possible using this approach, compared to the conventional, single observation time approach. In this work, we show that a similar improvement is possible in the coverage of path delay faults. For path delay faults, the multiple observation times approach is important even for circuits with a fault free reset mechanism, since uncertainty as to whether or not delay faults affect the values latched in the flip-flops gives rise to unspecified values [6]. The relevant material from [6] is reviewed and illustrated in Section 2.

The use of multiple fast clock cycles was first proposed in [5]. The method of [5] used fast clock cycles for initialization, activation and propagation, and required relaxation of the fault model (a fault must not increase the maximum delay in the circuit to more than the duration of two clock cycles). The combination of slow and fast clock cycles under which a test sequence is applied is called a *clocking scheme*. In [6], a fault simulator was presented, that can simulate test sequences under clocking schemes containing multiple fast clock cycles, without relaxing the fault model. It was shown in [6] that faults exist, that can only be detected by a given test sequence if it is applied using multiple fast clock cycles. In Section 3, we show the existence of faults that require multiple fast clock cycles under any test sequence. Thus, the use of clocking schemes with multiple fast clock cycles is necessary to ensure detection of all detectable faults.

Motivated by the observations above, we describe in this work a test generation procedure that takes advantage of both the multiple observation times approach and the use of multiple fast clock cycles. This procedure can detect faults that were previously undetectable under the SFCSO approach. We consider a simplified version of the general MFCMO procedure. We compare the results of this procedure with the results of a complete SFCSO procedure, and demonstrate improved fault coverage for benchmark circuits. We consider in this work circuits having compact state-table representations for the following reasons : (1) To ensure that the test generation procedures are complete, and (2) To be able to avoid the potential loss of precision that results in three-value simulation at the gate level. This allows us to isolate the effects on fault coverage of three value simulation, multiple observation times and multiple fast clocks. For the same reasons, the existence of a reset mechanism is also assumed. This assumption can be substituted with the assumption of the existence of an initializing sequence. Finally, only robust tests are considered in this work.

The paper is organized as follows. Section 2 reviews the value system from [6] that is used for robust fault activation and fault propagation. An introduction to the multiple observation times and multiple fast clock cycles strategy is given in Section 3. Section 4 presents a simplified test generation procedure using multiple observation times and multiple fast clock cycles. Experimental results on MCNC benchmark circuits are given in Section 5. Section 6 concludes the paper.

2. Preliminaries

In this section, we review the logic value system from [6] for evaluating the behaviour of a synchronous sequential circuit under an input sequence applied using slow and fast clock cycles.

Let $(t_1, t_2, \ldots, t_{i-1}$ S, t_i F, $t_{i+1}, \ldots, t_n)$ be a test sequence of length n. $(t_1, \ldots, t_{i-1})$ is the initialization sequence which brings the circuit from the reset state to the desired state for fault activation. The initializing sequence is applied using a slow clock. Here, S is used as a modifier to distinguish the last initialization input, t_{i-1}, from the others. t_i is the activation input and it is applied using a fast clock. An F is attached to this input to indicate that the application of this input is done using a fast clock. Finally, $(t_{i+1}, \ldots, t_n)$ is the fault propagation sequence which is applied using a slow clock.

We start by illustrating the computation of values throughout a circuit before, during and after a fast clock is applied. To present the examples, we use both the gate level and the state table representation of a synchronous sequential circuit.

Table 1: State table

PS	NS,z x=0	NS,z x=1
00	01,1	10,0
01	10,0	11,1
10	11,0	00,1
11	00,1	01,0

Consider the input sequence (0S, 0F, 0) applied to the circuit given in Fig. 1. Table 1 gives its state table and Fig. 1 gives the gate level implementation of the circuit. Assume that reset is applied before the input sequence. Let the reset state be 00. Under the given test sequence, 0 is applied during time units 1 and 3 using a slow clock. During time unit 2, a 0 input is applied using a fast clock. Table 2 summarizes the simulation for the first two time units. Using the conventional three-valued fault free logic simulation rules for evaluating the output in time unit 1 , we get $Y_1Y_2 = 01$ and $Z = 1$. This can alternatively be seen from the state table. At time unit 2, on the application of a 0 input using a fast clock, primary input i remains at 0, present state y_1 remains at 0, and present state y_2 changes from 0 to 1 (represented as a $0 \rightarrow 1$ transition).

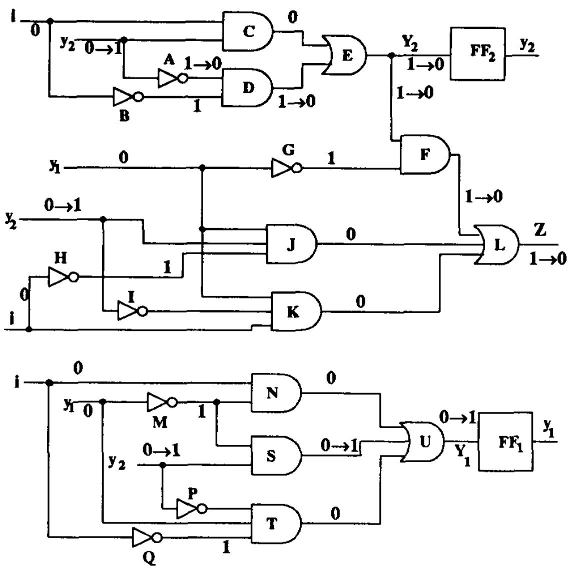

Fig. 1: Gate level implementation

Simulation of the circuit under these values is shown in Fig. 1. From Fig. 1, we find that the slow-to-rise fault along path $P_3 = (y_2ADEFLZ)$ is tested at the primary output Z, the slow-to-rise fault along $P_1 = (y_2SUY_1)$ is activated on next state line Y_1, and the slow-to-rise fault along $P_2 = (y_2ADEY_2)$ is activated on next state line Y_2. Associated with each next state line and primary output we have the set of paths which were robustly propagated to the line in the time unit considered. For simplicity, we denote the paths activated on next state line Y_1 and Y_2 by P_1 and P_2, respectively, and the path tested on primary output Z by P_3 (cf. Table 2).

Table 2: Simulating (0S, 0F) for Fig. 1

cycle	time	PI	PS		NS		PO
		i	y_1	y_2	Y_1	Y_2	Z
slow	1	0	0	0	0	1	1
fast	2	0	0	0→1	0→1 (P_1)	1→0(P_2)	1→0 (P_3)

The values latched by the flip-flops at the end of time unit 2 are denoted α / β, where α is the value in the fault free circuit and β is the value in the faulty circuit. They are computed as follows [6]. If the circuit is fault free, FF_1 and FF_2 will have the fault free values of 1 and 0 respectively. To compute the faulty values, we need to distinguish between the different sets of activated paths. Let us start with P_1. If P_1 is faulty, FF_1 will have the faulty value of 0. The value latched in FF_1 for the purposes of propagating P_1 is therefore 1/0. Since P_1 does not end at FF_2, FF_2 can latch the value 1 if P_2 is also faulty or the value 0 if P_2 is not faulty. An input sequence is a robust test for the path P_1 only if it detects the fault independent of other path delay faults which may be present in the circuit. Hence, we cannot assume any value for FF_2. Thus, FF_2 is set to unknown (x) in the faulty machine and the value it latches for propagating P_1 is 0/x. With P_2 as the fault under consideration, the value latched in Y_1 and Y_2 in time unit 2 can be computed similarly. The results are 1/x and 0/1, respectively. The simulation at time unit 3 for both paths P_1 and P_2 is given in Table 3. Under a 0 input in time unit 3, the fault effect due to path P_1 disappears. In the case of P_2, the fault effect does not propagate to the primary output, however, it does propagate to FF_2.

We have seen that when considering a fault f on a path P, there is uncertainty about the values latched by the flip-flops which are not at the end of P. Uncertainty also arises when hazardous signals are propagated to the flip-flops. This happens because no assumption can be made about the delays of other lines in the circuit. Hence, when a fault f is activated, the faulty machine may be in one of many possible states, each corresponding to different delays present in the machine.

Table 3: Simulation for time unit 3

path	cycle	time	PI	PS		NS		PO
			i	y_1	y_2	Y_1	Y_2	Z
P_1	slow	3	0	1/0 (P_1)	0/x	1/x	1/x	0/x
P_2	slow	3	0	1/x	0/1 (P_2)	1/x	1/0(P_2)	0/x

The value system of [6] allows us to consider test sequences with multiple fast clock cycles. Consider a flip-flop with present state value, at time i, $y(i) = 0/1$ and next state value $Y(i) = 0$. Suppose that a fast clock is applied at time $i + 1$. In the fault free machine, $y(i) = 0$ and $Y(i) = 0$ result in $y(i + 1) = 0$. In the faulty machine, $y(i) = 1$ and $Y(i) = 0$ create a transition $y(i + 1) = 1 \rightarrow 0$. The value of y at time $i + 1$ is thus $y(i + 1) = 0/1 \rightarrow 0$. The complete value system and implication rules can be found in [6]. For the purposes of this work, we only need the values and rules demonstrated by the example above.

3. Multiple observation times and multiple fast clock cycles

In this section, we present examples to illustrate how multiple observation times and multiple fast clock cycles help to find tests for faults declared untestable by the SFCSO strategy. We present a simplified MFCMO test generation procedure and prove its completeness for strongly connected machines with a reset mechanism. Finally, we present a general MFCMO test generation procedure that guarantees complete fault coverage for any machine.

The slow-to-rise fault on path P_1 in the example described in Section 2 belongs to the set of undetectable faults under the SFCSO strategy if three value logic simulation is used. However, it is detectable under the multiple observation times strategy. We show these claims next.

In order to activate the slow-to-rise fault on $P_1 = (y_2SUY_1)$, i and y_1 require a steady zero value (to ensure a steady zero on off path signal lines) and a 0→1 transition is needed on y_2. Thus, we need a transition from state 00 to state 01 under a 0 input using a slow clock followed by an input 0 under a fast clock. Hence, the input sequence that activates the fault is (t_{i-1} S, t_i F) = (0S, 0F), and the machine has to be in state 00 before the application of t_{i-1}. There is no need for an initialization sequence, since 00 is the reset state. Hence, ($t_1, t_2, \ldots, t_{i-2}$) is empty. However, a propagation sequence is necessary to test this path. The SFCSO approach would try to generate it in the following manner.

It would try the two possible input combinations, 0 and 1, at time unit 3. For a 0 input, the fault effect disappears. Hence, the propagation sequence cannot start with a 0. The simulation of the input sequence (0S, 0F, 0) was considered in Section 2.

Table 4: Simulating (0S, 0F, 1)

cycle	time	PI	PS		NS		PO
		i	y_1	y_2	Y_1	Y_2	Z
slow	1	0	0	0	0	1	1
fast	2	0	0	0→1	0→1 (P_1)	1→0	1→0
slow	3	1	1/0 (P_1)	0/x	0/1 (P_1)	0/x	1/x

For a 1 input at time unit 3, the result is given in Table 4. The fault effect does not propagate to the primary output. However, it propagates to a next state line. Again, there is a choice of two input combinations for time unit 4. The outcome of the two choices is given in Table 5. The fault effect disappears on the application of a 0 input. Under a 1 input, the fault effect is present and it is identical to that we latched when we started the search for a propagation sequence in time unit 3. Hence, the SFCSO strategy would declare this fault as untestable.

Table 5: Simulation for time unit 4

cycle	time	PI	PS		NS		PO
		i	y_1	y_2	Y_1	Y_2	Z
slow	4	0	0/1 (P_1)	0/x	0/x	1/x	1/x
slow	4	1	0/1 (P_1)	0/x	1/0 (P_1)	0/x	0/x

Let us reconsider the fault effects for P_1 after activation, given in Table 3. The fault free machine is in state 10 and the faulty machine state is 0x, i.e., the possible faulty states are {00,01}. If we now consider the propagation sequence (0, 0) for each one of the faulty states separately, we obtain the values given in Tables 6 and 7. A single observation of the response at time unit 4 detects the fault regardless of the state after activation.

Table 6: Simulating (0S, 0F, 0, 0) for faulty state 00

cycle	time	PI	PS		NS		PO
		i	y_1	y_2	Y_1	Y_2	Z
slow	1	0	0	0	0	1	1
fast	2	0	0	0→1	0→1 (P_1)	1→0	1→0
slow	3	0	1/0	0/0	1/0	1/1	0/1
slow	4	0	1/0	1/1	0/1	0/0	1/0

Table 7: Simulating (0S, 0F, 0, 0) for faulty state 01

cycle	time	PI	PS		NS		PO
		i	y_1	y_2	Y_1	Y_2	Z
slow	1	0	0	0	0	1	1
fast	2	0	0	0→1	0→1 (P_1)	1→0	1→0
slow	3	0	1/0	0/1	1	1/0	0
slow	4	0	1	1/0	0/1	0/1	1/0

It is interesting to note here that the fault was declared untestable under the SFCSO strategy when three value simulation was used. If the faulty states are carried over accurately for each time unit, (0S, 0F, 0, 0) is a test sequence under the SFCSO strategy too. Thus, accurate simulation with all possible faulty states after fault activation could correctly identify detection of a fault, whereas loss of precision due to three value logic simulation may not identify this. Next we give an example of a fault that can be detected using the multiple observation times strategy, but not if the SFCSO strategy is used (even with accurate simulation that avoids loss of precision in three value logic simulation).

Table 8 gives the state table of the circuit and the gate level implementation for Y_3 is given in Fig. 2. The implementation used is the following :

$Y_1 = y_1 + y_2y_3 + iy'_2$

$Y_2 = y_2 + y_1y'_3 + iy'_3 + y'_1y_3i'$

$Y_3 = y_1y_2y'_3 + y'_1y'_2y'_3i' + y'_2y_3i + y_1y'_2y_3 + y'_1y_2y_3i'$

$Z = y_2y'_1 + y_1y'_2y_3 + y_3y'_2i$

Here, y'_i is the complement of y_i.

The slow-to-rise fault along path (y_3CGJY_3), denoted as P_3, has only one initialization and activation sequence (t_{i-1} S, t_i F) = (0S, 0F) and the machine has to be in the reset state 000 before its application. The simulation results under this sequence are tabulated in Table 9. With P_3 as the fault under consideration, the fault free state is 010 and the set of possible faulty states are {001, 011}. It

Table 8: Effectiveness of multiple observation times

PS	NS,z x=0	NS,z x=1
000	001,0	110,0
001	010,0	101,1
010	010,1	010,1
011	111,1	110,1
100	110,0	110,0
101	101,1	101,1
110	111,0	111,0
111	110,0	110,0

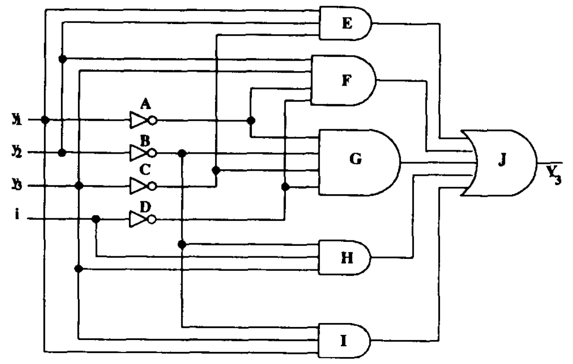

Fig. 2: Gate level implementation for Y_3

can be seen from the state table that the only possible output response for the fault free state 010, for any input sequence starting at time unit 3, is 1111... and for the faulty state 011 it is 10000... . Hence, no matter what the input sequence, these two states are distinguished at time units 4, 5, 6 and so on. The faulty state 001, on the other hand, has two possible output responses 01111... and 1111... under input sequences 0*xxx*... and 1*xxx*..., respectively, where *x* stands for any input value. The sequences 1*xxx*... do not distinguish 001 from the fault free state because their output responses under these sequences are identical to the fault free output sequence. Under the sequences 0*xxx*..., the faulty state 001 is distinguished from the fault free state only at time unit 3 (when the first 0 is applied). Thus, we find that no sequence can be constructed that will distinguish both faulty states from the fault free state at the same time unit. Hence, the fault is untestable under the SFCSO strategy, even under accurate simulation. We can, however, construct a test sequence for P_3 under the multiple observation times strategy and it is (0S, 0F, 0, 0).

To summarize the advantages of the multiple observation times approach in this context, note that the SFCSO strategy tries to generate a propagation sequence by searching for a single sequence which will distinguish the fault free state from all the possible faulty states by setting a specific output at a specific time to D or D'. Under the multiple observation times approach, a single sequence is required, which will distinguish the fault free state from every faulty state. However, the faulty states do not have to be distinguished from the fault free state in the same time unit or on the same output.

Table 9: Simulating (0S, 0F) for Fig. 2.

cycle	time	PI	PS			NS			PO
		i	y_1	y_2	y_3	Y_1	Y_2	Y_3	Z
slow	1	0	0	0	0	0	0	1	0
fast	2	0	0	0	0→1	0	0→1 (P_2)	1→0 (P_3)	0

An example is now presented to illustrate how multiple fast clock cycles with multiple observation times detect a fault declared untestable by the multiple observation times approach alone and hence by the SFCSO strategy. Consider the circuit represented by the state table in Table 10. The gate level representation only for Y_1 is given in Fig. 3. The implementation used is the following.

$Y_1 = y_2 + y'_3 i + y_1 y_3 i' + y'_1 i + y'_1 y'_3$
$Y_2 = y_2 + y_1 i' + y_1 y'_3 + y'_1 y_3 i$
$Y_3 = y_2 y'_3 + y'_1 y'_2 i + y'_1 y_2 i' + y_1 y'_2 i' + y_1 y'_2 y_3$
$Z = y_2 y'_3 + y'_1 y'_2 y_3 + y_1 y'_2 i' + y'_1 y_2 i'$

The slow-to-rise fault along path $(y_1 BFHY_1)$, denoted as P_1, has only one initialization and activation sequence $(t_{i-1}$ S, t_i F) = (0S, 0F) and the machine has to be in the reset state 000. Let P_2, P_3 be the sets of paths ending at flip-flops FF_2 and FF_3 and robustly activated at time unit 2 of this sequence. Let P_4 be the set of paths detected on Z at time unit 2. The outcome of the application of this sequence is given in Table 11.

Table 10: Effectiveness of multiple fast clock cycles

PS	NS,z x=0	NS,z x=1
000	100,0	101,0
001	000,1	111,1
011	111,1	110,0
100	011,1	110,0
101	111,1	001,0
110	111,1	111,1
111	110,0	110,0

The fault free state at time unit 3 is 011 and the faulty state for P_1 is 1xx, i.e., {100,101,110,111}. The multiple observation times strategy would try to distinguish all these states from the fault free state with a single distinguishing sequence. However, one does not exist. Multiple

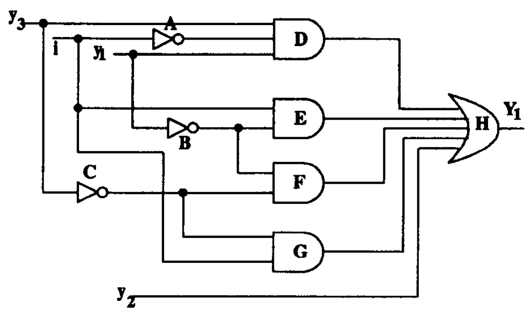

Fig. 3: Gate level implementation for Y_1

Table 11: Simulating (0S, 0F) for Fig. 3.

cycle	time	PI	PS	NS			PO
		i	y_1 $y_2 y_3$	Y_1	Y_2	Y_3	Z
slow	1	0	0 0 0	1	0	0	0
fast	2	0	$0 \rightarrow 1$ 0 0	$1 \rightarrow 0 (P_1)$	$0 \rightarrow 1 (P_2)$	$0 \rightarrow 1 (P_3)$	$0 \rightarrow 1 (P_4)$

observation times with multiple fast clock cycles is necessary to find a test for the fault under consideration.

Multiple fast clock cycles are employed as follows. The sequence (1,0,0,1) distinguishes faulty states (101,110) from the fault free state and the sequence (0,0) distinguishes states (100,111) from the fault free state. Combining the initialization and activation sequences obtained earlier with these propagation sequences we obtain two tests $TEST_1$ = (0S, 0F, 1, 0, 0, 1), $TEST_2$ = (0S, 0F, 0, 0). These two tests can be used to test P_1 in the following manner. Initially, reset the machine to the all zero state. Apply $TEST_1$. If the state reached after (0S, 0F) is 101 or 110, the fault will be detected. Otherwise, let the state reached be S. Reset the machine again and apply $TEST_2$. State S is reached again after the application of (0S, 0F). If P_1 is faulty, S must be one of {100, 111} and $TEST_2$ will detect the fault. Otherwise, we conclude that the machine is free of the fault P_1. Thus, if the path P_1 is faulty, the response to either $TEST_1$ or $TEST_2$ will be faulty. The procedure employed in the previous example is a special case of the general MFCMO procedure. It is called the restricted MFCMO procedure and denoted rMFCMO. The rMFCMO procedure is described next.

Let the initialization and activation sequence for the fault under consideration be T_1. T_1 contains a single fast clock cycle. Let the state of the faulty machine after fault activation be one of $\{s_1, s_2, ..., s_k\}$ and let the fault free state be s_0. Suppose that it is possible to find distinguishing sequence $D_1, \ldots, D_k$, such that D_j is a distinguishing sequence for s_0 and s_j. A test sequence is constructed as follows.

Apply reset followed by T_1 and D_1. This will detect the fault if the machine reaches state s_1 after T_1. If not, we apply reset followed by T_1 and D_2. The state reached after the application of reset and T_1 is identical to the state reached the first time T_1 was applied. If this state is s_2, D_2 will detect the fault. Otherwise, we apply reset, T_1 and D_3. We continue to apply reset, T_1 and D_4 and so on. If the fault is present and the machine reaches state s_i after applying reset and T_1, then it will be detected when reset, T_1 and D_i are applied.

In practice, instead of using a distinguishing sequence D_i for every state s_i, we search for a minimal number of distinguishing sequences such that every faulty state is distinguished by at least one of the distinguishing sequences from the fault free state reached after T_1 is applied.

Instead of applying reset between every two test sequences, it is also possible to use a transfer sequence from the final state reached back to the reset state. Alternatively, the final state reached can be used as a starting state for the next test sequence.

The procedure rMFCMO constructs test sequences of the form $T_1 D_1 T_1 D_2 ... T_1 D_k$, where the fast clock is applied with the last symbol of T_1 and reset (or an initializing sequence) is applied before the first symbol of T_1. A general MFCMO test sequence has a fast clock at arbitrary places along the test sequence and contains arbitrary input vectors. The following definition describes such a test sequence.

Definition 1 : A fault f is said to be *detectable* if there exists an input sequence T and a clocking scheme C such that the response produced by the fault free machine to T applied under C is different from the response produced by the faulty machine containing f, independent of other delays in the circuit (by "independent of other delays" we mean that the test is to detect the fault without specific knowledge about the extra delays along paths other than the one for which the test is being derived. This is inspite of the fact that under the multiple observation times approach we distinguish among different faulty states reached upon fault activation).

Definition 2 : A fault is said to be *undetectable* if it is not detectable.

Let F_{MFCMO} be the set of faults detectable by the general MFCMO strategy. Let F_{rMFCMO} be the set of faults detectable by the restricted MFCMO strategy.

Theorem 1 : $F_{rMFCMO} \subseteq F_{MFCMO}$.

Proof: If $f \in F_{rMFCMO}$, then the rMFCMO procedure can be used to construct a test sequence T and a clocking scheme C that detects f. Hence, $f \in F_{MFCMO}$. This is true for every $f \in F_{rMFCMO}$. Thus, $F_{rMFCMO} \subseteq F_{MFCMO}$. □

We now prove that for strongly connected machines with a reset mechanism, the rMFCMO strategy detects all the faults in the circuit detectable by the general MFCMO procedure (cf. Definition 1).

Let $A_k = (t_{k-1}\ \mathrm{S},\ t_k\ \mathrm{F})$ (cf. Section 2) be a sequence that activates the fault f under consideration when the machine is in state S_k. f is said to be activated in S_k by A_k.

Theorem 2 : If the sequential machine is strongly connected and it has a reset mechanism, $F_{MFCMO} \subseteq F_{rMFCMO}$.

Proof: Consider a fault $f \notin F_{rMFCMO}$. f may be undetectable under the restricted MFCMO strategy because of one of two reasons.

CASE I : The fault cannot be robustly activated.

If this condition is true, no activation sequence can be found for fault f under the general MFCMO strategy and this implies $f \notin F_{MFCMO}$.

CASE II : The fault can be robustly activated, however, cannot be detected under the rMFCMO strategy. In this case, the path corresponding to f cannot end at a primary output. To show this, assume that the path ends at primary output i and that A_k is an activation sequence from state S_k. Then by applying reset, a transfer sequence from the reset state to S_k and then applying A_k the fault can be detected at primary output i.

Let us consider the sequences A_k that activate f. For every sequence A_k that activates f, there exists a faulty state equivalent to the fault free state (otherwise, distinguishing sequences can be found for the fault free state and every faulty state reached after activation, and a test sequence can be constructed by the rMFCMO procedure). Consider an arbitrary activation sequence A_1. Let I_1 be the corresponding initialization sequence. I_1 exists since the machine is strongly connected and it has a reset mechanism. On activation of the fault f using A_1, let the set of faulty states be $\{S_1^f, S_2^f, \ldots, S_i^f\}$ and let the fault free machine be in S. At least one of the faulty states is equivalent to the fault free state. Let $S_1^f \equiv S$. Since $S_1^f \equiv S$, we cannot find a pairwise distinguishing sequence for them using only slow clocks. We have to show that there does not exist a distinguishing sequence using a clocking scheme containing fast clock cycles.

In any arbitrary clocking scheme C, a fast clock cycle can be of one of two types :

Case i : A fast clock cycle that does not reactivate f. (The fault effects may reach a primary output for some circuit delays, however, due to the independence assumption, the fault is not detected for all circuit delays. Therefore, such a fast clock cycle cannot detect f).

Since f is not reactivated, the values latched in the flip-flops for the purposes of propagating f are either of the form α/α or of the form α/x (cf. Section 2). The values to the right of the / represent the set of faulty states and the value to the left represents the state that would be reached if a slow clock were applied. Let us denote this state by S'. We find that S' is contained in the set of faulty states. Thus, after the application of this kind of a fast clock cycle, there again exists one faulty machine state indistinguishable from the fault free state.

Case ii : A fast clock cycle that reactivates f.

The following part of the proof is also illustrated in Fig. 4. Let the faulty machine state be R and the fault free machine state be S^* before reactivation of the fault. Let the possible set of faulty states on reactivation of f be $\{R_1^f, R_2^f, \ldots, R_j^f\}$ and let the fault free machine state be S^{**}. Let the faulty machine state be R^* if the fast clock cycle had been a slow clock. Since the fault cannot be detected using only slow clocks, we know that :

$$R^* \equiv S^{**} \quad \text{---------}\ (1)$$

The fault was activated in R in the faulty machine. This could have been done also by using a transfer sequence from reset state to R and then applying the activation sequence A_k which activates the fault in R. Also, we know that for every A_k, there exists at least one faulty state equivalent to the fault free state for that activation. In this case, suppose that it is the faulty state R_1^f. We have,

$$R_1^f \equiv R^*. \quad \text{--------}\ (2)$$

From (1) and (2), we find that

$$S^{**} \equiv R_1^f.$$

Hence, we find that on a reactivation of the fault also there exists at least one faulty machine state equivalent to the fault free machine state.

From Cases i and ii, we find that fast clock cycles cannot be used to construct a distinguishing sequence for S_1^f and S. Hence, $f \notin F_{MFCMO}$.

From CASE I and CASE II, we find that $f \notin F_{rMFCMO}$ implies $f \notin F_{MFCMO}$. This implies, $F_{MFCMO} \subseteq F_{rMFCMO}$ for a strongly connected machine with a reset mechanism. □

FAULT FREE MACHINE

$$\begin{tikzcd}
 & & S \arrow[r, dashed] & S^{*} \arrow[rr] & & S^{**} \\
\text{Reset} \arrow[r, dashed, "I_1 A_1"] & {} \arrow[ur, "\text{fault free}"] \arrow[dr, "\text{faulty}"'] & & & & R^{*} \\
 & & S_1^f \arrow[r, dashed] & R \arrow[urr, "\text{slow}"] \arrow[drr, "\text{fast}"'] & & \\
 & & & & & R_1^f, R_2^f, \ldots, R_j^f
\end{tikzcd}$$

FAULTY MACHINE

$$S_1^f \equiv S \qquad R \equiv S^{*} \qquad R^{*} \equiv S^{**}, R^{*} \equiv R_1^f$$

Fig. 4: Illustration of the proof

From Theorems 1 and 2, we have the following result.

Theorem 3 : If the machine is strongly connected and it has a reset mechanism, then $F_{rMFCMO} = F_{MFCMO}$.

It should be noted that the assumption on reset can be relaxed to the assumption of the existence of a known synchronizing sequence without affecting the statements of Theorems 1-3.

4. Test generation

In this section, we describe a test generation procedure based on multiple observation times and multiple fast clock cycles. To test the efficacy of the proposed procedure, an SFCSO test generator has also been implemented. The procedures are given for sequential circuits that have compact state-table representations. Application to large circuits is currently under investigation.

The proposed rMFCMO test generation procedure is divided into two phases. The first phase involves test generation with multiple observation times. The second phase also uses multiple fast clock cycles. An attempt is made to generate a single test sequence, with a reset applied at the beginning of the test sequence. The data structure used is explained below.

Let I and J be input combinations. For every transition from state S_i to state S_j under I in the state table, we simulate each combination (I S, J F) with the machine in state S_i and collect the following information : *Pdet* is the list of faults detected at the primary outputs; *Pact* is the list of faults activated at the next state lines. This information is stored in a structure called DAS (Detection, Activation Structure). DAS = $\{I, J, Pdet, Pact, flag\}$ with flag initially set to zero. All the DASs for a state transition are stored in a linked list.

The *bestedge* from S_i is defined as the edge having a DAS with the maximum number of faults detected. If there is more than one, it is the edge with a DAS having the larger number of faults activated. If, again, there is more than one choice, the *bestedge* is chosen arbitrarily from the available choices. The chosen DAS is referred to as DAS_c.

The test generation algorithm with multiple observation times is given in Procedure 1. The multiple fast clock cycles strategy is used to improve fault coverage after the completion of the first phase of test generation with multiple observation times.

Procedure 1: Proposed test generation procedure with multiple observation times.

1. Obtain the DASs for every $S_i \rightarrow S_j$ in the state table.
2. Set *cur_st* = *reset state.*
3. Pick the *bestedge* from *cur_st*. If no such edge exists (i.e. if DAS_c has $Pdet = 0$ and $Pact = 0$ or if it has flag set to 1) :
    4. Pick the *bestedge* from the states that can be reached from the *cur_st* with the application of a single input. If no such edge exists :
        5. Pick the *bestedge* from the entire state table. If no such edge exists, stop.
6. Let the *bestedge* be from *next_cur_st*. Determine the shortest input sequence to go from *cur_st* to *next_cur_st*. Call it *inter_seq*. Set *cur_st* = *next_cur_st*.
7. Simulate the input sequence (I S, J F) with the machine in *cur_st*. Here, I and J are obtained from DAS_c. Determine the propagation sequence for the faults activated (discussed below).
8. Update the path information (*Pdet*, *Pact*) in all DASs by removing the paths detected, if any, from them.
9. (The faulty and the fault free machines are brought to the same state by the propagation procedure.) Set *cur_state* = fault free state reached. Goto step 3.

The propagation procedure is discussed in detail next. Consider a sequential circuit with three flip-flops. On the simulation of a particular activation sequence (I S, J F) with the machine in state S_i, let path delay faults P_1 and P_2 be activated on flip-flop 1 and let P_3 and P_4 be activated, respectively, on flip-flop 2 and 3. Let the set of possible faulty states associated with P_1 and P_2 be $faulty_1$, with P_3 be $faulty_2$ and with P_4 be $faulty_3$. In order to determine the propagation sequence, it does not matter which path delay fault caused a particular faulty state. If we can find a single sequence to distinguish the set $FAULTY = faulty_1 \cup faulty_2 \cup faulty_3$ from the fault free state, we have found a test for P_1, P_2, P_3 and P_4. To clarify this, consider a specific case where the three flip-flops, respectively, have the following latched effects : 1/0 (P_1, P_2), 1/0 (P_3), 1/0 (P_4). The fault free machine state is 111. The faulty machine state to be considered while generating a test for P_1 and P_2 is 0xx, i.e., the set {000,001,010,011}, for path P_3 it is x0x, i.e., the set {000,001,100,101} and for path P_4, it is xx0, i.e., the set {000,010,100,110}. If we can find a single propagation sequence to distinguish the union of all these faulty states from the fault free state, then we have found a test for P_1, P_2, P_3 and P_4. It is a test, because no matter what the faulty state, the faulty machine behaviour is different from the fault free machine in some time unit along the test sequence.

If such a propagation sequence cannot be found, we try to find a propagation sequence for the delay faults activated at a single flip-flop, denoted as P_{one}. An attempt is made to find a sequence to distinguish the set of faulty states associated with P_{one} from the fault free state. If such a sequence exists and if any path $\in P_{one}$ is faulty, the test sequence will detect it. In the example described above, assume a single distinguishing sequence does not exist. An attempt is then made to propagate P_1 and P_2 only. Let D_1 be a sequence which distinguishes $faulty_1$ = {000, 001, 010, 011} from the fault free state 111. Let T_1 be the initialization and activation sequence. Then (T_1, D_1) is applied to the machine under test. If D_1 does not exist, the flag in DAS_c is set to 1. Only these edges will be considered in the rMFCMO phase of test generation.

To continue test generation for the other faults, it is more convenient if the faulty and the fault free machines are in the same state. Continuing with the above example, assume that only path P_3 is faulty and the others are fault free. Hence, on the application of T_1, the faulty machine is in state 101. If D_1 distinguishes state 101 from the fault free state 111, (T_1, D_1) is a test for P_3 too. Otherwise, (T_1, D_1) is not a test. However, we can not conclude that the faulty and fault free machines are in the same state. In this case, we extend the test sequence to ensure that. We find a sequence M_1 such that all the possible faulty machine states $\in faulty_1$ are either in the same state as the fault free machine or are distinguished from it along M_1.

The only difference in the implementations of the test generator with multiple observation times and the SFCSO test generator (with accurate simulation) is in the fault propagation procedure. For the SFCSO test generator, an input sequence is a propagation sequence only if the set of possible faulty states are distinguished from the fault free state in the same time unit and at the same output.

In the multiple fast clock cycles strategy, the propagation procedure involves the generation of minimal input sequences to distinguish each faulty state from the fault free state. The test is constructed as in the rMFCMO procedure described in Section 3.

The SFCSO test generator with three value simulation is described next. For every state transition $S_i \rightarrow S_j$ under I in the state table, we simulate each input combination (I S, J F) with the machine in state S_i. Let *init_seq* be an input sequence from the reset state to S_i. If a path P is detected at a primary output, then (*init_seq*, I S, J F) is a test for P. For a path P activated at a next state line, we try to find a sequence to distinguish the faulty machine state from the fault free state using three value simulation. In the example above, to test P_1 and P_2, we would find a sequence, say *prop*, to distinguish 0xx from the fault free state 111 by simulation. If *prop* exists, (*init_seq*, I S, J F, *prop*) is a test for P_1 and P_2. This test is then simulated to see if other path delay faults are detected.

5. Experimental Results

The SFCSO test generator and the proposed test generation procedure were implemented in the *C* programming language. The program was run on MCNC benchmark circuits. State assignment was done using NOVA [12] and the circuits were optimized using the two-level and multi-level optimizers ESPRESSO [13] and MIS [14].

The proposed test generation procedure is designed so that there is no need to generate an initialization sequence for any path delay fault test except may be for the first one detected. When a test is found for a path delay fault, the state reached on the application of the test found is used for continuing test generation for path delay faults which are yet undetected [15]. Experimentally we found that we were always able to generate a single sequence with a reset applied only at the beginning of the test sequence. In this way, short test lengths can be obtained.

In Table 12, columns *pi*, *po* and *ff* give the number of primary inputs, primary outputs and flip-flops in the circuit. The number of path delay faults is given next. We then report the results of the following strategies : the SFCSO strategy using three value simulation, the SFCSO strategy using accurate simulation, the multiple observation times approach, and the rMFCMO approach. The number of faults detected is given under "detd". The fault coverage is given in parentheses. Under the "Len" column, the test length obtained is given. The time taken in seconds for each strategy on a SUN Sparc2 workstation is given under "time". Results are given for the rMFCMO strategy only when there was an improvement in the fault coverage. The number of faults detected increases significantly as the accuracy of the simulation procedure is increased. For example, the number of faults detected for *dk*16 by the accurate simulation procedure and the multiple observation times strategy is, respectively, 2 and 2.2 times that of the SFCSO strategy using three value simulation. The test length for the same circuit reduced by about 12% by the multiple observation times approach over the SFCSO strategy with accurate simulation.

The reduction in test length for the multiple observation times procedure results mainly because it is easier to derive a single test to detect all path delay faults activated in a particular time unit than it is for the SFCSO strategy.

The improvement in fault coverage goes upto 15% with added precision over the SFCSO strategy with three value simulation. The reduction in test length for the test generator with multiple observation times over the SFCSO strategy with accurate simulation is upto 53%.

6. Conclusions

We considered the problem of test generation for path delay faults in synchronous sequential circuits. We illustrated how multiple fast clock cycles and multiple observation times can be used to detect faults which are considered undetectable by the conventional single fast clock cycle, single observation time approach. A test generation procedure using multiple fast clock cycles and multiple observation times was presented. Experimental results on MCNC benchmarks were presented to demonstrate that significant improvements in fault coverage and reductions in test length are achievable. Future work will concentrate on extending these concepts to large circuits.

References

[1] Y.K. Malaiya and R. Narayanaswamy, "Modeling and Testing for Timing Faults in Synchronous Sequential Circuits", IEEE Design and Test of Computers, Nov. 1984, pp. 62-74.

[2] S. Devadas, "Delay Test Generation for Synchronous Sequential Circuits", 1989 Intl. Test Conf., pp. 144-152.

[3] P. Agrawal, V.D. Agrawal, S.C. Seth, "A New Method for Generating Tests for Delay Faults in Non-Scan Circuits", in Proc. 5th Intl. Conf. VLSI Design, pp. 4-11, Jan 1992.

[4] T.J. Chakraborty, V.D. Agrawal and M.L. Bushnell, "Delay Fault Models and Test Generation for Random Logic Sequential Circuits", in 1992 Design Automat. Conf., pp. 165-172.

[5] I. Pomeranz and S.M. Reddy, "At-speed Delay Testing of Synchronous Sequential Circuits," in ACM/IEEE Design Automat. Conf., pp. 177-181, 1992.

[6] I. Pomeranz, L.N. Reddy, and S.M. Reddy, "SPADES : A Simulator for Path Delay Faults in Sequential Circuits," in European Design Automat. Conf., pp. 428-435, 1992 ; also "SPADES-ACE : A Simulator for Path Delay Faults in Sequential Circuits with Extensions to Arbitrary Clocking Schemes," in IEEE trans. on CAD, pp. 251-263, Feb 1994.

[7] G.L. Smith, "Model for Delay Faults Based Upon Paths," in Proc. Intl. Test Conf., pp. 342-349, 1985.

[8] S.M. Reddy, M.K. Reddy, and V.D. Agrawal, "Robust Tests for Stuck-Open Faults in CMOS Combinational Logic Circuits," in Intl. Symposium on Fault-Tolerant Computing, pp. 44-49, 1984.

[9] E.S. Park and M.R. Mercer, "Robust and Nonrobust Tests for Path Delay Faults in Combinational Circuits," in Proc. Intl. Test Conf., pp. 1027-1034, 1987.

[10] I. Pomeranz and S.M. Reddy, "Test Generation for Synchronous Sequential Circuits using Multiple Observation Times," in Proc. Fault-Tolerant Computing Symposium, pp. 52-59, 1991.

[11] J.L. Carter, V.S. Iyengar, and B.K. Rosen, "Efficient Test Coverage Determination for Delay Faults," in Proc. Intl. Test Conf., pp. 418-427, 1987.

[12] T. Villa and A. Sangiovanni-Vincentelli, "NOVA : State assignment of finite state machines for optimal two-level logic implementations," in IEEE trans. on CAD, pp. 905-924, Sept. 1990.

[13] R.K. Brayton, G.D. Hachtel, C. McMullen, and A.L. Sangiovanni-Vincentelli, "Logic Minimization Algorithms for VLSI Synthesis," Kluwer Academic Publishers, Boston, MA 1984.

[14] R.K. Brayton, R. Rudell, A.L. Sangiovanni-Vincentelli, and A.R. Wang, "MIS : A Multiple-Level Logic Optimization System," in IEEE trans. on CAD, Vol. 6, pp. 1062-1081, Nov. 1987.

[15] I. Pomeranz and S.M. Reddy, "On Achieving a Complete Fault Coverage for Sequential Machines Using the Transition Fault Model", in 1991 Design Automat. Conf., pp. 341-346.

Table 12: Results for various strategies

circuit	pi	po	ff	faults	SFCSO			SFCSO			Multiple Obs			Multiple Fast		
					3-value sim.			accurate sim.								
					detd	Len	time	detd	Len	time	detd	Len	time	detd	Len	time
bbara	4	2	4	478	51(10.67)	244	2.6	55(11.51)	137	2.2	55(11.51)	133	2.2	58(12.13)	169	2.5
bbsse	7	7	4	576	119(20.66)	403	232.4	132(22.92)	205	1930.5	134(23.26)	189	332.2	139(24.13)	252	349.3
bbtas	2	2	3	72	18(25.00)	90	0.2	29(40.28)	111	0.3	29(40.28)	102	0.3			
beecount	3	4	3	190	45(23.68)	105	0.5	47(24.74)	105	0.5	47(24.74)	93	0.5			
dk14	3	5	3	736	108(14.67)	251	0.9	109(14.81)	173	0.8	110(14.95)	156	0.8			
dk16	2	3	5	1178	61(5.18)	189	2.9	121(10.27)	335	2.5	136(11.54)	329	2.0			
dk17	2	3	3	486	46(9.47)	118	0.3	51(10.49)	126	0.3	58(11.93)	127	0.3			
dk27	1	2	3	88	16(18.18)	41	0.2	23(26.14)	57	0.2	23(26.14)	44	0.2			
dk512	1	3	4	272	22(8.09)	90	0.2	24(8.82)	94	0.3	25(9.19)	87	0.3			
ex2	2	2	5	892	48(5.38)	156	1.2	95(10.65)	273	1.1	98(10.99)	267	1.0			
ex3	2	2	4	296	56(18.92)	174	0.5	74(25.00)	183	0.5	74(25.00)	148	0.5			
ex4	6	9	4	386	39(10.10)	131	11.9	43(11.14)	83	11.8	43(11.14)	69	11.4			
ex5	2	2	4	618	31(5.02)	91	0.4	39(6.31)	95	0.4	43(6.96)	105	0.3			
ex7	2	2	4	614	51(8.31)	134	0.4	58(9.45)	133	0.4	58(9.45)	104	0.4			
keyb	7	2	5	1868	115(6.16)	420	695.8	151(8.08)	382	1198.7	158(8.46)	271	664.3			
mark1	5	4	4	360	56(15.56)	108	3.3	59(16.39)	72	4.7	60(16.67)	58	3.5			
opus	5	6	4	612	91(14.87)	142	11.3	105(17.16)	147	25.9	105(17.16)	127	17.3	111(18.14)	166	17.4
shiftreg	1	1	3	32	10(31.25)	47	0.1	10(31.25)	55	0.2	14(43.75)	36	0.2			
sse	7	7	4	576	119(20.66)	403	232.0	132(22.92)	205	1933.4	134(23.26)	189	330.9	139(24.13)	252	346.2