Testable Path Delay Fault Cover for Sequential Circuits *

Angela Krstić
Department of ECE, University of California, Santa Barbara, CA 93106
Srimat T. Chakradhar
C & C Research Laboratories, NEC USA, Princeton, NJ 08540
Kwang-Ting (Tim) Cheng
Department of ECE, University of California, Santa Barbara, CA 93106

Abstract

We present an algorithm for identifying a set of faults that do not have to be targeted by a sequential delay fault test generator. These faults either cannot independently affect the performance of the circuit or no test can be generated for them. To find such faults, our methodology takes advantage of the sequential behavior of the circuit as well as of the information about uncontrollable signals in the sequential circuit. It can handle sequential circuits described as two- or multi-level netlists. The outcome of applying our methodology is smaller fault set and possibly smaller test set. We present experimental results on several ISCAS 89 benchmark circuits demonstrating that a large number of path delay faults in these circuits either cannot or does not have to be examined for delay defects.

1 Introduction

Defects that occur during fabrication of an integrated circuit can slow down or speed up the performance (delay) of the circuit. Delay testing ensures that the fabricated circuit meets pre-specified timing constraints. Path delay fault model has been frequently used to model delay defects [1, 2, 3]. A circuit has a delay fault if the delay of any *combinational path* exceeds the rated clock period. A combinational path is an ordered set of gates $g_0, \ldots, g_n$ where g_0 and g_n are a primary input and output, respectively, of the circuit. Also, gate g_i is an input to gate g_{i+1} $(0 \leq i \leq n-1)$. A delay defect on a path in the circuit can be observed by propagating a transition through the path. Therefore, a path delay fault specification consists of a physical path and a transition that will be applied at the beginning of the path. Testing for path delay faults can uncover small manufacturing defects that are otherwise not detected by at-speed or stuck-at fault tests. A major limitation of this fault model is that the number of paths in a circuit can be very large (possibly exponential in the number of gates in the circuit). Testing for all path delay faults is impractical for most circuits. Also, this can result in an unacceptably large test set.

Recent work [4, 5, 6, 7] shows that not all path delay faults in a combinational circuit have to be considered for delay testing. For example, consider the circuit in Figure 1(a). Path P_1 (shown in bold face) consists of the gates a, b, c, d and e. We may not be able to observe a delay defect slowing down the falling transition on this path. This is because the logic value of gate d can be determined by a transition on input signal a rather than signal c on the path P_1. Let P_2 denote the path consisting of gates a, d and e (shown in bold face in Figure 1(b)). Clearly, if path P_2 does not have a delay defect that slows down the propagation of a falling transition, then the value on gate d is determined by signal a and not signal c. Therefore, delay defects on path P_1 can affect the delay of the circuit only if path P_2 also has delay defects. This implies that one does not have to test the path P_1 for a falling transition if path P_2 is tested for a falling transition.

Path delay faults in combinational circuits can be classified into two disjoint sets as shown in Figure 2. A *path delay fault cover* (PDFC) is a set of faults that will be considered for delay testing. A fault that is not in the PDFC cannot alter the circuit delay unless one or more faults in PDFC also occur. Therefore, tests for faults in PDFC can detect delay defects on any path. Many different path delay fault covers are possible for a given circuit. Since the number of faults in a PDFC determines the number of vectors and the effort required to generate the delay test set, it is important to find a PDFC

*This work was supported by NEC USA, Inc., by MICRO and by the National Science Foundation under Grant MIP-9409174.

EURO-DAC '96 with EURO-VHDL '96
0-89791-848-7/96 $4.00

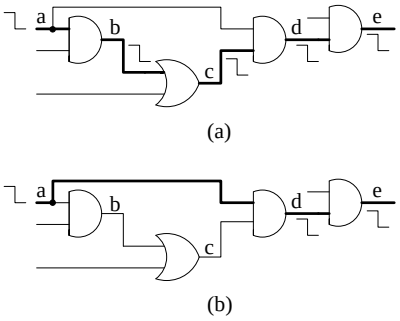

Figure 1: An example circuit.

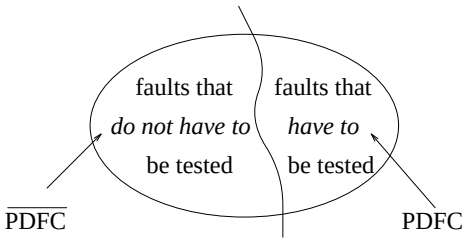

Figure 2: Classification in combinational circuits.

with the minimum number of faults. However, identifying such a PDFC may be intractable for practical circuits. We briefly review techniques to identify PDFCs for combinational circuits in Section 3. Identification of small PDFCs for sequential circuits is important for at least two reasons: (1) testing a path delay fault in a sequential circuit is significantly more difficult than testing a fault in a combinational circuit, (2) the number of *sequential paths*, often unknown, is significantly larger than the number of combinational paths. A sequential path is a concatenation of several combinational paths in the iterative array model of the sequential circuit. A recent paper [8] presents a method for identifying PDFCs in sequential circuits for which state transition diagrams are available. However, constructing state transition diagrams for most practical circuits is intractable.

No method has been reported to identify PDFCs for sequential circuits described as multi-level netlists. In this paper, we investigate identification of PDFCs for general sequential circuits. We propose a new method to identify path delay faults that do not have to be considered for delay testing. These faults either (1) cannot independently affect the performance of the sequential circuit, or (2) no test can be generated for these faults using gate-level delay test generation tools. The later are the *untestable path delay faults*. Figure 3 illustrates the proposed fault classification for sequential circuits. Only faults in the testable PDFC class will have to be considered for delay testing. The untestable path delay faults may adversely affect the delay of the circuit but gate-level delay test generators will be unable to find a test for these faults. Prior identification of these faults is desirable since test generators expend a significant amount of computing resources

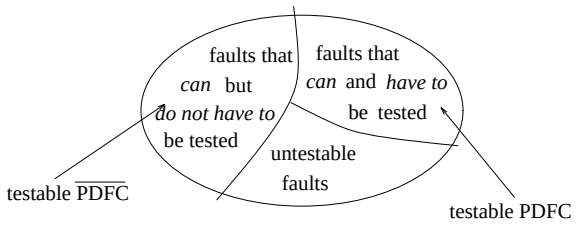

Figure 3: Classification in sequential circuits.

on these faults. Our method of identifying faults that do not belong to the PDFC is based on two key ideas: (1) we use the sequential behavior of the circuit to weed out faults that may belong to the PDFC of the combinational logic but they do not have to be included in the PDFC of the sequential circuit, and (2) we classify a fault by considering both vectors that are required to launch a given transition on a target path. This is unlike most existing classification techniques that ignore the first vector required to launch the transition. Consideration of the first vector usually entails high computational complexity. However, our technique of fault classification uses both vectors and is efficient on large circuits. Experimental results on several ISCAS 89 benchmark circuits show that a large number of path delay faults in these circuits do not have to be considered for delay testing.

2 Definitions

We use the terms on-inputs, off-inputs, controlling and non-controlling signal values as defined in [4, 5]. The following sensitization conditions are defined with respect to the vector that launches the transition on the path. A path is *static sensitizable* for a transition if, for each gate along the path, every off-input has a non-controlling value. Otherwise, the path is static unsensitizable. If a path is static sensitizable, then a delay defect on the path can adversely affect the delay of the circuit. Therefore, static sensitizable paths are included in any PDFC. A path is *functional sensitizable (FS)* [5] for a transition if, for each gate along the path, every off-input has a non-controlling value whenever the on-input has a non-controlling value. Otherwise, the path is *functional unsensitizable (FUS)* [5]. Note that for an FS fault there is no requirement on the off-input values when the corresponding on-input has a controlling value. Not all functional sensitizable faults have to be included in the PDFC [6]. Delay faults that are functional unsensitizable can never independently affect the circuit's performance and they do not have to be included in any PDFC. A functional sensitizable path is the same as a static cosensitizable path [9].

If a path is functional sensitizable for a given transition but it is not static sensitizable, then there is at least one off-input that has a controlling value whenever the corresponding on-input has a controlling value. This is true for all input vectors. Such an off-input is called *FS off-*

input. An off-input that has a controlling value whenever the corresponding on-input has a non-controlling value is called *FUS off-input*.

3 Prior work

Several methods have been proposed for identifying PDFCs for combinational circuits. Lam *et al.* [4] select a PDFC by identifying robust dependent faults (RD set). Their procedure is practical only for small designs because (1) identifying RD set requires multiple stuck-at fault test generation, and (2) the circuit has to be unfolded so that only PI's have more than one fanout. Cheng and Chen [5] attempt to identify a PDFC by using mandatory assignments and their implications to find paths that are functional unsensitizable. The PDFC identified by this procedure can be suboptimal (too large) for two reasons. First, only the second vector of the vector pair required to launch a transition is considered. Second, only local implications are used to identify functional unsensitizable faults. A better heuristic for identifying small PDFCs has also been recently reported [6]. Unlike the method presented in [5], this procedure also identifies functional sensitizable faults that do not have to be in the PDFC. The heuristic used in [6] involves ordering inputs of each gate in the circuit, i.e., finding an *input sort*. Given a target path and an input sort, the on-input partitions the off-inputs into higher and lower order off-inputs. For example, consider the four-input OR gate of Figure 4(a). The inputs a, b, c and d are assigned the integers 1, 2, 3 and 4, respectively. If input c is the on-input, then (for any target path through c) inputs a and b are lower order off-inputs while d is a higher order off-input.

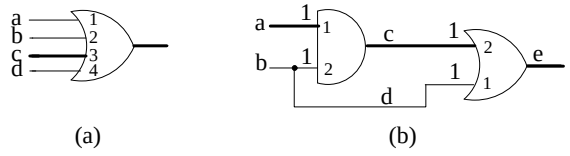

Figure 4: Using the *input sort* heuristic.

Given a path delay fault and an input sort, if a lower order off-input cannot be assigned a non-controlling value, then the off-input is *partially static unsensitizable (PSUS)*. It was shown in [6] that if an FS path has at least one PSUS off-input for all input vectors, then the FS path does not have to be included in the PDFC. For example, consider the circuit shown in Figure 4(b). The path consisting of gates a, c and e is functional sensitizable for a rising transition. However, the off-input d has a lower order than on-input c and d always assumes a controlling value. Therefore, the target path does not have to be included in the PDFC. Note that there are no mandatory assignments for higher order off-inputs. Also, different input sorts can lead to different PDFCs.

An *enchanced scan design* testing strategy has been proposed [10] for sequential circuits but high area overhead and long test application time make this testing strategy impractical. A procedure to find redundant path delay faults in sequential circuits has been reported recently [8]. However, this method can only be applied to circuits for which state transition diagrams are available.

4 Sequential PDFC

We use *segments* to precisely define a sequential path. A segment is an ordered set of gates $g_0, g_1, \ldots, g_n$. Here, g_0 is a primary input or the output of a flip-flop, and g_n is a primary output or the input of a flip-flop. Also, gate g_i is an input to gate g_{i+1} $(0 \leq i \leq n-1)$ and gates $g_1, \ldots, g_n$ are all Boolean logic gates. For example, consider the circuit shown in Figure 5(a). The ordered set of gates d, j and k is a segment. This segment begins at flip-flop d and terminates at the gate k. A sequential path is a concatenation of segments. The first segment begins at a primary input and the last segment terminates at a primary output. For example, consider again the circuit of Figure 5(a). A sequential path of three segments is as follows: (a, j), (c, g, h, i), and (b, e, f). Note that the first segment begins at the primary input a. This segment terminates at the input of flip-flop c. The second segment begins at the flip-flop c and it terminates at the input of flip-flop b. The last segment terminates at the primary output f.

A sequential circuit can have a huge number of sequential paths. Also, unlike combinational circuits where the number of paths is known, it may not be possible to count the number of sequential paths in circuits with feedback cycles. Therefore, finding a path delay fault cover by considering one sequential path at a time may be computationally infeasible for circuits with feedback cycles. Instead, we simultaneously examine all sequential paths that include a given segment. We model two faults for every segment to capture the propagation of the rising and falling transition through the segment. In the sequel, a path delay fault for a sequential circuit will be referred to as a *segment fault* and it is specified by a segment and a transition. There are two advantages of using segments. First, the target fault list for path delay fault testing will have a finite number of faults. The set of modeled faults is equal to the number of path delay faults in the combinational logic. Second, a PDFC of the sequential circuit can be specified as a finite set of segments rather than a possibly infinite set of sequential paths.

We distinguish between the PDFC of the combinational logic and the PDFC of the sequential circuit. A PDFC of the combinational logic, denoted as PDFC_C, is derived assuming that all flip-flop input and output signals are also primary inputs and outputs, respectively. A segment fault that is in PDFC_C may not belong to the PDFC of the sequential circuit (denoted as PDFC_S). This happens when none of the sequential paths that include the segment can independently affect the delay of the circuit. For example, consider again the sequential circuit of Figure 5(a). The segment fault $\{(d, j)$, rising$\}$ is in PDFC_C because the fault is static sensitizable. However, this fault

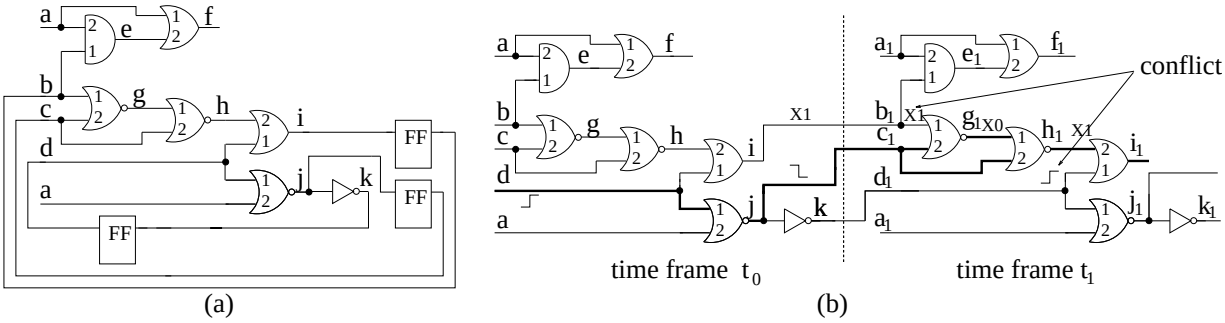

Figure 5: An example for PDFC_S $\subseteq$ PDFC_C.

does not have to be included in PDFC_S because there is no sequential path through the segment (d, j) that can propagate a rising transition and independently affect the circuit delay. To see this, consider two successive frames of the iterative array model of the circuit. The two frames are shown in Figure 5(b). This figure also shows the input sort for both frames. The implications of having a rising transition on signal d are indicated in the figure. Signal implications are obvious if we consider the rising transition to be the same as the application of two logic values: 0 followed by the logic value 1. As an example, gate i is assigned the value $X1$. This means that the application of a logic value 0 on d does not uniquely determine the value on i and this is indicated by the symbol X. However, if we apply the logic value 1 at d, then i assumes the value 1. Other signal values can be derived similarly. Sequential paths through the segment (d, j) will have to include either segment (c_1, g_1, h_1, i_1) or the segment (c_1, h_1, i_1) in the time frame t_1. The segment fault $\{(c_1, g_1, h_1, i_1)$, falling$\}$ is functional unsensitizable. This is because the on-input c_1 has a non-controlling value and the off-input b_1 has a controlling value for the second vector of the vector pair that launches the transition on signal d. Therefore, no sequential path propagating a falling transition through segment (c_1, g_1, h_1, i_1) has to be included in the PDFC of the sequential circuit. The segment fault $\{(c_1, h_1, i_1)$, falling$\}$ is functional sensitizable. However, because of the input sort, gate i_1 has a PSUS off-input d_1 for any test sequence that launches a rising transition on gate d. The off-input d_1 has a lower order and it assumes a controlling value for the second vector of the vector pair that launches the transition on d. Therefore, no sequential path that propagates a falling transition through segment (c_1, h_1, i_1) has to be included in the PDFC of the sequential circuit. This implies that we can exclude the segment fault $\{(d, j)$, rising$\}$ from PDFC_S. In general, a sequential path does not have to be considered for delay testing if it has a FUS or PSUS off-input for all input sequences that initiate a transition on the path. If a segment fault is not in PDFC_C, then there is already at least one FUS or PSUS off-input. Therefore, this fault does not have to be included in PDFC_S. Only a subset of faults in PDFC_C will be included in PDFC_S.

5 Untestable segment faults

Sequential circuits can have segment faults for which no test can be found by the test generator. Also, for some faults it may be impossible to prove that no test is possible from any initial state of the circuit. For example, consider again the circuit of Figure 5(a) and the segment fault $\{(c, h, i)$, rising$\}$. It can be shown that this fault is static sensitizable if we consider only the combinational logic. Therefore, the fault is included in PDFC_C. If we consider the sequential circuit, the flip-flop c has to assume the value 1 to launch a transition on the segment. This implies that signal d has to assume the value 0. However, a test generator that starts with flip-flops in an unknown state will not be able to initialize d to the logic value 0. This is because d can be set to 0 during a clock period only if its value was 0 in the previous clock period. Since the test generator assumes that d starts with an unknown value, it will not be able to initialize d to 0. Therefore, it is not possible to determine if the segment fault $\{(c, h, i)$, rising$\}$ should be included or excluded from PDFC_S. We refer to such faults as *untestable* segment faults.

A delay test generator that processes one sequential path at a time will consider every sequential path through the segment and prove the path to be untestable. However, this method will require significant computing resources. It is desirable to identify untestable segments and eliminate them from consideration by a delay test generator. To reduce the number of untestable faults, one can consider each initial state of the circuit separately and determine if the sequential path has a test. If a delay test is possible for every initial state, then we can include the fault in PDFC_S. However, this method is impractical for most circuits of interest. Multiple observation time strategy [11] is another option but this technique also requires prohibitively high computational resources.

Several known techniques [12, 13, 14, 15] can be used to determine the set of values that a signal in the sequential circuit can or cannot assume in any time frame. We refer to this information as the *functional signal constraint (FSC)* of the signal. The FSC information for a signal is derived assuming an unknown initial state for the sequential circuit. If a signal assumes a value of 0 (1) in every time frame, we assign the symbol $C0$ ($C1$) to the

signal. If a signal cannot assume a value 0 (1) in any time frame, we assign the symbol $U0$ ($U1$). If it is impossible to justify a logic value of 0 or 1 on a signal, we assign the symbol U. Finally, if both 0 and 1 value on a signal can be justified, we assign the symbol G. These symbols have been used in an earlier work [15]. We refer to these symbols as the *FSC values*. As an example, consider again the circuit of Figure 5(a). Since signal d cannot assume the value 0, we assign it an FSC value of $U0$.

The FSC values are useful in quickly identifying untestable segment faults. If any signal on a target segment fault has an FSC value other than G, then it will not be possible to initiate a transition on the segment. Therefore, the segment fault is untestable. We refer to such faults as *unexcitable* segment faults. For example, consider the AND gate shown in Figure 6(a). This gate

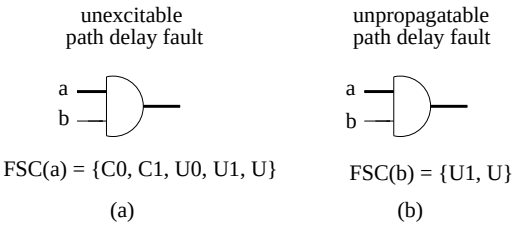

Figure 6: Untestable faults through an AND gate.

has two inputs a and b. Let a and b be the on-input and off-input, respectively. If the FSC value of signal a is $C0$, $C1$, $U0$, $U1$ or U, then no transition can be launched on signal a. Therefore, all sequential paths through signal a are untestable. Note that our analysis considers both vectors of the vector pair that can initiate a transition on signal a. Based on the FSC value, it may be the first or the second vector of the transition initiating vector pair that cannot be justified starting from an unknown initial state of the sequential circuit.

If an off-input cannot assume a non-controlling value, then the test generator cannot derive a delay test. For example, consider the AND gate of Figure 6(b). If signal b has an FSC value of Uor $U1$ (cannot assume value 1), then it will not be possible to derive a delay test for a sequential path through a. We refer to such untestable segment faults as *unpropagatable* segment faults.

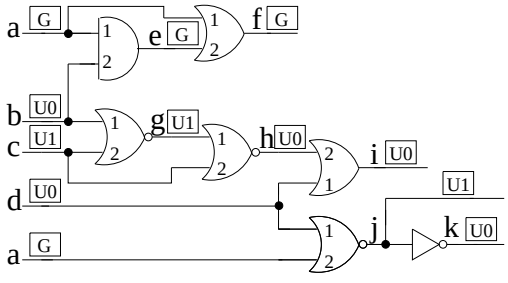

Figure 7: FSC values for the circuit in Figure 5(a).

Figure 7 shows the FSC values for the circuit shown in Figure 5(a). The FSC values were derived using the symbolic simulation technique described by Liang, Lee and Chen [15]. Only segments (a, f) and (a, e, f) have to be considered by the delay test generator. All other segments correspond to untestable segment faults because a transition cannot be initiated on one or more on-inputs.

6 Identifying sequential PDFC

Our algorithm consists of two parts. First, we identify a segment fault that belongs to the PDFC of the combinational logic. Known techniques [5, 6] can be employed to identify segment faults in PDFC_C. However, these techniques do not take advantage of the sequential behavior of the circuit. We modify the algorithm of Sparmann *et al.* [6] to include the untestable segment fault identification ideas discussed in Section 5.

Second, we examine if the segment fault has to be included in the PDFC of the sequential circuit. Based on the characteristics of the segment, one of the following steps is used to determine if the fault belongs to PDFC_S:

1. If the segment starts at a PI and terminates at a PO, then there is only one sequential path through this segment. We include the segment fault in PDFC_S.
2. If the segment starts at a PI and terminates as an input to a flip-flop, then we enter the **Forward** phase. We consider a finite number of frames of the iterative array model of the circuit. The frames are labeled $t_0, t_1, \ldots, t_k$. Here, frame t_{i+1} immediately follows frame t_i $(0 \leq i \leq k-1)$ in time. The target segment is in frame t_0. We implicitly examine all paths in the iterative array model that include the target segment. We also consider the sequential nature of the circuit and imply mandatory assignments across time frames. The iterative array model can be considered as a combinational circuit. Here, inputs to frame t_0 are considered as primary inputs and outputs of frame t_k are considered as primary outputs. If none of the combinational paths that include the target segment have to be included in the PDFC of the iterative array model, then the segment fault can be excluded from the PDFC of the sequential circuit. Again, we use FSC values and the algorithm of Sparmann *et al.* to efficiently process combinational paths. If FSC values are used to exclude the segment fault from PDFC_S, then the segment fault is an untestable segment fault.
3. If the segment starts at a flip-flop and terminates at a PO, then we enter the **Backward** phase. Again, we consider a finite number of frames of the iterative array model of the circuit. The frames are labeled $t_{-k}, t_{-k+1}, \ldots, t_0$. The target segment is in frame t_0. We implicitly examine all paths in the iterative array model that terminate at the target segment. If none of these paths have to be included in the PDFC of the iterative array model, then the segment fault can be excluded from the PDFC of the sequential circuit.

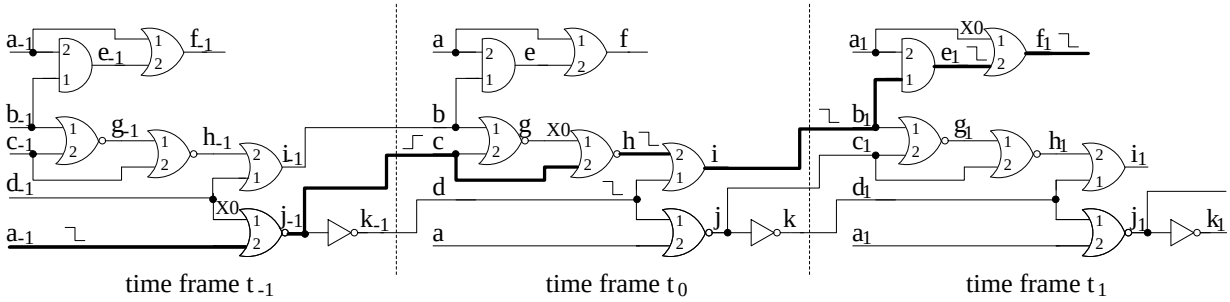

Figure 8: Untestable path delay faults.

Again, if FSC values are used to exclude the segment fault, then we classify the fault as untestable.

4. If the segment starts at flip-flop and terminates as an input to a flip-flop, then we first employ the **Forward** phase described in Step 2. If the segment fault cannot be excluded from the PDFC_S or the fault cannot be classified as untestable, then we employ the **Backward** phase described in Step 3.

We illustrate the mechanics of our algorithm using an example. To keep the discussion simple, we do not use FSC values. Consider again the circuit shown in Figure 5(a). Let the target segment fault be $\{(c, h, i), \text{rising}\}$. The corresponding segment starts at the flip-flop c and terminates as an input to the flip-flop b. If we do not use FSC values, then it can be shown that the segment fault is included in PDFC_C. Therefore, we enter the **Forward** phase. The iterative array model consisting of three frames labeled t_{-1}, t_0 and t_1 is shown in Figure 8. The **Forward** phase adds the frame t_1. We consider paths in frame t_1 that begin from the target segment. If we consider the segment (b_1, e_1, f_1) in frame t_1, then we have reached a PO. Implications of the mandatory assignments for the target segment fault are not adequate to exclude segment (b_1, e_1, f_1) from the PDFC of frame t_1. Therefore, based on the **Forward** phase, the target segment fault may have to be included in the PDFC_S. Next, we enter the **Backward** phase. We add the frame t_{-1} and examine paths terminating at the target segment. Frame t_{-1} has a segment (a_{-1}, j_{-1}) that starts at a PI and terminates at the target segment. Mandatory assignments in frame t_{-1} are again, not adequate to exclude the segment (a_{-1}, j_{-1}) from the PDFC of frame t_{-1}. Therefore, we include the target segment fault in PDFC_S. Note that if FSC values are used, then this fault can be classified as untestable (see Section 5). Also, one can use a branch and bound algorithm instead of implications to more accurately determine the status of the segment fault. However, this would require significant computing resources.

7 Experimental results

Our algorithm for identifying testable PDFC in sequential circuits has been implemented in the C programming language. Our implementation does not distinguish between faults that do not have to be tested and the untestable faults. This is because a delay test generator does not have to process either type of faults. All experiments reported here were performed on a SUN Sparc 5 workstation.

Table 1 reports experimental results for several ISCAS 89 benchmark circuits. The total number of segments in a circuit is shown under column *number of paths*. The column *PDFC_C - paths(%)* reports the percentage of segments that are included in the PDFC of the combinational logic. The number of CPU seconds required to do the classification is shown under column *PDFC_C - cpu(s)*. For computing the PDFC of the sequential circuit, it is possible to use different input sorts across time frames but this would require us to compute and store the input sort for every time frame. In our implementation we use the same input sort for every primary output and time frame in the iterative array model. Our input sort is based on the number of FS combinational paths that pass through a signal in the circuit.

Ckt	num. of paths	*PDFC_C*		*PDFC_S*			
				without FSC		with FSC	
		paths (%)	cpu (s)	paths (%)	cpu (s)	paths (%)	cpu (c)
s208	290	100.0	2	76.2	6	36.5	3
s298	462	79.6	3	79.6	8	79.6	8
s349	730	91.9	5	91.9	33	87.8	22
s382	800	92.6	6	92.5	28	56.2	20
s386	414	100.0	8	93.5	18	81.4	12
s400	896	87.4	6	87.3	30	57.8	18
s420	738	100.0	7	71.7	21	14.4	5
s444	1070	78.8	7	78.8	32	51.0	19
s510	738	100.0	13	100.0	69	0.0	6
s526	820	88.3	9	88.3	38	35.2	13
s526n	816	88.5	9	88.5	37	35.4	12
s641	3444	65.9	18	65.5	31	39.3	16
s713	43624	11.4	33	11.4	64	5.8	30
s820	984	100.0	38	100.0	172	96.4	109
s832	1012	98.8	40	98.8	189	95.4	119
s838	2018	100.0	25	75.8	90	5.25	15
s1423	89452	51.8	619	51.8	3461	10.6	811
s1488	1924	99.6	119	99.6	319	99.6	319
s1494	1952	98.9	121	98.9	328	98.9	328
s5378	27046	83.4	261	70.3	1190	31.4	264
s9234	489708	14.5	3211	10.4	15787	1.9	3384

Table 1: Results for ISCAS 89 benchmark circuits.

We perform two experiments. We compute the PDFC_S with and without the use of the FSC values. Column *without FSC* reports the percentage of segments that will have to be included in the PDFC of the sequential circuit if no FSC values are considered. Column *with FSC* reports the percentage of segments that have to be processed by a delay test generator when FSC values are used. The CPU seconds required for identifying the PDFC are reported in column *cpu(s)*. For both experiments, we used at most 5 frames each for the **Forward** and **Backward** phase.

As an example, consider the circuit **s9234**. This circuit has 489,708 segment faults. If we only consider the combinational logic, then 14.5% of segment faults have to processed by a sequential delay test generator. If we use the iterative array model but no FSC values, then 10.4% of faults would have to be considered by a delay test generator. However, if FSC values are also used, then only 1.9% of the total number of segment faults have to be examined. Our experimental results show that for many sequential circuits only a small fraction of segment faults have to be considered for delay test generation.

8 Conclusions

Not all path delay faults in a sequential circuit have to be tested to guarantee correct timing behavior of the circuit. Faults that can never determine the performance of the circuit unless some other faults also happen will be detected by testing the faults in a path delay fault cover. For some path delay faults in a sequential circuit a test cannot be found. Therefore, these faults should be eliminated before sequential gate-level delay test generation starts. We have presented a methodology to identify a testable path delay fault cover for sequential circuits represented as multi-level netlists. Eliminating faults that do not have to be considered for delay test generation reduces the computational effort of test generators as well as the test set size. The results of our experiments have shown that in many large sequential circuits the size of a testable PDFC is significantly smaller than the total number of path delay faults.

Acknowledgment - The authors would like to thank members of C&C Research Labs, NEC USA: Rabindra Roy for valuable discussions and to Vijay Gangaram and Steve Rothweiler for providing FSC data. Thanks are also due to Prof. Uwe Sparmann of the University of Saarland, Germany for providing the tool for input sort heuristic.

References

[1] G. L. Smith. Model for Delay Faults Based upon Paths. *Proceedings of IEEE International Test Conference*, pages 342–349, November 1985.

[2] C. J. Lin and S. M. Reddy. On Delay Fault Testing in Logic Circuits. *IEEE Transactions on CAD*, CAD-6(5):694–703, September 1987.

[3] S. T. Chakradhar, M. Iyer, and V. D. Agrawal. Energy Models for Delay Testing. *IEEE Transactions on CAD*, 14(6):633–634, June 1995.

[4] W. K. Lam, A. Saldanha, R. K. Brayton, and A. L. Sangiovanni-Vicentelli. Delay Fault Coverage, Test Set Size, and Performance Trade-Offs. *IEEE Transactions on CAD*, 14(1):32–44, January 1995.

[5] K.-T. Cheng and H.-C. Chen. Delay Testing For Non-Robust Untestable Circuits. *Proceedings of IEEE International Test Conference*, pages 954–961, October 1993.

[6] U. Sparmann, D. Luxenburger, K.-T. Cheng, and S. M. Reddy. Fast Identification of Robust Dependent Path Delay Faults. *Proceedings of 32nd Design Automation Conference*, pages 119–125, June 1995.

[7] M. A. Gharaybeh, M. L. Bushnell, and V. D. Agrawal. Classification and modeling of Path Delay Faults and Test generation Using Single Stuck-Fault Tests. *Proceedings of IEEE International Test Conference*, pages 139–148, October 1995.

[8] R. Tekumalla and P. R. Menon. Identifying Redundant Path Delay Faults in Sequential Circuits. *VLSI Design*, pages 406–411, January 1996.

[9] S. Devadas, K. Keutzer, and S. Malik. Delay Computation in Combinational Logic Circuits: Theory and Algorithms. *Proceedings of IEEE/ACM International Conference on Computer-Aided Design*, pages 176–179, November 1991.

[10] K.-T. Cheng, S. Devadas, and K. Keutzer. Delay-Fault Test Generation and Synthesis for Testability Under a Standard Scan Design Methodology. *IEEE Transactions on CAD*, 12(8):1217–1231, August 1993.

[11] I. Pomeranz and S. M. Reddy. The Multiple Observation Time Test Strategy. *IEEE Transactions on CAD*, 41(5):627–637, May 1992.

[12] K.-T. Cheng. Redundancy Removal for Sequential Circuits Without Reset States. *IEEE Transactions on CAD*, 12(1):13–24, January 1993.

[13] I. Pomeranz and S. M. Reddy. On Identifying Undetectable and Redundant Faults in Synchronous Sequential Circuits. *Proceedings of 12th IEEE VLSI Test Symposium*, pages 8–14, April 1994.

[14] V. D. Agrawal and S. T. Chakradhar. Combinational ATPG Theorems for Identifying Untestable Faults in Sequential Circuits. *IEEE Transactions on CAD*, 14(9):1115–1127, September 1995.

[15] H.-C. Liang, C. L. Lee, and J. E. Chen. Identifying Untestable Faults in Sequential Circuits. *IEEE Design & Test of Computers*, 12(3):14–23, Fall 1995.