NAIST-IS-DD0361215

Doctoral Dissertation

Studies on Test Generation Complexity for Stuck-At and Path Delay Faults Based on τ^k-Notation

Chia Yee Ooi

September 22, 2006

Department of Information Processing
Graduate School of Information Science
Nara Institute of Science and Technology

A Doctoral Dissertation
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Chia Yee Ooi

Thesis Committee:

Professor Hideo Fujiwara (Supervisor)
Professor Hiroyuki Seki (Co-supervisor)
Associate Professor Michiko Inoue (Co-supervisor)

Studies on Test Generation Complexity for Stuck-At and Path Delay Faults Based on τ^k-Notation[*]

Chia Yee Ooi

Abstract

This dissertation introduces τ^k-notation to be used for analyzing test generation complexity of a class of circuits with a type of faults based on combinational test generation complexity. Using τ^k-notation, the test generation complexity for acyclic sequential circuits with stuck-at faults are reconsidered. The test generation complexity for combinational circuits and acyclic sequential circuits with robust and non-robust path delay faults are also analyzed. On the other hand, easily testable classes of cyclic sequential circuits are defined in the aspects of the number of time frames and running time taken by the state justification and differentiation for both cases of stuck-at faults and path delay faults. Apart from classification of sequential circuits based on combinational test generation complexity, a new class of sequential circuits called acyclically testable sequential circuits is introduced. The application based on the classification of sequential circuits is two-fold. Firstly, the test generation method can be applied for combinational circuits with stuck-at faults, which is more efficient. Second, a design for testability or a synthesis for testability method can be introduced based on the properties defined for each class of sequential circuits.

Keywords:

τ^k-notation, easy testability, combinational/acyclic test generation complexity, design/synthesis for testability(DFT/SFT), stuck-at faults, path delay faults

[*]Doctoral Dissertation, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD0361215, September 22, 2006.

List of Publication

Journal Papers

1. C. Y. Ooi, T. Clouqueur and H. Fujiwara, "Classification of Sequential Circuits Based on τ^k Notation and Its Applications," IEICE Trans. on Information and Systems, pp. 2738-2747, December 2005.

2. C. Y. Ooi, T. Clouqueur and H. Fujiwara, "Analysis of Test Generation Complexity for Stuck-At and Path Delay Faults Based on τ^k-Notation," IEICE Trans. on Information and Systems (Conditional Acceptance).

International Conferences (Reviewed)

1. C. Y. Ooi and H. Fujiwara, "Classification of Sequential Circuits Based on τ^k-Notation," 13th IEEE Asian Test Symposium, pp. 348-353, 2004.

2. C. Y. Ooi, T. Clouqueur and H. Fujiwara, "Test Generation Complexity for Path Delay Faults Based on τ^k-Notation," 6th IEEE Workshop on RTL and High Level Testing, pp. 61-72, 2005.

3. C. Y. Ooi and H. Fujiwara, "A New Class of Sequential Circuits with Acyclic Test Generation Complexity," 24th IEEE International Conference on Computer Design, October 2006 (To Appear).

4. C. Y. Ooi and H. Fujiwara, "A New Scan Design Technique Based on Pre-Synthesis Thru Functions," 15th IEEE Asian Test Symposium, November 2006 (To Appear).

Technical Reports

1. C. Y. Ooi and H. Fujiwara, "Classification of Sequential Circuits Based on Combinational Test Generation Complexity," Technical Report of IEICE, (DC2003-101), Vol. 103, No. 668, pp. 67-72, February 2004.

2. C. Y. Ooi, T. Clouqueur and H. Fujiwara, "A New Class of Sequential Circuits with Acyclic Test Generation Complexity," Technical Report of IEICE, (DC2005-80), Vol. 105, No. 607, pp. 49-54, February 2006.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1. The Time Complexity of Test Generation Problem

It has been known for about three decades that the test generation problem, even for combinational circuits with stuck-at faults, is NP-complete[1]. In other words, there does not exist an algorithm that solves an arbitrary instance of the problem in polynomial time, unless $P = NP$. However, empirical observation showed that the time complexity of test generation for practically encountered combinational circuits with single stuck-at faults seems to be polynomial, that is $O(n^r)$ for some constant r, where n is the size of the circuits [2,3]. For example, the automatic test pattern generation (ATPG) tool named SPIRIT [4] can achieve 100% fault efficiency for benchmark circuits ITC'99, surpassing the existing commercial ATPGs. Consequently, the works related to the classification of sequential circuits based on combinational test generation complexity started. In this dissertation, combinational test generation is assumed to achieve complete fault efficiency based on the work in [4]. Therefore, combinational test generation complexity is fundamental in the discussion on the time complexity of test generation of sequential circuits. When more and more classes of sequential circuits is introduced, it is helpful to use a general notation to discuss the test generation complexity. Therefore, a notation called τ^k-notation is defined based on combinational test generation complexity. The combinational test generation complexity is assumed as $\Theta(n^r)$ for some constant $r > 2$ and it is denoted by $\tau(n)$, where n is the size of the circuit.

1.2. Classification of Sequential Circuits

Several classes of acyclic sequential circuits have been introduced in the previous works. These include balanced sequential circuits [5], strongly balanced sequential circuits [6], internally balanced sequential circuits[7], switched balanced sequential circuits [8] and switched internally balanced sequential circuits[9], all of which are acyclic. The test generation for internally balanced sequential circuits and balanced sequential circuits with stuck-at faults has been shown to be reducible into that for combinational circuits with stuck-at faults [10,11].

Although a test generation model was proposed to transform any acyclic sequential circuit into its combinational equivalent with logic duplicates at most d time frames where d is the sequential depth in [12], the test generation complexity for acyclic sequential circuits has not yet been clarified. Neither has been that for cyclic sequential circuits whose test generation problem is generally modeled by an iterative logic array and may have greater time complexity than acyclic sequential circuits. Apart from stuck-at fault model, which is the representative fault of static faults, path delay fault model, which is a powerful timing fault model, is important to ensure the temporal correctness of a circuit. The relationships between the test generation for combinational circuits with path delay faults and that with stuck-at faults have been discussed in [13,14,15,16,17]. They showed that the ATPG for stuck-at faults plus some polynomial circuit transformation can be used as an ATPG for robust and non-robust path delay faults. The test generation was not discussed explicitly in the aspect of time complexity. Neither was the test generation complexity for sequential circuits with path delay faults.

In this dissertation, the analysis of test generation complexity is done on the acyclic sequential circuits and some cyclic sequential circuits with stuck-at faults and path delay faults using τ^k-notation. As the first step of the work, only robust and non-robust faults are discussed for path delay faults. Besides analyzing the test generation complexity of several existing classes of circuits with stuck-at faults and path delay faults, several new classes of sequential circuits are introduced. As mentioned, previously defined classes of sequential circuits with combinational test generation complexity are acyclic. In this dissertation, some cyclic sequential circuits with combinational test generation complexity are identified. These classes include l-length-bounded testable circuits, l-length-bounded validity-identifiable circuits, t-time-bounded testable circuits and t-time bounded validity-identifiable circuits. Classification of sequential circuits with combinational test generation complexity is extended to classification of sequential circuits with acyclic test generation complexity. Also, since the test generation of acyclic sequential circuits seems not very difficult for practically encountered circuits, classification of sequential circuits is also done based on the time complexity of acyclic sequential circuits, or simply acyclic test generation complexity. A new class of sequential circuits with acyclic test generation complexity called acyclically testable sequential circuits is also introduced.

1.3. Application of Classification of Sequential Circuits

Clarifying the test generation complexity for sequential circuits or classification of sequential circuits leads to two applications.

First, based on the properties of a known class, a special ATPG is designed to run the test generation on the circuit, which is generally more efficient than does the general sequential ATPG. For instance, the test generation technique involving separating of separable inputs, wire replacing, combinational test generation and sequence transformation, is used to obtain a test in an internally balanced sequential circuits, the test generation time of which has been proved reduced [7]. Another application is design for testability (DFT) and synthesis for testability (SFT). For example, based on the feature of balanced structure, the DFT called BALLAST was introduced in [5]. For a given arbitrary sequential circuti (resp. arbitrary design), a DFT method (resp. SFT method) is designed and applied to augment the circuit into one of the easily testable classes of sequential circuits.

In the dissertation, the introduction of each new class of sequential circuits is followed by a test generation procedure and a DFT or SFT method.

1.4. Dissertation Organization

The dissertation is organized as follows. Chapter 2 introduces some fault models, including single stuck-at fault model, multiple stuck-at fault model and path delay fault model. The chapter also introduces the test generation model for acyclic sequential circuits called time-expansion model (TEM) [18]. The extension of TEM called double time expansion model [19] is also elaborated.

When more and more classes of circuits are introduced and the discussion of test generation complexity for each class with different type of faults becomes important, it would be useful if there is a general notation to be used in the discussion. Therefore, chapter 3 presents a new notation called τ^k-notation, which is defined using asymptotic notation. Using τ^k-notation, the test generation complexity for existing classes of sequential circuits is reconsidered.

Chapter 4 classifies sequential circuits with stuck-at faults based on combinational test generation complexity. Classes of easily testable sequential circuits have been considered for only acyclic sequential circuits. This chapter introduces several classes of sequential circuits that include some cyclic sequential circuits. The test generation procedure for each new class of sequential circuits and a DFT method that augments an arbitrary sequential circuit into one belonging to the new class are also presented.

Chapter 5 classifies sequential circuits with path delay faults based on combinational test generation complexity. Besides, the relationship between test generation for sequential circuits with stuck-at faults and test generation for sequential circuits with path delay faults is discussed. There exists a class of sequential circuits whose test generation for stuck-at faults and test generation for path-delay faults are not equivalent.

Chapter 6 extends the work of classifying sequential circuits based on combinational test generation complexity to classfying sequential circuits based on acyclic test generation complexity. In this chapter, new concept of several circuit properties are introduced. Based on the properties, a new class of sequential circuits called acyclically testable sequential circuits is introduced. This is followed by a test generation procedure for the new class and a DFT method to augment an arbitrary circuit into acyclically testable. Chapter 7 concludes the dissertation by its main contributions and identifies some future works.

Chapter 2

Fault Models and Test Generation Models

2.1. Introduction

Fault modeling alleviates the test generation complexity because it obviates the need for deriving tests for each possible defect. In fact, many physical defects map to a single fault at the higher level. Therefore fault modeling is essential in testing. On the other hand, test generation model is another important concept in testing, which is used to model the problem of test generation. This chapter introduces fault models at logical level. These fault models include single stuck-at fault model, multiple stuck-at fault model, path delay fault model and segment delay fault model. Besides, fault models, the chapter also elaborates a test generation model called time expansion model (TEM)[18] and its extended model called double time expansion model (DTEM)[19].

2.2. The Single Stuck-At Fault Model

The single stuck-at-fault model is the most widely studied and used in testing. Alhough it is not universal, it is useful because it represents many different physical faults and is independent of technology. Furthermore, it has empirically shown that tests that detect single stuck-at faults detect many other faults as well. In structural testing, it is necessary to make sure that the interconnections in the given circuit are able to carry both logic 0 and 1 signals. The stuck-at-fault model is derived directly from these requirements. A line is stuck-at 0(SA0) or stuck-at 1(SA1) if the line remains fixed at a low or high voltage level, respectivley. A single stuck-at-fault that belongs to the single stuck-at-fault model is only assumed to happen on only one line in the circuit. If the circuit has k lines, it can have $2k$ single SAFs, two for each line.

2.3. The Multiple Stuck-At Fault Model

If the stuck-at-fault occurs on more than one line in the circuit, the faults are said to belong to the multiple stuck-at-fault model. To model a circuit with a multiple stuck-at-fault by a model containing only one single stuck-at fault, m extra gates are added into the circuit as follows, where m is the multiplicity of faults.

- A two-input OR (resp. AND) gate is inserted in a line if the line is stuck-at 1, SA1 (resp. stuck-at 0, SA0) and one of the input lines of the gate is fed from the ground of the circuit as a fanout branch of a ground line G. The input of AND gate that is fed from line G is inverted. The multiple fault is then represented by a single SA1 fault on the fanout stem G.

The controlling value of each gate is the same as the value at which the line is stuck. Thus, the faulty value appears on the output of each gate if the SA1 fault on line G is activated. Otherwise, the gate forces the correct value on it. Thus, the model is satisfying the conditions of circuit equivalence and fault equivalence.

Example 2.3.1. Figure 2.1(a) shows four lines with inputs a, b, c and d, and the respective outputs A, B, C and D. A multiple SAF here consists of the first two lines stuck-at 1 and the others stuck-at 0. Figure 2.1 (b) shows the representation of a multiple stuck-at-fault with a single stuck-at-fault model.

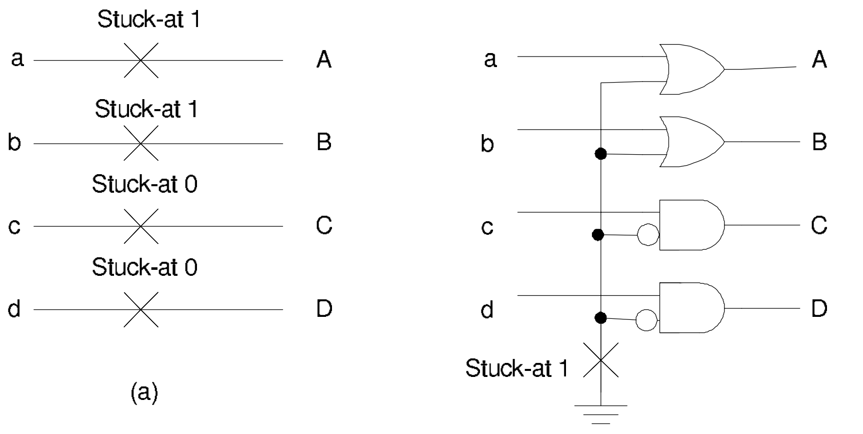


Figure 2.1. (a) A multiple SAF. (b) An equivalent single SAF.

2.4. Delay Fault Models

Several fault models have been introduced in the previous works. These include transition fault model, gate delay fault model, line delay fault model, path delay fault model and segment delay fault model. In this section, segment delay fault model besides path delay fault model is also discussed because it is useful in relating the test generation of sequential circuits with path delay faults to the test generation of combinational circuits with stuck-at faults in the dissertation.

2.4.1 Path Delay Fault Model

In a sequential circuit, a path is defined as an ordered set of gates $\{g_1, g_2, ..., g_m\}$, where g_1 is a primary input or a FF and g_m is a primary output or a FF (Figure 2.2). Also, gate g_i is an input to gate g_{i+1} $(1 \leq i \leq m-1)$. A path has a delay fault if for any input sequence that generates a rising or falling transition through the path, the propagation time of such transition exceeds a specified clock period. Such a delay fault on a path is said to be a ***path delay fault (PDF)***[20].

Let $\mathcal{S}$ denote a sequential circuit and P denote a path in $\mathcal{S}$. Let f be a rising (resp. falling) PDF on P and let $\mathcal{S}_f$ be the faulty circuit of $\mathcal{S}$ in the presence of f. The fault is also denoted by $P\uparrow$ (resp. $P\downarrow$). The fault f is ***testable*** if there exists an input sequence t for $\mathcal{S}$ and $\mathcal{S}_f$ such that the following conditions hold.

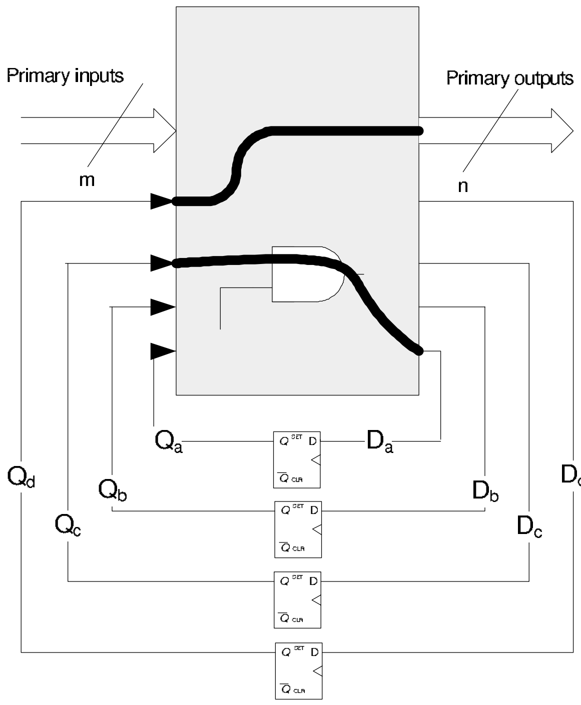


Figure 2.2. PDFs in a sequential circuit

1. A rising (resp. falling) signal transition is launched at the starting point (a FF or a primary input) of P in $\mathcal{S}$ by t at time T_1.
2. The transition launched at the starting point of P is propagated to the ending point (a FF or a primary input) of P along P in $\mathcal{S}$ by t.
3. The captured or observed value induced by the transition at the ending point of P in $\mathcal{S}_f$ is different from that of $\mathcal{S}$ at time T_2 where interval $T_2 - T_1$ is the maximum allowable path delay for the fault-free circuit.
4. The output sequence of $\mathcal{S}$ and that of $\mathcal{S}_f$ are different at time T_2.

Such an input sequence t is regarded as a ***PDF test sequence***.

2.4.2 Segment Delay Fault Model

In a combinational circuit, a segment S is defined as an ordered set of gates $\{g_1, g_2, ..., g_m\}$, where the output of g_1 is the starting point of S, the output of g_m is the ending point of S and m is the length of the segment. The length of the segment, m, can be anywhere from 1 to the number of gates in the longest path in the circuit. Also, gate g_i is an input to gate g_{i+1} $(1 \leq i \leq m-1)$. A segment has a delay fault if the propagation time of the rising or falling signal transition through the segment exceeds a specified limit. Such a delay fault on a segment is said to be a ***segment delay fault (SDF)*** [21]. It is assumed that an SDF is large enough to cause a delay fault on all the paths that include the segment.

Let C be a combinational circuit and S be a segment in C. Let f be a rising (resp. falling) SDF on S and let C_f be the faulty circuit of C in the presence of f. The fault f is also denoted by $S \uparrow$ for rising transition ($S \downarrow$ for falling transition). The fault f is ***testable*** if there exists an input vector pair $< v_1, v_2 >$ for C and C_f such that the following conditions hold.

1. A rising (resp. falling) signal transition is launched at the starting point (a gate or a primary input) of S in C by $< v_1, v_2 >$ at time T_1.
2. The transition launched at the starting point of S is propagated to the ending point (a gate or a primary output) of S along S in C by $< v_1, v_2 >$.

3. The value induced by the transition at the ending point of S of C_f is different from that of C at time T_2 where interval $T_2 - T_1$ is the maximum allowable segment delay in the fault-free circuit.

4. The second output vector of C and that of C_f are different at time T_2.

Such an input vector pair $< v_1, v_2 >$ is regarded as an ***SDF 2-pattern test***.

A controlling value for a gate g is the value at its input that determines the value at the output independent of the other inputs, and is denoted as $A(g)$. A non-controlling value for a gate g is the value at its input which is not a controlling value for the gate, and is denoted as $I(g)$. Let P (resp. S) be a path (resp. segment). The side-inputs of g_i along P (resp. S) are denoted as $SI(g_i, P)$ (resp. $SI(g_i, S)$).

A path (resp. segment) $P = \{g_1, g_2, ..., g_m\}$ (resp. $S = \{g_1, g_2, ..., g_m\}$) is said to be robust testable for the rising transition at g_m by the vector pair $< v_1, v_2 >$ if at each node g_i, $g_i(v_1) \neq g_i(v_2)$ yields the desired transition being tested, and for each node $h_j \in SI(g_i, P)$ (resp. $h_j \in SI(g_i, S)$):

1. $h_j(v_2) = I(g_i)$, and

2. if $g_{i-1}(v_1) = I(g_i)$, then there is no transition on h_j.

A path (resp. segment) $P = \{g_1, g_2, ..., g_m\}$ (resp. $S = \{g_1, g_2, ..., g_m\}$) is said to be non-robust testable for the rising transition at g_m by the vector pair $< v_1, v_2 >$ if at each node g_i, $g_i(v_1) \neq g_i(v_2)$ yields the desired transition being tested, and for each node $h_j \in SI(g_i, P)$ (resp. $h_j \in SI(g_i, S)$), $h_j(v_2) = I(g_i)$. Note that a robust testable PDF is also a non-robust testable PDF but the converse is not true.

Example 2.4.1. Let's consider the two-line segment $\{g_1, g_2\}$ in the circuit shown in Figure 2.3. The falling transition is launched at line g_1 by launching a rising transition at x_1. Then the transition is propagated to g_2 by assigning two-pattern test 11 to x_2. Finally the transition at g_2 is propagated to output z. The test at $x_1x_2x_3$ is $< 010, 110 >$, which is a robust test.

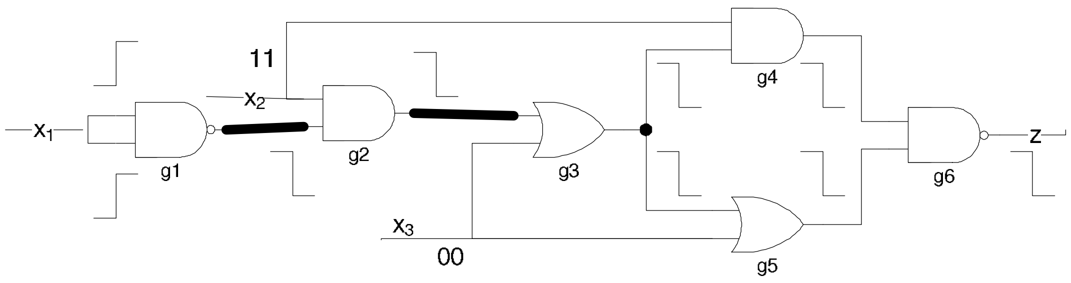


Figure 2.3. SDF test generation

2.5. Test Generation Models

This section reviews time expansion model, TEM[18] and double time-expansion-model, DTEM [19] that was extended from TEM.

2.5.1 Time Expansion Model

Time expansion model has been widely used as an approach of test generation of acyclic sequential circuits as the tests can be generated by applying combinational test generation to the time expansion model.

Definition 2.5.1. A topology graph is a directed graph $G = (V, A, r)$ where a vertex $v \in V$ denotes a combinational logic block which contains primary inputs/outputs and logic gates, and an arc $(u, v) \in A$ denotes a connection or a bus from u to v. Each arc has a label $r : A \mapsto Z^+$ (Z^+ denotes a set of non-negative integers), and $r(u, v)$ represents the number of registers on a connection (u, v).

Definition 2.5.2. Let $\mathcal{S}^{\mathcal{A}}$ be an acyclic sequential circuit and let $G = (V, A, r)$ be the topology graph of $\mathcal{S}^{\mathcal{A}}$. Let $E = (V_E, A_E, t, l)$ be a directed graph, where V_E is a set of vertices, A_E is a set of arcs, t is a mapping from V_E to a set of integers, and l is a mapping from V_E to a set of vertices V in G. If graph E satisfies the following four conditions, graph E is said to be a ***time expansion graph (TEG)*** of G.

- ***C1(Logic block preservation):*** The mapping l is surjective, i.e., $\forall v \in V$, $\exists u \in V_E \, s.t. \, v = l(u)$.

- ***C2(Input preservation):*** Let u be a vertex in E. For any direct predecessor $v(\in pre(l(u))$ of $l(u)$ in G, there exists a vertex u' in E such that $l(u') = v$ and $u' \in pre(u)$. Here, $pre(v)$ denotes the set of direct predecessors of v.

- ***C3(Time consistency):*** For any arc $(u, v)(\in A_E)$, there exists an arc $(l(u), l(v))$ such that $t(v) - t(u) = r(l(u), l(v))$.

- ***C4(Time uniqueness):*** For any vertices $u, v(\in V_E)$, if $t(u) = t(v)$ and if $l(u) = l(v)$, then the vertices u and v are identical, i.e., $u = v$.

Definition 2.5.3. Let $\mathcal{S}^{\mathcal{A}}$ be an acyclic sequential circuit, let $G = (V, A, r)$ be the topology graph of $\mathcal{S}^{\mathcal{A}}$, and let $E = (V_E, A_E, t, l)$ be a TEG of G. The combinational circuit $C_E(\mathcal{S}^{\mathcal{A}})$ obtained by the following procedure is said to be the ***time expansion model (TEM)*** of $\mathcal{S}^{\mathcal{A}}$ based on E.

1. For each vertex $u \in V_E$, let logic block $l(u)$ $(\in V)$ be the logic block corresponding to u.

2. For each arc $(u, v) \in A_E$, connect the output of u to the input of v with a bus in the same way as $(l(u), l(v))(\in A)$. Note that the connection corresponding to (u, v) has no register even if the connection corresponding to $(l(u), l(v))$ has a register (i.e. $r(l(u), l(v)) > 0$).

3. For a line or a logic gate in each logic block obtained by Step (1) and (2), if it is not reachable to any input of other logic blocks, then it is removed.

2.5.2 Double Time Expansion Model

Iwagaki et. al.[19] introduced ***double time expansion model (DTEM)*** to generate test sequences for all the testable transition faults. The idea of pattern dependency from this work is applied to introduce a test generation method that generates test sequences for all the testable robust and non-robust PDFs for acyclic sequential circuits by using SAF test generation method.

Definition 2.5.4. Let $\mathcal{S}^{\mathcal{A}}$ be an acyclic sequential circuit, and $C(\mathcal{S}^{\mathcal{A}})$ be a TEM of $\mathcal{S}^{\mathcal{A}}$. Then, a combinational circuit obtained by the following procedure is said to be a ***double time-expansion model (DTEM)*** $C^*(\mathcal{S}^{\mathcal{A}})$ of $\mathcal{S}^{\mathcal{A}}$.

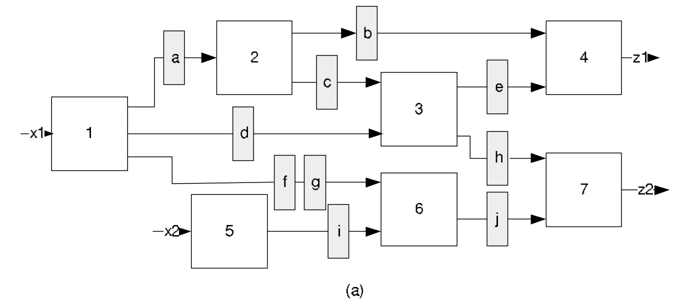
S1: Duplicate $C(\mathcal{S}^{\mathcal{A}})$ so that $C^*(\mathcal{S}^{\mathcal{A}})$ consists of two copies of $C(\mathcal{S}^{\mathcal{A}})$, namely $C^*_{V1}(\mathcal{S}^{\mathcal{A}})$ and $C^*_{V2}(\mathcal{S}^{\mathcal{A}})$.

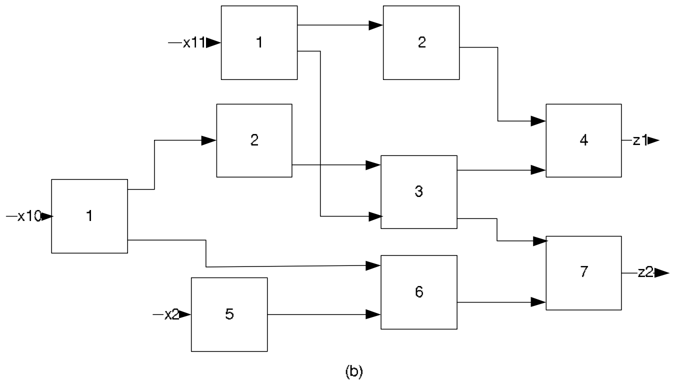
S2: Connect any pair of primary inputs u in $C^*_{V1}(\mathcal{S}^{\mathcal{A}})$ and v in $C^*_{V2}(\mathcal{S}^{\mathcal{A}})$ such that $t(v) - t(u) = 1$ and $l(u) = l(v)$, and feed a new primary input W into them.

Figure 2.4 shows an acyclic sequential circuit, its TEM and its DTEM.

2.6. Conclusion

Stuck-at fault model and path delay fault model are two representative fault models for logical faults and timing faults, representatively in testing. Both test generation for stuck-at faults and path delay faults and their relationship are discussed in the dissertation. Time expansion model (TEM) is a test generation model for acyclic sequential circuits. The time complexity of acyclic sequential circuits with stuck-at faults is discussed under TEM. The test generation for acyclic sequential circuits with path delay faults is discussed using extended TEM called double time expansion model or DTEM.





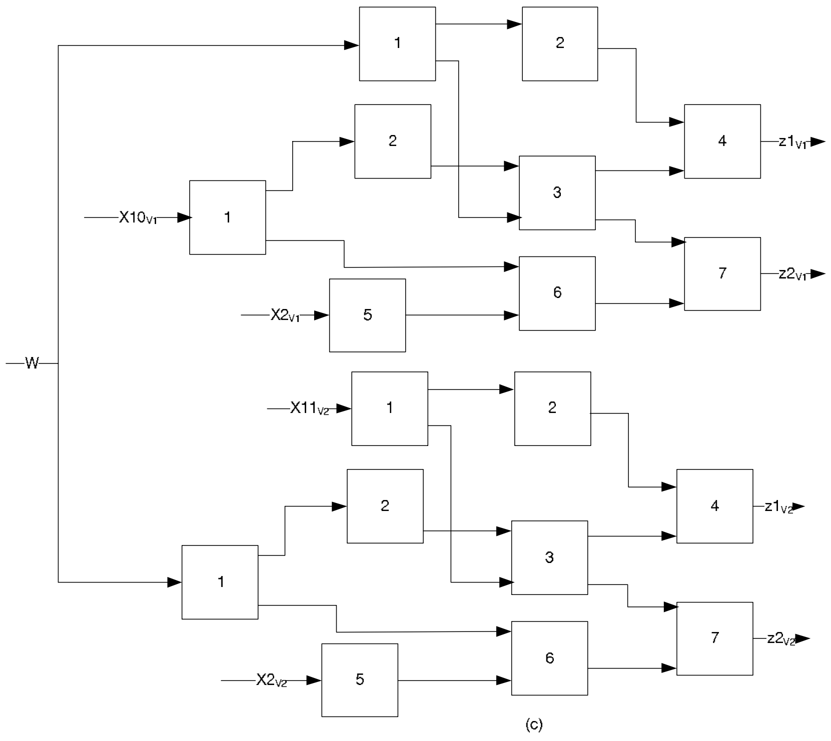


Figure 2.4. (a)An acyclic sequential circuit. (b) Its TEM. (c) Its DTEM.

Chapter 3

τ^k-Notation

3.1. Introduction

In this chapter, τ^k notation is defined using asymptotic notation. Generally, asymptotic notation is used to describe the asymptotic running time of an algorithm. This notation is also convenient for describing the worst-case running time of the test generation problem. Let $g(n)$ be a given function. The following describes briefly $\Theta(g(n))$ and $O(g(n))$. A function $f(n)$ belongs to the set $\Theta(g(n))$ if $g(n)$ is an asymptotically tight bound for $f(n)$. A function $f(n)$ belongs to the set $O(g(n))$ if $g(n)$ is an asymptotically upper bound for $f(n)$[22].

3.2. Definitions of τ^k-Notation

To facilitate the discussion, the time complexity of test generation problem is defined as follows.

P_C: ***Combinational Test Generation Problem***

Instance: A combinational circuit C and a fault f.
Question: Is there a test pattern to detect f in C?

P_S: ***Sequential Test Generation Problem***

Instance: A sequential circuit S and a fault f.
Question: Is there a test sequence to detect f in S?

P_α: ***Class α Test Generation Problem***

Instance: A sequential circuit S in α and a fault f.

Question: Is there a test sequence to detect f in S?

Definition 3.2.1. ***The time complexity of a problem P*** is the time complexity of the fastest algorithm for the problem P. Let $T_C(n)$, $T_S(n)$ and $T_\alpha(n)$ be the time complexity of P_C, P_S and P_α, respectively, where n is the size of the problem instance. $T_C(n)$, $T_S(n)$ and $T_\alpha(n)$ are also called ***test generation complexity*** for class C, class S and class α, respectively.

In this discussion, the following assumption about the combinational test generation complexity is used.
Assumption: The combinational test generation complexity is $\Theta(n^r)$ for some constant $r > 2$, where n is the size of the combinational circuit considered.

To further clarify the test generation complexity, τ^k notation is defined. $T_C(n)$ is considered as a basic unit of the time complexity of the test generation problem, therefore $\tau(n)$ is used to denote $T_C(n)$ in the following text, where $\tau(n) = T_C(n) = \Theta(n^r)$ for some constant $r > 2$.

Definition 3.2.2. $T(n)$ is τ^k***-equivalent*** if and only if $T(n) = \Theta(\tau^k(n))$ and τ^k***-bounded*** if and only if $T(n) = O(\tau^k(n))$, where $k > 0$.

Definition 3.2.3. Class α is τ^k***-equivalent*** if and only if $T_\alpha(n) = \Theta(\tau^k(n))$ and τ^k***-bounded*** if and only if $T_\alpha(n) = O(\tau^k(n))$, where $k > 0$.

3.3. Test Generation Complexity of Existing Classes of Sequential Circuits with Stuck-At Faults

The test generation problem for the existing easily testable classes of acyclic circuits in terms of τ^k notation gives a clearer picture of the test generation complexity.

3.3.1 Balanced Sequential Circuits

Let a directed graph $G = (V, A, H)$ represent a sequential circuit. The set V of vertices represents a set of clouds where each cloud is a maximal region of connected combinational logic such that its inputs are either primary inputs or outputs of registers and its outputs are either primary outputs or inputs to registers. The set A of arcs represents a set of connections between two clouds through a register. Arcs in $H \subseteq A$ represent HOLD registers. A sequential circuit is said to be a ***balanced sequential circuit*** if

1. G is acyclic;

2. $\forall v_i, v_j \in V$, all directed paths (if any) from v_i to v_j are of equal length;

3. $\forall h \in H$, if h is removed from G, the resulting graph is disconnected.

Theorem 3.3.1. *Balanced sequential circuits is τ-equivalent.*

Proof. [5] shows that every balanced sequential circuit can be transformed to its combinational equivalent by only replacing all registers in the circuit with delayless wires. The circuit transformation can be done in time $O(n)$. Let $T_B(n)$ denote the test generation complexity for balanced sequential circuits, where n is the size of the circuits. $T_B(n)$ consists of the circuit transformation time complexity and the test generation complexity for the combinational equivalent.

$$\begin{aligned} T_B(n) &= O(n) + \tau(n) \\ &= \Theta(\tau(n)). \end{aligned} \tag{3.1}$$

□

From Definition 3.2.3, the class of balanced sequential circuits is τ-equivalent.

3.3.2 Strongly Balanced Sequential Circuits

Let a directed graph $G = (V, A, w)$ represent a sequential circuit. V represents a set of clouds, where each cloud is a maximal region of connected combinational logic such that its inputs are either primary inputs or outputs of registers and its outputs are either primary outputs or inputs to registers. A represents a set of connections between two clouds. A weight, $w(a)$ on the arc $a = (v_i, v_j)$ equals to the number of registers between the corresponding clouds. A sequential circuit is a ***strongly balanced sequential circuit*** when the following conditions are satisfied.

1. G is acyclic;

2. $\forall v_i, v_j \in V$, all directed paths (if any) from v_i to v_j are of equal length;

3. There exists a function t from v to a set of integers such that $t(v_i) = t(v_j) + w(a)$ for $\forall a = (v_i, v_j)$.

Theorem 3.3.2. *Strongly balanced sequential circuits is τ-equivalent.*

Proof. Since the class of strongly balanced sequential circuits is a subclass of balanced sequential circuits, a given strongly balanced sequential circuit can be transformed to its combinational equivalent by replacing all registers in the circuit with delayless wires. This can be done in time $O(n)$. There is no logic duplication in the transformed circuit. Let $T_{SB}(n)$ denote the test generation complexity for strongly balanced sequential circuits, where n is the size of the circuits. $T_{SB}(n)$ consists of circuit transformation time complexity and the test generation complexity for the combinational equivalent of the sequential circuits.

$$\begin{aligned} T_{SB}(n) &= O(n) + \tau(n) \\ &= \Theta(\tau(n)). \end{aligned} \tag{3.2}$$

□

Therefore, strongly balanced sequential circuits is τ-equivalent.

3.3.3 Internally Balanced Sequential Circuits

According to [7], if a circuit resulting from operation 1 of the extended combinational transformation (C^*-transformation) on an acyclic sequential circuit is a balanced sequential circuit, then the circuit is regarded as an ***internally balanced sequential circuit***. In [7], the concept of *separable* is defined for branches of a primary input. The concept will be used in the definition of C^*-transformation. Suppose x is a primary input and x_i and x_j are branches of x. If no path exists such that a primary output z_k can be reached from x_i and x_j over equal depth paths, then x_i and x_j are called *separable*. Equal depth paths are the paths where the number of flip-flops included in each of the paths is same.

C^*-transformation consists of the following two operations.

1. For a primary input with fanout branches, the set of fanout branches of that primary input is denoted by X. Let us obtain the smallest partition of X which satisfies the following statement: If branches x_i and x_j belong to different blocks $X(i)$, $X(j)$ of partition $\Pi(x_i \subset X(i), x_j \subset X(j), X(i) \neq X(j))$, then x_i and x_j are separable. Each partitioned block is provided with a new primary input separated from the original primary input;

2. All flip-flops are replaced by wires.

Theorem 3.3.3. *Internally balanced sequential circuits is τ-equivalent.*

Proof. [7] has proved that if a fault f in an internally balanced sequential circuit S can be tested then the corresponding fault f_C in $C^*(S)$ can be tested. And, there is no logic duplication in C^*-transformation. $C^*(S)$ can be done in time $O(n^2)$. Let $T_{IB}(n)$ denotes the test generation complexity for internally balanced sequential circuits, where n is the size of the circuits. $T_{IB}(n)$ includes C^*-transformation time complexity and the test generation complexity for the combinational equivalent. Then, the test generation time complexity is

$$T_{IB}(n) \quad = \quad O(n^2) + \tau(n) = \Theta(\tau(n)). \qquad (3.3)$$

□

From Definition 3.2.3, the class of internally balanced sequential circuits is τ-equivalent.

3.3.4 Acyclic Sequential Circuits

An ***acyclic sequential circuit*** is a sequential circuit without feedback. The test generation complexity for this class is not τ-equivalent under the test generation model called time expansion model or TEM [18].

Lemma 3.3.4. Let u and v be arbitrary logic blocks of an acyclic sequential circuit where $u \in pre(v)$. The logic block u will be mapped to q different logic blocks in TEM if there are p connections between logic block u and v with q different labels where $p \geq q$.

Proof. Let v' be the corresponding logic block of v in TEM ($l(v') = v$) and let $r_i(u, v)$ be labels for each connection (u, v) where $0 \leq i \leq q$. Let also u'_j denote each corresponding logic block of u in TEM. From the condition of input preservation and time consistency [18],

$$t(u'_j) = t(v') - r_i(u, v) \tag{3.4}$$

Since $1 \leq i \leq q$, the range of j is also $1 \leq j \leq q$. Since $u = l(u'_j)$, the lemma is proved. □

Theorem 3.3.5. *There exists an acyclic sequential circuit whose test generation complexity represented by TEM is not τ-equivalent.*

Proof. Let an acyclic sequential circuit, S_a have a structure represented by a topology graph $G = (V, A, r)$ as follows.

- $V = \{u, v\}$ where $u \in pre(v)$ and $A = \{a_i \mid 0 \leq i \leq d\}$;
- $r_i(u, v) = i$ for $0 \leq i \leq d$ where $r_i(u, v)$ represents a label on arc a_j and d is the sequential depth of S_a, which is not a constant.

Let n_0 and n_1 be the size of the logic block represented by vertices u and v, respectively where $n_0 = n_1 = \frac{n}{2}$ as shown in Figure 3.1.

From Lemma 3.3.4, vertex u in the topology graph is mapped to $(d + 1)$ different vertices in TEM as shown in Figure 3.2. Note that no logic portion can be removed. Thus, the size of the combinational equivalent of the acyclic sequential circuit represented in TEM is

$$\begin{aligned} N &= \frac{n}{2} \times (d + 1) + \frac{n}{2} \\ &= \frac{d \times n}{2} + n \end{aligned} \tag{3.5}$$

Since a combinational test generation is applied on the time expansion model to generate tests, the test generation complexity of the acyclic sequential circuit is

$$\begin{aligned} T_A = \tau(N) &= \tau(\frac{d \times n}{2} + n) \\ &= \tau(d \times n) \\ &= \Theta(d^r \times n^r) \neq \Theta(n^r) \\ & \text{for some constant } r. \end{aligned} \tag{3.6}$$

The equation 3.6 proves the theorem. □

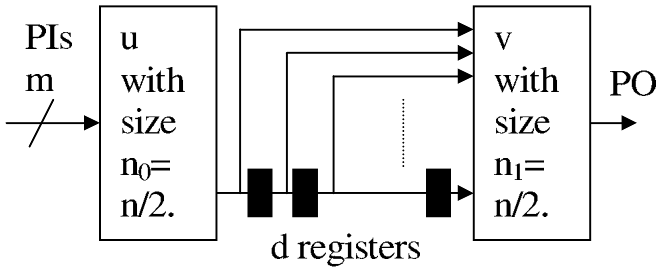


Figure 3.1. Structure of S_a.

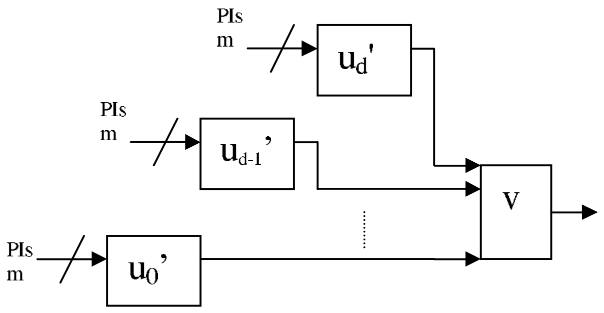


Figure 3.2. TEM of S_a.

However, there might be other test generation models for acyclic sequential circuits besides TEM. "Is T_A τ-equivalent?" remains an open question. No one has proved the answer is "Yes" but it might probably be "No" since existing studies show that the logic duplication might happen for at most d time frames in the test generation problem, where d is the sequential depth under some test generation models. Therefore, we have the following conjecture and theorem.

Conjecture 3.3.1. *The class of acyclic sequential circuits is not τ-equivalent.*

Theorem 3.3.6. *The class of acyclic sequential circuits is τ^2-bounded.*

Proof. [12] shows that the number of time frames in which logic duplication might take place is at most d, where d is the sequential depth. Therefore, the size of the transformed circuit to be used for the test generation is at most $n \times (d + 1)$. Since $d \leq n$, the test generation complexity of acyclic sequential circuits is $O(\tau^2(n))$. □

The practical observation shows that the test generation of acyclic sequential circuits is close to $\Theta(\tau(n))$ instead of $\Theta(\tau^2(n))$ bound. In other words, its test generation is still not very hard. Figure 3.3 shows the relationships among existing classes of acyclic sequential circuits.

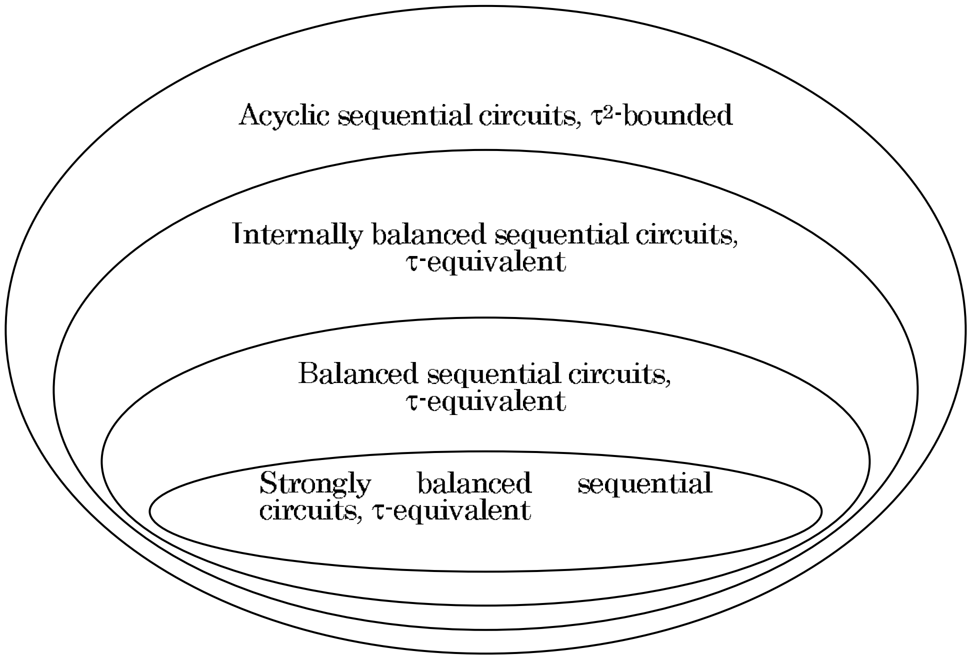

Acyclic sequential circuits, τ^2-bounded
Internally balanced sequential circuits, τ-equivalent
Balanced sequential circuits, τ-equivalent
Strongly balanced sequential circuits, τ-equivalent

Figure 3.3. Classes of acyclic sequential circuits.

3.4. Conclusion

τ^k notation has been introduced. Using τ^k notation, balanced sequential circuits, strongly balanced sequential circuits and internally balanced sequential circuits are shown τ-equivalent in test generation for stuck-at faults. Acyclic sequential circuits are τ^2-bounded in test generation for stuck-at faults. Under time expansin model (TEM), it is proved that the class of acyclic sequential circuits is not τ-equivalent. τ^k notation is also used for discussing the test generation complexity of other classes of sequential circuits with stuck-at faults and path delay faults in the following chapters.

Chapter 4

Classification of Sequential Circuits with Stuck-At Faults Based on Combinational Test Generation Complexity

4.1. Introduction

In this dissertation, a class is considered to be *easily testable* if its test generation complexity is τ^2-bounded. In other words, τ^2-bounded classes and τ-equivalent classes are easily testable. In this chapter, several classes of easily testable sequential circuits are introduced. The test generation for stuck-at faults in some cyclic sequential circuits is τ-equivalent. Also, synthesis for testability (SFT) can be used to augment a given sequential circuit into an easily testable circuit belonging to one of these classes. Case studies are done to compare the test application time and area overhead of these augmented circuits with their corresponding scan designed circuits.

4.2. Classes of Easily Testable Sequential Circuits

Generally, the test generation problem of a cyclic sequential circuit is modeled by an iterative logic array that consists of several time frames so that it can be solved by combinational test generation techniques. The model is shown in Figure 4.1. The test generation problem involves the following three steps.

1. Derivation of the excitation state;
2. State justification for i time frames; and
3. State differentiation for j time frames.

Generally, backtracks may occur between the three steps. For a given fault, step 1 is performed to obtain an excitation state for state justification and state differentiation. If state justification or state differentiation fails, step 1 is performed again to get a different excitation state for state justification and state differentiation. Logic duplication of the combinational part takes place at every time frame for state justification and state differentiation. In the worst case, i and j equal 2^p, where p is the number of memory elements. These factors result in high complexity for test generation of cyclic sequential circuits.

However, there exist classes of sequential circuits which are τ-equivalent or τ^2-bounded and include some cyclic sequential circuits. In such classes, it is guaranteed that any excitation state can be justified and the fault effect of any activated fault can be propagated to a primary output. Since the derivation of the excitation state is done by the test generation on the combinational part at time frame 0, the time complexity $T_E(n)$ is always τ-equivalent. Therefore, if the state justification and state differentiation can be reduced to the problems which are τ^2-bounded or τ-equivalent or with less time complexity, the circuits become easily testable. Let T_J, T_E and T_D denote the time complexity of state justification, derivation of excitation state and state differentiation, respectively. The test generation complexity for a class of easily testable sequential circuits,

$T_S(n)$ is

$$\begin{aligned} T_S(n) &\leq T_E(n) + T_J + T_D \qquad (4.1) \\ &= \tau(n) + T_J + T_D \end{aligned}$$

The following sub-sections introduce several new classes of easily testable sequential circuits, which cover some cyclic sequential circuits.

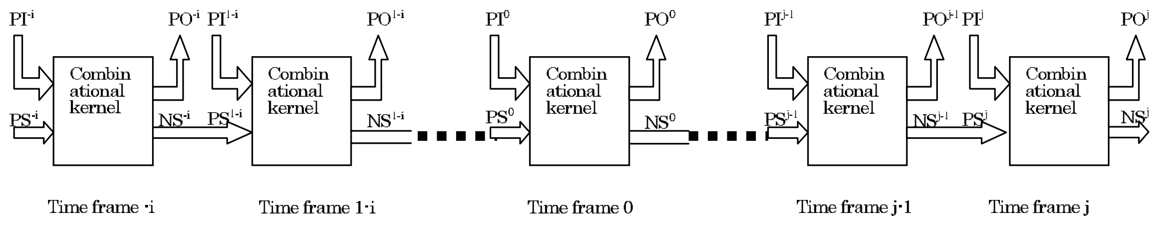


Figure 4.1. Iterative logic array model.

4.2.1 l-Length-Bounded Testable Circuits

The number of time frames expanded by the state justification and state differentiation accounts for the length of a test sequence. In this sub-section, a new class of easily testable sequential circuits called l-length-bounded testable circuits, the test sequence length of which is bounded so that the class becomes easily testable, is introduced.

Definition 4.2.1. A sequential circuit S is ***l-length-bounded testable*** with respect to a fault set F if the following conditions are satisfied.

1. For any state s_i, there exists a state justification sequence of length at most l;

2. For any pair of states (s_j, s_{jf}), there exists a state differentiation sequence of length at most l, where s_j is a fault-free state and s_{jf} is a state in the faulty sequential circuit with a fault $f \in F$.

Theorem 4.2.2. *l-length-bounded testable circuits is τ^2-bounded if l is $O(n)$, where n is the size of the sequential circuits.*

Proof. To generate a test sequence for a given fault f in a l-length-bounded testable circuit, firstly combinational test generation is performed at time frame 0 to derive an excitation state s_i so that f is excited and the fault effect is propagated to a next-state line or a primary output. Second, state justification sequence is generated for the excitation state s_i and lastly if the fault effect is not propagated to a primary output by the first two steps, fault propagation sequence is generated for a pair of (s_j, s_{jf}), where s_{jf} is the next state of time frame 0 and s_j is the corresponding state in the fault-free circuit. Condition 1 of Definition 4.2.1 guarantees that for any state s_j, there exists a state justification sequence and Condition 2 of Definition 4.2.1 guarantees that for any pair of states (s_j, s_{jf}), there exists a fault propagation sequence. Consequently, the test generation for a given fault is a one-way procedure consisting of the three steps mentioned above, which means no backtracks occur between the steps. Condition 1 of Definition 4.2.1 implies that the state justification expands the combinational part for at most l time frames to justify the excitation state s_i. Therefore, the justification is performed on the duplication of combinational part of size at most $l \times n$.

The state justification T_J for an excitation state s_i can be reduced into a combinational test generation problem of the iterative logic array (Figure 4.2(b)) that satisfies the following conditions.

- It consists of at most l time frames;
- each next state line of the last time frame is connected to an input of an AND gate. The next state line is connected to the AND gate through an inverter if the bit signal of s_i at the line is 0; and
- the output of the AND gate has stuck-at 1 fault.

Therefore, the time complexity of the state justification T_J is τ-bounded ($T_J(N_J) = O(\tau(N_J))$) where N_J is the size of the state justification problem. The time complexity of the state justification $T_J(l \times n)$ is

$$T_J(l \times n) = O(\tau(l \times n)). \qquad (4.2)$$

Condition 2 of Definition 4.2.1 implies that the state differentiation expands the combinational part for at most l time frames in order to propagate the fault effect from time frame 0 to a primary output. Similar to the problem of the state justification, the fault propagation T_D for an activated fault can be reduced into a combinational test generation problem of the iterative logic array (Figure 4.2(c)) that satisfies the following conditions.

- It consists of at most l time frames; and
- it has the multiple faults including a fault in each time frame that corresponds to the activated fault in the excitation time frame and a stuck-at f_{bit} fault in each present-state line of the first time frame that receives the fault effect of the activated fault in excitation time frame. The value of f_{bit} is 0 (resp. 1) if present-state line is 0 (resp. 1) in s_{jf}.

This test generation problem with multiple faults can be further reduced into a combinational test generation problem of the iterative logic array that consists of at most l time frames with a single stuck-at fault at a primary input connected to ground (signal value 0) as shown in Figure 4.2(d) by using the transformation introduced in Chapter 2. Therefore the time complexity of the fault propagation T_D is τ-bounded ($T_D(N_D) = O(\tau(N_D))$) where N_D is the size of the state differentiation problem. The time complexity of the state differentiation $T_D(l \times n)$ is

$$T_D(l \times n) = O(\tau(l \times n)). \tag{4.3}$$

Let $T_{LBT}(n)$ be the test generation complexity for l-length-bounded testable circuits and l be $O(n)$. Then,

$$\begin{aligned} T_{LBT}(n) &\leq T_E(n) + T_J(l \times n) + T_D(l \times n) \\ &= \tau(n) + O(\tau(l \times n)) + O(\tau(l \times n)) \\ &= \tau(n) + O(\tau^2(n)) + O(\tau^2(n)) \text{ for } l \text{ is } O(n) \\ &= O(\tau^2(n)). \end{aligned} \tag{4.4}$$

Hence, the test generation complexity for the l-length-bounded testable circuits is τ^2-bounded if l is $O(n)$. Note that justification (resp. propagation) is a subproblem of combinational test generation. Therefore, the time complexity of justification (resp. propagation) is $O(\tau(n))$. □

4.2.2 l-Length-Bounded Validity-Identifiable Circuits

Valid states are the reset state and all states in the machine that are reachable from the reset state using an input sequence [24]. There are some sequential circuits where not all the states are valid states. Based on this characteristic, the class of l-length-bounded validity-identifiable circuits is defined.

Definition 4.2.3. A sequential circuit S is ***l-length-bounded validity-identifiable*** with respect to a fault set F if the following conditions are satisfied.

1. There exists a combinational circuit of size $O(n)$ called validity checker (Figure 4.3) that can identify the validity of states, where n is the size of the sequential circuits;

2. For any valid state s_i, there exists a state justification sequence of length at most l from initial state S_0;

3. For any pair of states (s_j, s_{jf}), there exists a state differentiation sequence of length at most l, where s_j is a fault-free valid state and s_{jf} is a state in the faulty sequential circuit with a fault $f \in F$.

Theorem 4.2.4. *The class of l-length-bounded validity-identifiable circuits is τ^2-bounded if l is $O(n)$, where n is the size of the sequential circuits.*

Proof. From Condition 1 of Definition 4.2.3, prior to the test generation of a l-length-bounded validity-identifiable circuit a combinational circuit of size $O(n)$ that can identify the validity of the states is constructed and embedded in the combinational part C of the l-length-bounded validity-identifiable circuit such that any excitation state s_i derived by the combinational test generation is a valid state. The transformed combinational part as shown in Figure 4.3 is denoted by C', which has size $O(2 \times n)$. It can be shown that a fault is testable in C with a valid state if and only if the fault is testable in C'. To generate a test sequence for a given fault f in a l-length-bounded validity-identifiable circuit, firstly combinational test generation is performed on C' at time frame 0 to derive an excitation state s_i so that f is excited and propagated to a next-state line or a primary output. Note that s_i is a valid state. Second, state justification sequence is generated for the excitation state s_i and lastly if the fault effect is not propagated to a primary output by the first two steps, state differentiation sequence is generated for a pair of states (s_j, s_{jf}), where s_{jf} is the next state of time frame 0 and s_j is the corresponding state in fault-free circuit. Condition 2 in Definition 4.2.3 guarantees that for any valid state s_i, there exists a state justification sequence and Condition 3 in Definition 4.2.3 guarantees that for any pair of states (s_j, s_{jf}), there exists a state differentiation sequence. Consequently, the test generation is a one-way procedure consisting of the above three steps, which means no backtracks occur between the steps. Let $T_E(2 \times n)$ be the test generation complexity of C'. The derivation of excitation state can be done in

$$\begin{aligned} T_E(2 \times n) &= \tau(2 \times n) \\ &= \tau(n). \end{aligned} \tag{4.5}$$

From Condition 2 of Definition 4.2.3,

$$T_J = O(l). \tag{4.6}$$

From Condition 3 of definition 4.2.3,

$$T_D = O(l). \tag{4.7}$$

Let $T_{LBVI}(n)$ be the test generation complexity for l-length-bounded validity-identifiable circuits. Then,

$$\begin{aligned} T_{LBVI}(n) &\leq T_E(2n) + T_J(l \times n) + T_D(l \times n) \\ &= \tau(n) + O(\tau^2(n)) + O(\tau^2(n)) \\ &= O(\tau^2(n)), \text{which is } \tau^2\text{-bounded.} \end{aligned} \tag{4.8}$$

□

4.2.3 *t*-Time-Bounded Testable Circuits

Classification of sequential circuits is also considered from the aspect of time dimension instead of length dimension. In this section, another new class of sequential circuits called t-time-bounded testable circuits is introduced. The t-time-bounded testable circuit is defined as follows.

Definition 4.2.5. A sequential circuit S is ***t-time-bounded testable*** with respect to a fault set F if the following conditions are satisfied.

1. For any state s_i, there exists a state justification sequence which can be obtained in time $O(t)$;

2. For any pair of states $(s_j,\ s_{jf})$, there exists a state differentiation sequence which can be obtained in time $O(t)$, where s_j is a fault-free state and s_{jf} is a state in the faulty sequential circuit with a fault $f \in F$.

Theorem 4.2.6. *The class of t-time-bounded testable circuits is τ-equivalent (resp. τ^2-bounded) if t is $\tau(n)$ (resp. $\tau^2(n)$), where n is the size of the sequential circuits.*

Proof. To generate a test sequence for a given fault f in a t-time-bounded testable circuit, firstly combinational test generation is performed at time frame 0 to derive an excitation state so that f is excited and propagated to a next-state line or a primary output. Second, state justification sequence is generated for the excitation state s_i and lastly if the fault effect is not propagated to a primary output by the first two steps, fault propagation sequence is generated for the pair of states (s_j, s_{jf}), where s_{jf} is the next state of time frame 0 and s_j is the corresponding state in fault-free circuit. Condition 1 of Definition 4.2.5 guarantees that for any state s_i, there exists a state justification sequence and Condition 2 of Definition 4.2.5 guarantees that for any pair of states (s_j, s_{jf}), there exists a fault propagation sequence. Consequently, the test generation for a given fault is a one-way procedure consisting of the above three steps, which means no backtracks occur between the steps. From Condition 1 of Definition 4.2.5,

$$T_J = O(t). \tag{4.9}$$

From Condition 2 of Definition 4.2.5,

$$T_D = O(t). \tag{4.10}$$

Let $T_{TBT}(n)$ be the test generation complexity for t-time-bounded testable circuits. Then,

$$\begin{aligned} T_{TBT}(n) &\leq T_E(n) + T_J + T_D \\ &= \tau(n) + O(t) + O(t) \\ &= \tau(n) + O(\tau(n)) + O(\tau(n)) \text{ for } t \text{ be } \tau(n) \qquad (4.11) \\ &= O(\tau(n)). \end{aligned}$$

Hence, the test generation complexity for t-time-bounded testable circuits is τ-equivalent if t is $\tau(n)$. In the case where t is $\tau^2(n)$,

$$\begin{aligned} T_{TBT}(n) &\leq T_E(n) + T_J + T_D \qquad (4.12) \\ &= \tau(n) + O(t) + O(t) \\ &= \Theta(\tau(n)) \text{ if } t = \tau(n) \text{ or} \\ &\quad\; O(\tau^2(n)) \text{ if } t = \tau^2(n). \end{aligned}$$

Therefore, the class of t-time-bounded testable circuits is τ-equivalent if $t = \tau(n)$ and τ^2-bounded if $t = \tau^2(n)$. □

4.2.4 t-Time-Bounded Validity-Identifiable Circuits

The test generation of t-time-bounded validity-identifiable circuits is also bounded by the time complexity. However, different from the t-time-bounded testable circuits, not all the states of a t-time-bounded validity-identifiable circuit are valid states.

Definition 4.2.7. A sequential circuit S is ***t-time-bounded validity-identifiable*** with respect to a fault set F if the following conditions are satisfied.

1. There exists a combinational circuit of size $O(n)$ called validity checker (Figure 4.3) that can identify the validity of states, where n is the size of the sequential circuits;

2. For any valid state s_i, there exists a state justification sequence from initial state S_0 which can be obtained in time $O(t)$;

3. For any pair of states (s_j, s_{jf}), there exists a state differentiation sequence which can be obtained in time $O(t)$, where s_j is a fault-free valid state and s_{jf} is a state in the faulty sequential circuit with a fault $f \in F$.

Theorem 4.2.8. *The class of t-time-bounded validity-identifiable circuits is τ-equivalent (τ^2-bounded) if t is $\tau(n)$ ($\tau^2(n)$), where n is the size of the sequential circuits.*

Proof. From Condition 1 of Definition 4.2.7, prior to the test generation of a t-time-bounded validity-identifiable circuit a combinational circuit of size $O(n)$ that can identify the validity of the states is constructed and embedded in the combinational part C of the time-bounded validity-identifiable circuit such that any excitation state s_i derived by the combinational test generation is a valid state. The transformed combinational part as shown in Figure 4.3 is denoted by C', which has size $O(2 \times n)$. It can be shown that a fault is testable in C with a valid state if and only if the fault is testable in C'. To generate a test sequence for a given fault f in a time-bounded validity-identifiable circuit, firstly combinational test generation is performed on C' at time frame 0 to derive an excitation state s_i so that f is excited and propagated to next-state lines or primary outputs. Note that s_i is a valid state. Second, state justification sequence is generated for the excitation state s_i and lastly if the fault effect is not propagated to primary output by the first two steps, fault propagation sequence is generated for a pair of states (s_j, s_{jf}), where s_{jf} is the next state of time frame 0 and s_j is the corresponding state in fault-free circuit. Condition 2 in Definition 4.2.7 guarantees that for any valid state s_i, there exists a state justification sequence and Condition 3 in Definition 4.2.7 guarantees that for any pair of states (s_j, s_{jf}), there exists a fault propagation sequence. Consequently, the test generation is a one-way procedure consisting of the above three steps, which means no backtracks occur between the steps. Let $T_E(2 \times n)$ be the test generation complexity of C'. The derivation of excitation state can be done in

$$\begin{aligned} T_E(2 \times n) &= \tau(2 \times n) \\ &= \tau(n). \end{aligned} \tag{4.13}$$

From Condition 2 of Definition 4.2.7,

$$T_J = O(t). \tag{4.14}$$

From Condition 3 of Definition 4.2.7,

$$T_D = O(t). \tag{4.15}$$

Let $T_{TBVI}(n)$ be the test generation complexity for t-time-bounded validity-identifiable circuits. Then,

$$\begin{aligned} T_{TBVI}(n) &\leq T_E(2n) + T_J + T_D \\ &= \tau(n) + O(t) + O(t) \\ &= \Theta(\tau(n)) \text{ if } t = \tau(n) \text{ or} \\ &\quad O(\tau^2(n)) \text{ if } t = \tau^2(n) \end{aligned} \tag{4.16}$$

Therefore, the class of t-time-bounded validity-identifiable circuits is τ-equivalent if $t = \tau(n)$ and τ^2-bounded if $t = \tau^2(n)$. □

4.3. Examples of Easily Testable Sequential Circuits

Although several classes of easily testable sequential circuits have been introduced in the previous section, the definitions are too general to easily categorize a given circuit into those classes. Therefore, this section introduces two special subclasses, namely counter-cycle finite state machine realizations and state-shiftable finite state machine realizations.

4.3.1 Counter-Cycle Finite State Machine Realizations

A ***counter-cycle finite state machine realization*** satisfies the following conditions.

1. The number of valid states is in $O(n)$ and there exists a validity checker of

size $O(n)$.

2. There exists an input symbol ϵ that strongly connects all valid states accordingly in a linear feedback shift register (LFSR) counter with a feedback polynomial.

3. Column ϵ (counter-cycle operation) is realized by AND-OR logic shown in Figure 4.5(a) (resp. NOT-OR-AND logic shown in Figure 4.5(b)) feeding each flip-flop such that

    - the combinational logic assigns 0 (resp. 1) to the input of the OR gate (resp. AND gate) when the input combination corresponds to column ϵ;
    - a primary input is fed to an input of the AND gate (resp. NOT gate that connected to an input of OR gate) and the LFSR counter logic is connected to the other input of the AND gate (resp. OR gate).

4. All flip-flops are resettable and the flip-flop that represents the most significant bit of the state is observable.

Figure 4.4(a) shows an example of a counter cycle with 6 states while Figure 4.6(a) illustrates the realization of counter-cycle FSM.

Theorem 4.3.1. *The class of counter-cycle FSM realizations is τ-equivalent.*

Proof. To generate a test for a counter-cycle FSM realization, first an excitation state is derived in time $\Theta(\tau(n))$. Then, the excitation state is justified by an input sequence of constant value ϵ and length at most p from the reset state, where p is the number of valid states. Lastly, the faulty next state generated after the derivation of excitation state is distinguished from the fault-free next state by an input sequence of constant value ϵ and length at most p. Both justification and differentiation are done by forward implication. Since $p = O(n)$, the time complexities of justification and differentiation are $O(n^2)$, respectively, where n is the size of the circuit. The class of counter-cycle FSM realizations is a subclass of t-time-bounded validity-identifiable circuits, where $t = O(n^2)$. □

4.3.2 State-Shiftable Finite State Machine Realizations

Another class of easily testable sequential circuits called ***state-shiftable FSM*** is introduced. A state-shiftable finite state machine [23] with m states is a machine that possesses

1. transfer sequences of length at most $\lceil log_2 m \rceil$ to carry the machine from state s_0 to state s_i for all i, and
2. distinguishing sequences of length $\lceil log_2 m \rceil$, which are arbitrary input sequences consisting of two input symbols.

A sequential circuit that is realized from the state-shiftable finite state machine is called state-shiftable finite state machine realization.

Theorem 4.3.2. *The class of state-shiftable FSM realizations is τ-equivalent if the following conditions are satisfied.*

1. *The FSM contains a two-column submachine equivalent to a binary shift register, where the input symbols of the two columns are denoted by ϵ_0 and ϵ_1, respectively;*
2. *When input combination corresponds to column ϵ_0 (resp. ϵ_1), flips-flops are in shifting operation with bit 0 (resp. bit 1) being shifted into the flip-flop that represents the least significant bit of the state.*
3. *Columns ϵ_0 and ϵ_1 (shifting operation) are realized by an AND-OR logic (Figure 4.5(a)) connected to each flip-flop such that*
   - *the combinational logic assigns 0 to the OR gate when the input combination corresponds to either column ϵ_0 or ϵ_1.*
   - *a primary input is fed to an input of the AND gate and the output of a flip-flop is connected to the other input of the AND gate.*
4. *The flip-flop that represents the most significant bit of the state is observable.*

The realization is shown in Figure 4.6(b). As an example, the transitions happening in columns ϵ_0 and ϵ_1 of the SSFSM of degree 2 are shown in Figure 4.4(b).

Proof. To generate a test for a state-shiftable FSM realization, first an excitation state is derived in time $\Theta(\tau(n))$. Then, the excitation state is justified by an input sequence which consists of ϵ_0 and ϵ_1. The input sequence has length at most p where $p = log_2 m = O(n)$. Lastly, the faulty next state generated after the derivation of excitation state is distinguished from the fault-free next state by an input sequence which consists of ϵ_0 and ϵ_1. The input sequence has length at most p. Both justification and differentiation are done by forward implication. Since $p = O(n)$, the time complexities of justification and differentiation are $O(n^2)$, respectively, where n is the size of the circuit. The class of state-shiftable FSM realizations is a subclass of t-time-bounded testable circuits, where $t = O(n^2)$. $\square$

4.4. Synthesis for Testability Method

In synthesis for testability, testability is considered during the synthesis process itself. Since state-shiftable FSM realizations and counter-cycle FSM realizations are cyclic sequential circuits defined at the FSM level and gate structural level that depends on the synthesis method, a given sequential design at FSM can be augmented into one of the easily testable classes using synthesis for testability method during synthesis instead of design for testability method after synthesis. Note that DFT considers the structure of the circuit kernel while SFT considers the whole circuit at FSM level in augmenting a given circuit into an easily testable circuit.

4.4.1 SFT for Counter-Cycle FSM Realizations and State-Shiftable FSM Realizations

To synthesize a given design into a counter-cycle FSM realization, the following steps are implemented.

1. Add at most one column of ϵ into the state table of the design so that the design becomes a counter-cycle FSM.

2. Perform state encoding and logic minimization in the synthesis.

3. Realize column ϵ as in Condition 3 of the definition of counter-cycle FSM realizations and add a primary output to the flip-flop which is the most significant.

The following shows the SFT method to synthesize a given design into a state-shiftable FSM realization.

1. Add at most two columns of ϵ_0 and ϵ_1 into the state table of the design so that the design becomes state shiftable FSM.

2. Perform state encoding and logic minimization in the synthesis.

3. Realize columns ϵ_0 and ϵ_1 as in Condition 3 of Theorem 4.3.2 and add a primary output to the flip-flop which is the most significant.

4.4.2 Case study: Comparison between SFTs and DFTs

Case studies are conducted on five MCNC [25] benchmark finite state machines, namely bbsse (16 states), cse (16 states), dk16 (27 states), opus (10 states) and tma (20 states). State encoding and logic minimization for these designs were performed using Synopsys Design Analyzer. Synopsys TetraMax tool is used to run the test generation on the circuits.

Original circuits of each benchmark do not have complete fault efficiency except opus. bbsse, cse, dk16 and tma have 96.56%, 99.42%, 99.72% and 99.52%, respectively. Since full scan designed circuits are also easily testable circuits, state-shiftable FSM realizations and counter-cycle FSM realizations are compared with full scan designed circuits. Table 4.1 indicates the result on the comparison of the area overhead (number of gates) between the scan design, counter-cycle FSM realizations and state-shiftable FSM realizations. The values in the parenthesis are the overhead in percentage. One of the columns of tma corresponds to column ϵ_1 of a state-shiftable FSM. Therefore, only one column is added to the state table during SFT, thus reducing the area overhead. State-shiftable FSM realizations have less area overhead compared to counter-cycle FSM realization. Comparing the state-shiftable FSM realizations and full scan design, the state-shiftable FSM realizations of cse, dk16 and tma have less overhead. The flip-flops used to realize counter-cycle FSMs are asynchronous flip-flops. The logic complexity of these flip-flops explains in part that counter-cycle FSM realizations have larger area overhead. Moreover, the counter-cycle FSM realizations of bbsse and cse need five flip-flops, which is one extra flip-flop compared to the state-shiftable FSM realizations. Extra XOR gates are necessary to realize the LFSR counter-cycle operation while no extra gates are needed for shifting operation in state-shiftable FSM.

Table 4.2 presents the result on the comparison of the test application time (in clock cycles) among the scan designs, counter-cycle FSM realizations and state-shiftable FSM realizations. From the result, state shiftable FSM realizations have shorter test application time compared to full scan designs and counter-cycle FSM realizations. The justification and differentiation sequence of counter-cycle FSM is at most the number of valid states while that of state-shiftable FSM is at most the logarithm of the number of states. In the experiment, the number of valid states of counter-cycle FSM are higher than the logarithm of the number of states in state-shiftable FSM. Table 4.3 shows that the circuits in scan techniques and our method have complete fault efficiency while the original circuits do not.

Table 4.1. Experimental result: area overhead(gate count)

B/mark	Ori.(%)	FS(%)	CCFSM(%)	SSFSM(%)
bbsse	130	146(12.3)	191(46.9)	150(15.4)
cse	205	221(7.8)	241(17.6)	215(4.9)
dk16	215	235(9.3)	287(33.5)	215(0)
opus	92	108(17.4)	140(52.2)	131(42.4)
tma	205	225(9.8)	269(31.2)	206(0.5)

Table 4.2. Experimental result: test application time(clock cycles)

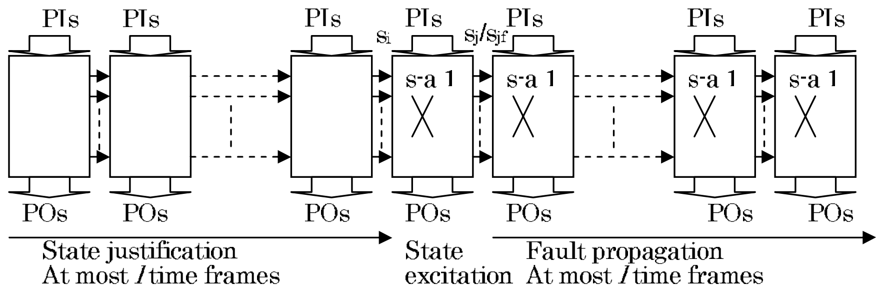
B/mark	Original	FS	CCFSM	SSFSM
bbsse	161	294	459	233
cse	359	434	766	458
dk16	421	455	1100	335
opus	167	184	204	179
tma	342	461	968	436

Table 4.3. Experimental result: fault efficiency(%)

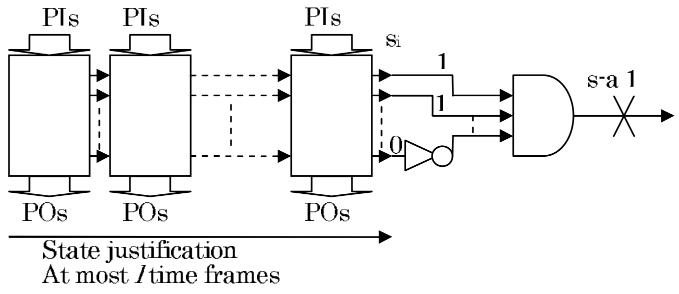
B/mark	Original	FS	CCFSM	SSFSM
bbsse	96.36	100	100	100
cse	99.44	100	100	100
dk16	99.62	100	100	100
opus	99.71	100	100	100
tma	99.08	100	100	100

4.5. Conclusion

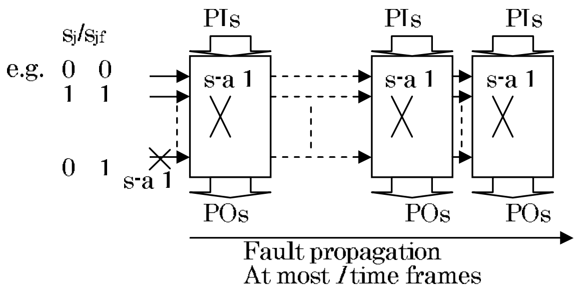
Several classes of easily testable cyclic sequential circuits have been introduced. These include l-length-bounded testable circuits and l-length-bounded validity-identifiable circuits with $l = O(n)$, t-time-bounded testable circuits and t-time-bounded validity-identifiable circuits with $t = \tau(n)$ or $\tau^2(n)$, state-shiftable FSM realizations and counter-cycle FSM realizations. The case studies indicated that state-shiftable FSM realization can be better than its corresponding counter-cycle FSM realization and its corresponding full scan designed circuit in certain cases while full scan designed circuit has better result than its corresponding counter-cycle FSM realization in certain cases.



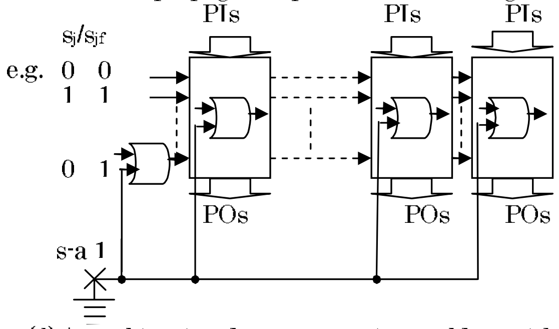
(a) Iterative logic array of a l-length-bounded testable circuits.



(b) A combinational test generation problem which is equivalent to the state justification problem of the l-length-bounded testable circuits.



(c) A combinational test generation problem with multiple faults that is equivalent to the fault propagation problem of the l-length-bounded testable circuits.



(d) A combinational test generation problem with single fault which is equivalent to the fault propagation problem of the l-length-bounded testable circuits.

Figure 4.2. The problems of state justification and fault propagation of l-length-bounded testable circuits.

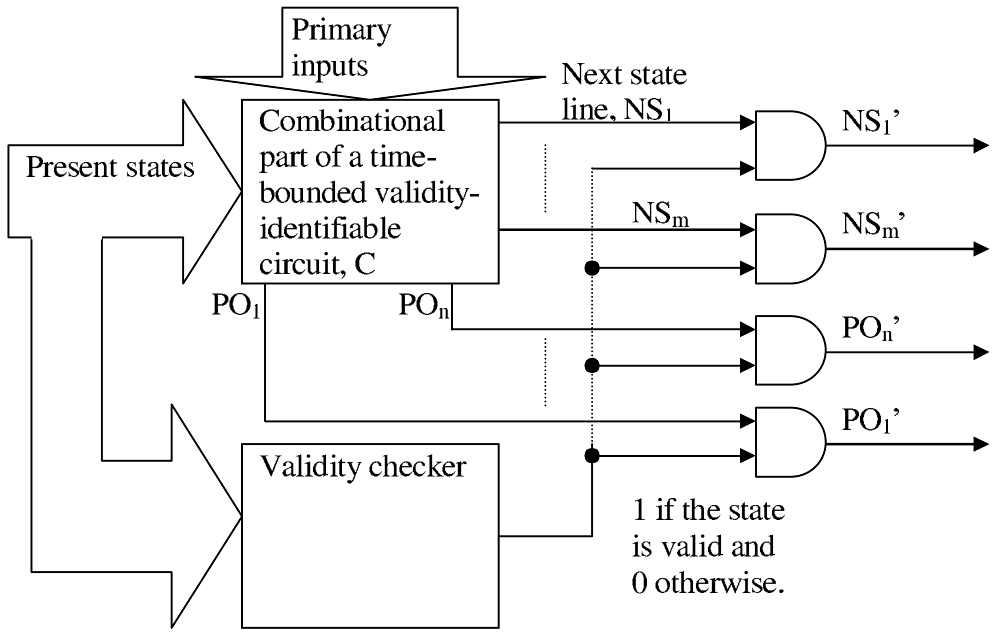

Primary inputs
Present states
Combinational part of a time-bounded validity-identifiable circuit, C
Next state line, NS1
NSm
PO1
POn
Validity checker
NS1'
NSm'
POn'
PO1'
1 if the state is valid and 0 otherwise.

Figure 4.3. Transformed combinational part C' that consists of C embedded with a validity checker.

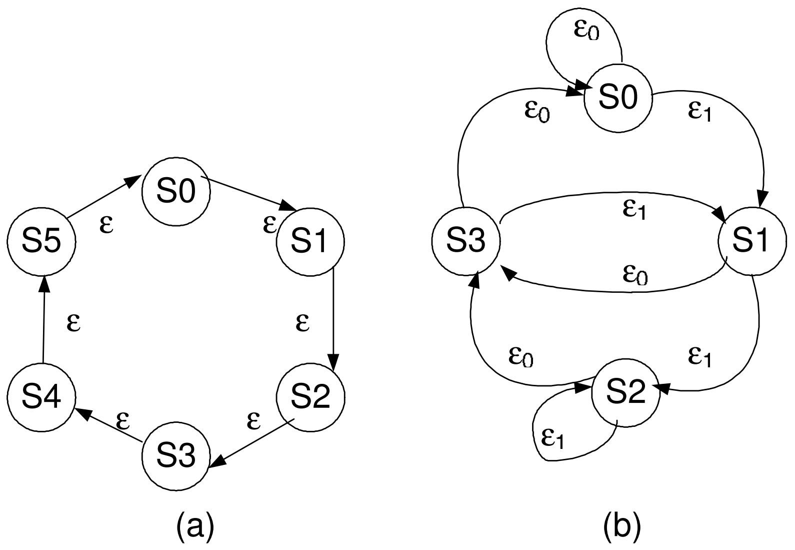


Figure 4.4. (a) State diagrams of counter-cycle FSM. (b) State-shiftable FSM.

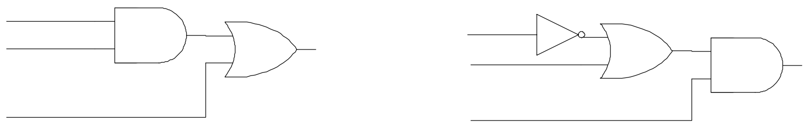
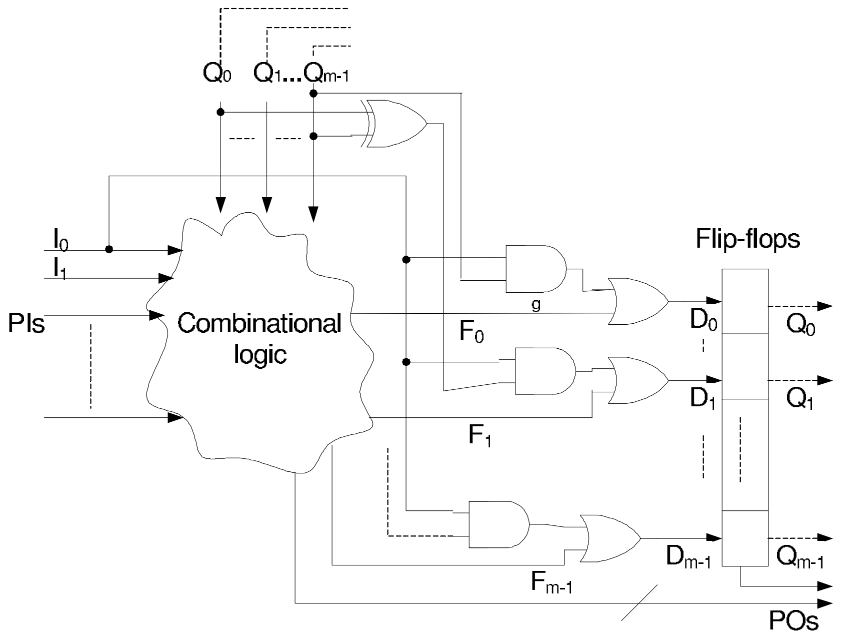
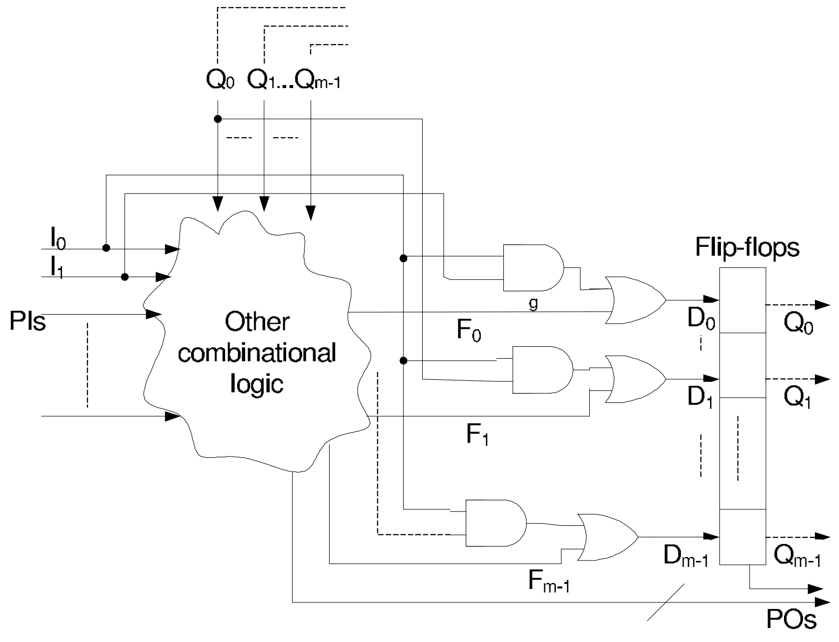


Figure 4.5. (a) AND-OR logic. (b) NOT-OR-AND logic.



(a)



(b)

Figure 4.6. (a) Realizations of counter-cycle FSM. (b) State-shiftable FSM with feedback polynomial 0...11.

Chapter 5

Classification of Sequential Circuits with Path Delay Faults Based on Combinational Test Generation Complexity

5.1. Introduction

In Chapter 3, τ^k-notation is introduced. Using the notation, test generation complexity for several classes of sequential circuits with stuck-at faults is studied. In this chapter, test generation complexity for sequential circuits with path delay faults is discussed based on τ^k-notation. In the following text, the test generation problem for path delay faults is simply called PDF test generation while the test generation problem for stuck-at faults is simply called SAF test generation.

Delay testing is very important to ensure the temporal correctness of a circuit. According to Cheng's classification, path delay faults (PDFs) can be classified into four categories by the conditions of their off-inputs: (1) *robust testable*, (2) *non-robust testable*, (3) *functional sensitizable* and (4) *functional unsensitizable*[20]. PDF test generation problem has been studied and several algorithms have been introduced. Some approaches increased the complexity of the problem. Therefore, works have been done on establishing the relationship between PDF test generation problem and SAF test generation problem. Saldanha et al.[13] proved that a single stuck-at fault (SAF) test generation tool could be used for the robust test generation in a combinational circuit by transforming it into a rising-smooth-circuit or falling-smooth-circuit. Gharaybeh et al.[14] showed that the single SAF test generation tool can be used for the non-robust test generation for a combinational circuit by first transforming a given circuit into a two-level circuit. Ohtake et al.[15] introduced a method of PDF test generation using SAF test generation algorithms. They introduced path-leaf transformation as a circuit pseudo-transformation to generate a partial leaf-dag from a given combinational circuit. Ohtake et al. in [16] also showed the reducibility between the PDF test generation of balanced sequential circuits and that for the corresponding segment path delay faults (SDF test generation) of its combinational equivalent circuit for non-robust tests. Majumder et al. [17] presented a complete characterization of the path delay faults through stuck-at faults under various classifications schemes. They considered only the combinational circuits. In Chapter 4, the faults being studied are stuck-at faults, which are logical faults. In this chapter, the work is extended to the delay faults, which are timing faults. The relationship between PDF test generation and SAF test generation for several classes of circuits is discussed based on τ^k-notation. The concern is to compare the test generation complexity between SAF and PDF. They are equivalent for combinational circuits, balanced sequential cricuits, and internally balanced sequential circuits. For acyclic sequential circuits, they are equivalent under time expansion model (TEM) with slow-fast-slow clock, but still it is not known whether they are equivalent. Since it looks equivalent, it is concluded as a conjecture. For cyclic sequential circuits, they are equivalent for t-length-bounded testable circuits with $t = \tau(n)$ and two-column SSFSM realizations with observable shifting logic under slow-fast-slow clock but it is not known if they are equivalent for two-column distributive SSFSM realizations. Then, the concern is whether there is any class of circuits, the test generation complexity for both SAF and PDF of which are not equivalent. If there exists such class of circuits, the following question arises. “Which complexity is higher, SAF’s or PDF’s?” This chapter shows that there is a class of circuits, the SAF and PDF test generation complexity of which might be not equivalent; by showing its

PDF test generation is τ-equivalent while its SAF test generation is τ^2-bounded. The SAF test generation might be not τ-equivalent, hence it is concluded in a conjecture such that they are not equivalent and PDF test generation has less complexity than SAF test generation.

In this chapter, the equivalence of the robust and non-robust SDF test generation and the SAF test generation of combinational circuits is addressed. By using this result, the reducibility of the robust and non-robust PDF test generation of acyclic sequential circuits to the SAF test generation of combinational circuits is proved. Two easily testable classes of cyclic sequential circuits, namely two-column SSFSM realizations and two-column distributive SSFSM realizations are also introduced and their test generation complexities are discussed[27].

5.2. Path Delay Fault Test Generation Complexity of the Combinational Circuits

In this section, we define a single-path leaf-dag C_P^{LD} for path P and a path rising-smooth circuit C_P^{RS} for path P, which are the pseudo circuits transformed from a given combinational circuit prior to test generation, based on [13]. These transformations allow the PDF test generation of a given combinational circuit to be done by running ATPG on the transformed pseudo circuits with the corresponding SAFs. First, the transformation of a given circuit into C_P^{LD} for path P (resp. C_P^{RS} for path P) through single-path-leaf transformation (resp. path rising-smooth transformation) to generate tests for combinational circuits with non-robust PDFs (resp. robust PDFs) is illustrated.

Definition 5.2.1. A ***single-path leaf-dag*** C_P^{LD} ***for path*** P is a combinational circuit such that a fanout and an inverter along P are only permitted at the starting point of P and the output of the inverter, if one exists along P, is not allowed to have fanouts.

Definition 5.2.2. Let P denote a path in a given combinational circuit C. C can be transformed into a single-path leaf-dag C_P^{LD} for path P, by ***the single-path-leaf-transformation***:

- P consists of an ordered set of gates $\{g_1, g_2, ..., g_m\}$, where g_1 is a primary input and g_m is a primary output. Also, gate g_j is an input to gate g_{j+1} $(1 \leq j \leq m-1)$. Let $Pre(g_j)$ denote a set of predecessor gates $\{g_1, g_2, ..., g_{j-1}\}$ on P. Traversing from g_m, if a gate g_j has a fanout of two or more, each gate in $Pre(g_j)$ with the connections to its immediate predecessor gates are duplicated once. Let g'_k denote the duplicate of g_k, where $g_k \in Pre(g_j)$. For each g_k in $Pre(g_j)$ and for each immediate successor gate h_{k+1} of g_k which is not on P, the connection of g_k to h_{k+1} is changed to the connection of g'_k to h_{k+1}. The resulting path P is free of fanout.

- Starting from g_m along P, all the NAND (resp. NOR) gates on P are changed to the OR (resp. AND) gates using De Morgan's Law.

Let n denote the number of gates of a given combinational circuit. Let n_p and n'_p denote the number of gates along P and the number of gates along P that are duplicated. Note that the size of the resulting circuit after this transformations is $n' = n + n'_p$. Since $n'_p \leq n_p \leq n$, the size of the transformed circuit is at most $2n$.

Definition 5.2.3. The I-edge of path P with input i in a single-path leaf-dag C_P^{LD} refers to the first connection of P after the inverter, if it exists. The I-edge is said to be associated with input i.

Let i denote a primary input on path P, the I-edge of P and other fanout branches of i have a transition if i has a transition. The transition from a fanout branch of i may propagate to the side-input of a gate on P. Using I-edge as one of the properties, the pseudo-circuit called path rising-smooth circuit C_P^{RS} (resp. path falling-smooth circuit C_P^{FS}) is introduced. Except I-edge, a constraint is assigned to each fanout branch so that the two-pattern tests does not propagate a transition to any side input of P for a robust PDF.

Definition 5.2.4. A single-path leaf-dag C_P^{LD} for path P can be transformed into a path rising-smooth circuit C_P^{RS} (resp. path falling-smooth circuit C_P^{FS}) for path P by the path rising-smooth (resp. path falling-smooth) transformation:

- Let Q_{OR} (resp. Q_{AND}) denote the OR gates (resp. AND gates) along P that have a rising (resp. falling) transition along. A gate may have no parity, 0, 1 or both parities. A gate fed to the side-input of an OR gate (resp. AND gate) in Q_{OR} (resp. Q_{AND}) has parity 1 (resp. 0). Perform a reverse topological traversal of the gates Q in the transitive fanout of i, to determine the parity of all gates along the side-paths to P where i is the primary input on P. The parity is complemented across a NOT gate. If some fanouts of a gate have parity 1 and others have parity 0, the gate is assigned both parities.

- Duplicate the gates so that each resulting gate is either nothing, 0, or 1, but not both, depending on its successor gates.
  - Traversing from the primary output g_m on P, for each gate h_j with a parity (parities) and with a successor gate that is off path and without parity, h_j and the connections to its immediate predecessor gates are duplicated once and its duplicate h'_j has no parity. For each immediate successor gate h_{j+1} of h_j that is off path and has no parity, the connection from h_j to h_{j+1} is replaced by the connection from h'_j to h_{j+1}.
  - Traversing from the primary output g_m on P, each gate h_j with both parities and the connections to its immediate predecessor gates are duplicated once and assigned parity 1. Its duplicate h'_j is assigned parity 0. For each immediate successor gate h_{j+1} of h_j that has parity 0 (1 if there is an inversion between h_j and h_{j+1}), the connection from h_j to h_{j+1} is replaced by the connection from h'_j to h_{j+1}.
- Let input i denote the primary input on P. Assign 0 to any fanout branch of input i (or the first connection after the inverter, if it exists on the fanout branch) that is connected to a gate with parity 0 and 1 to any fanout branch of input i (or the first connection after the inverter, if it exists on the fanout branch) that is connected to a gate with parity 1.

Let n denote the number of gates of a given combinational circuit. Note that the size of the resulting circuit after the single-path-leaf transformation is $n' = n + n'_p$ where $n'_p \leq n_p \leq n$. During the path rising-smooth transformation on this circuit with size n', the gates that are potential to be duplicated are gates other than those on P and their duplicates, the number of which is $n' - n_p - n'_p$. Therefore, the size of the resulting circuit is $n'' = 2n' - n_p - n'_p = 2n + n'_p - n_p$ for $n' = n + n'_p$. The size of the resulting circuit is at most $2n$ because the maximum value of $n'_p - n_p$ is 0 where $n'_p \leq n_p$.

A parity at an input of an OR gate in Q_{OR} is corresponding to a signal that masks a transition from another input of the OR gate to the outupt. The parity at the fanout branch of i, which is not its I-edge, is the constraint that makes sure the generated tests do not propagate a transition from the fanout branch to the side input of a gate on path P.

Example 5.2.1. Figure 5.1 shows a combinational circuit(a), its full leaf-dag and rising smooth circuit, and its single-path leaf-dag and path rising-smooth circuit for $c2367x$. Gate 1 is duplicated once while gate 2 is duplicated twice in the transformation to a rising-smooth circuit (Figure 5.1(b)). Whereas, only gate 2 is duplicated twice in transforming the circuit into a path rising-smooth circuit shown in Figure 5.1(c).

Definition 5.2.5. [15] A vector pair $< \tilde{v}, v >$ is a ***single-input-change (SIC) two-pattern test*** if there exists a coordinate i such that $\tilde{v}_i = \overline{v}_i$ for the coordinate i of v and $\tilde{v}_j = v_j$ for each coordinate j other than i.

In the following text, $P \uparrow$ (resp. $P \downarrow$) denotes a rising (resp. falling) transition path delay fault where the rising (resp. falling) transition is the transition type at the starting point of path P.

Lemma 5.2.6. Let C denote a given combinational circuit with size n and $P \uparrow$ (resp. $P \downarrow$) denote a rising (resp. falling) PDF. The time complexity of the single-path-leaf transformation on C with $P \uparrow$ (resp. $P \downarrow$) is $O(n^2)$.

Proof. Let the `SuccessorGate of Gate` denote an immediate successor gate of `Gate`. Let the `PredecessorGates` denote all immediate predecessor gates of `Gate`. The following shows the pseudocode for the single-path-leaf transformation.

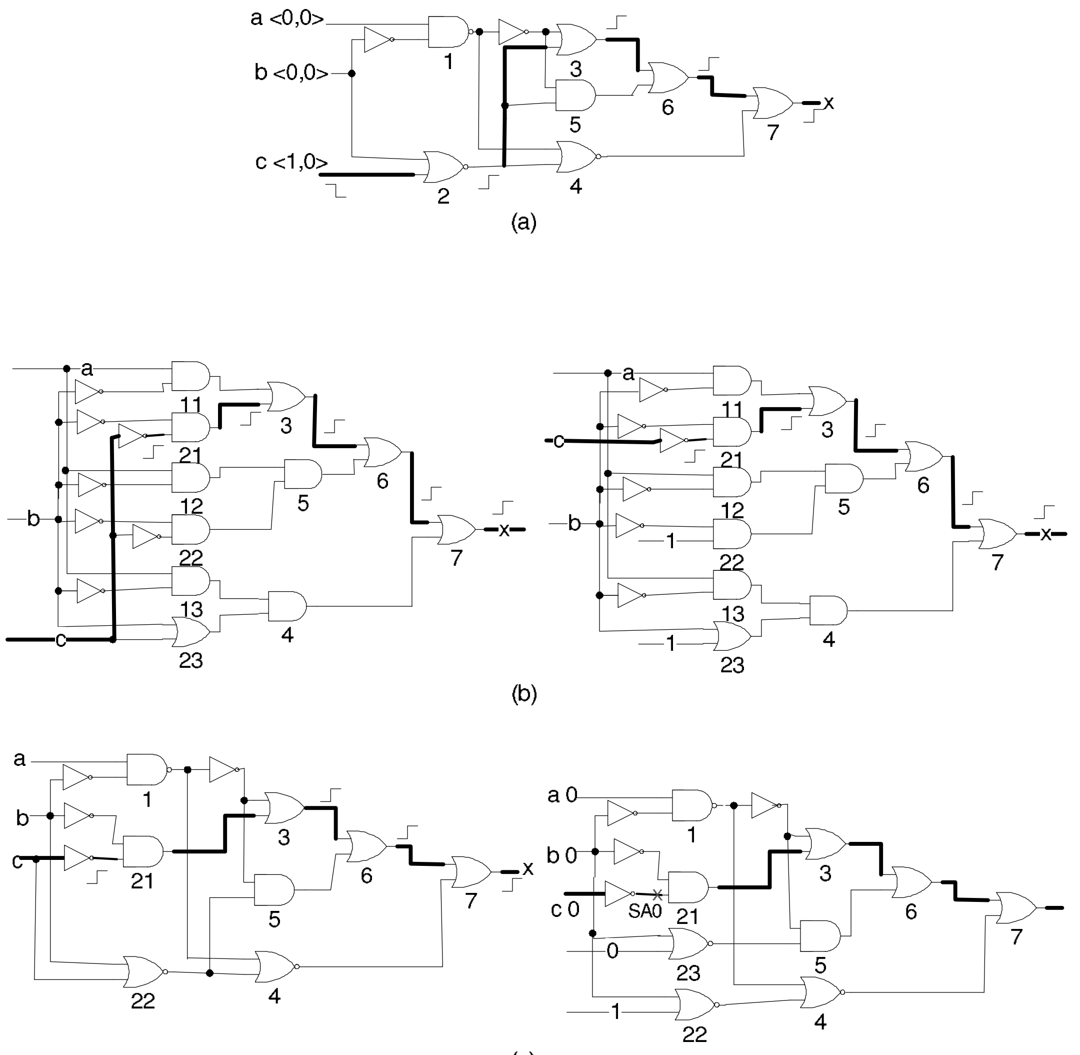


Figure 5.1. (a) A PDF $c2367x \uparrow$. (b) A full leaf-dag (left) and a rising-smooth circuit (right) (c) A single-path leaf-dag (left) and a path rising-smooth circuit (right).

```
 Single-path-leaf transformation (C, P)                times
                                                      (at most)
  FOR each Gate on P from the primary output to the      n+1
  primary input
    IF Gate has a fanout of more than one
      SET FirstFanout as Gate
      BREAK LOOP FOR
    END IF
  END FOR

  FOR each Gate on P from FirstFanout to the primary      n
  input
    duplicate Gate as Gate'
    connect Gate' to PredecessorGates
    FOR each SuccessorGate of Gate                       n*n
      IF SuccessorGate is not on P THEN
        Replace connection Gate to SuccessorGate by
        connection Gate' to SuccessorGate
      END IF
    END FOR
  END FOR

  FOR each Gate on P from the primary output to the      n+1
  primary input
    Change the NAND gates and NOR gates to the
    AND gates and OR gates, respectively
  END FOR
```

The pseudocode proves the lemma. The proof for $P\downarrow$ can be derived similarly. □

Lemma 5.2.7. Let C denote a single-path leaf-dag with size n and $P\uparrow$ (resp. $P\downarrow$) denote a rising (resp. falling) PDF. The time complexity of the path rising-smooth (resp. path falling-smooth) transformation on C with $P\uparrow$ (resp. $P\downarrow$) is $O(n^2)$.

Proof. The proof is only for $P\uparrow$. The similar proof can be derived for $P\downarrow$ by considering AND gates instead of OR gates and all the parity in the pseudocode below is complemented. Let g denote the number of the OR gates on P while h denote the total number of gates on P and their duplicates where g, $h \leq n$. Let `SuccessorGate of Gate` and `PredecessorGates` denote an immediate successor gate and all immediate predecessor gates, respectively of `Gate` and i denote the primary input on P. The following pseudocode proves the lemma.

```
Path rising-smooth transformation (C, P)                        times
                                                              (at most)
 FOR the Gate at the side-input of each ORGate on P               g+1
   IF there is an inversion between Gate and ORGate THEN
     Assign to Gate a parity 0
   ELSE
     Assign to Gate a parity 1
   END IF
 END FOR

 FOR each Gate from the primary output on P in the              n+1
 transitive fanout of the primary input i on P
   FOR each SuccessorGate of Gate                               n*n
     IF there is an inversion between Gate and
     SuccessorGate THEN
       Assign to Gate a parity complemented to the
       SuccessorGate's
     ELSE
       Assign to Gate a parity same as the SuccessorGate's
     END IF
   END FOR
 END FOR
```

```
FOR each Gate with a parity                                   n-h+1
  IF there is an off-path SuccessorGate without parity THEN
    Duplicate Gate as Gate’
    Connect Gate’ to PredecessorGates
    FOR each off-path SuccessorGate of Gate                 (n-h)*(n-h)
      IF SuccessorGate has no parity
        Replace connection from Gate to SuccessorGate
        by connection from Gate’ to SuccessorGate
      END IF
    END FOR
  END IF
END FOR

FOR each Gate with both parities                           n-h+1
  Duplicate Gate as Gate’
  Connect Gate’ to PredecessorGates
  Assign to Gate a parity 1
  Assign to Gate’ a parity 0
  FOR each SuccessorGate                                 (n-h)*(n-h)
    IF the parity is 0 (1 if there is an inversion
    between Gate and SuccessorGate) THEN
      Replace the connection from Gate to SuccessorGate
      by a connection from Gate’ to SuccessorGate
    END IF
  END FOR
END FOR

FOR each Gate connected to i                                2n
  Assign to Gate 1 if the parity is 1
  Assign to Gate 0 if the parity is 0
END FOR
```

□

Lemma 5.2.8. The time complexity of SIC two-pattern n-bit test transformation is $O(n)$.

Proof. Let v denote a test obtained from a SAF test generation. The first pattern $\tilde{v}$ can be derived by complementing the value v_i for $\tilde{v}_i$ and duplicate other v_j for $\tilde{v}_j$, where $i \neq j$ based on the definition of SIC, which can be done in linear time. □

Definition 5.2.9. A given combinational circuit C with a SAF f can be transformed into a circuit C_f^δ (Figure 5.2) by the δ ***transformation*** as follows:

S1. Let $o_1, ..., o_p$ denote the primary outputs of C. Let $c_1, ..., c_p$ denote the XOR function of each primary output of C and the corresponding primary output of the faulty circuit C_f. Let $G(C, C_f)$ be the circuit realizing c_1 OR ...OR c_p.

S2. Connect the output of $G(C, C_f)$ to a two-input AND gate A. The other input of the AND gate A is a primary input I while the output of the AND gate A is a primary output O in C_f^δ.

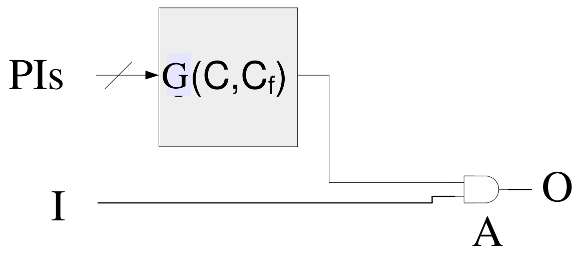


Figure 5.2. A circuit C_f^δ.

Example 5.2.2. Figure 5.2 shows the general block diagram for a circuit C_f^δ. Note that the transitive fanin of c can be any combinational logic. It is well-known that the SAF test generation problem of circuit C is polynomially transformable into CIRCUIT-SAT(C') problem where C' is an XOR function of circuit C and its faulty circuit C_f.

Lemma 5.2.10. Let $P = \{I, A, O\}$. Let C and C_f^δ denote a combinational circuit and its δ-transformed circuit, respectively. v is a test for a SAF f in C if and only if $< v \sim 0, v \sim 1 >$ (resp. $< v \sim 1, v \sim 0 >$) is a robust test for path $IAO \uparrow$ (resp. $IAO \downarrow$) in the corresponding circuit C_f^δ where $v \sim b$ denotes the concatenation of vector v and bit b that is a value at the primary input of P.

Proof. **If part:** $< v \sim 0, v \sim 1 >$ is a robust test for the $IAO \uparrow$ in a C_f^δ. The partial two-pattern vector $< 0, 1 >$ launches the $IAO \uparrow$ while the partial two-pattern vector $< v, v >$ generates the stable non-controlling values at the side-inputs of gates on IAO. This means c has a stable value 1 under $< v \sim 0, v \sim 1 >$. Therefore, v is a test for SAF f in C.
Only if part: v is a test for SAF f in C. v generates 1 at c as c is a boolean circuit casting the CIRCUIT-SAT problem that is equivalent to the test generation problem of SAF f. By applying $< v, v >$ to the partial circuit c, stable non-controlling values are generated at the side-inputs of all gates on IAO. Therefore, $< v \sim 0, v \sim 1 >$ is also a robust test for the $IAO \uparrow$. $\square$

Lemma 5.2.11. The robust PDF test generation of path IAO of C_f^δ is equivalent to the SAF test generation of combinational circuits, which is τ-equivalent.

Proof. Lemma 5.2.10 proves the equivalence of both problems while the previous chapter showed that the SAF test generation of combinational circuits is τ-equivalent. $\square$

Lemmas 5.2.10 and 5.2.11 will be used as part of the proof of the PDF test generation complexity and SDF test generation complexity in this section and the following section.

section and the following section.

5.2.1 Robust Testable PDFs

Lemma 5.2.12. [13] $< v_1, v_2 >$ is a robust test for the $P \uparrow$ (resp. $P \downarrow$) in the path rising-smooth-circuit C_P^{RS} for P, if and only if $< v_1, v_2 >$ is a robust test for the $P \uparrow$ (resp. $P \downarrow$) in the single-path leaf-dag C_P^{LD} for P.

Proof. The following is the proof for the case of $P \uparrow$. The proof for the case of $P \downarrow$ can be done analogously by considering AND gates instead of OR gates. Let

i be the input of P. Let P_I denote the side-paths to OR gates along P in C_P^{LD}, such that the I-edge of each $Q \in P_I$ is associated with input i.

If part: $< v_1, v_2 >$ is a robust test for the $P \uparrow$ in C_P^{LD}. By the definition of robust test, each side-input along path $Q \in P_I$ to an OR gate along P in C_P^{LD} is at a steady non-controlling (0) value with no transitions although there is a transition on the I-edge of each $Q \in P_I$ associated with i on the application of v_2 after v_1. This implies that asserting any constant value on the I-edge of each $Q \in P_I$ in C_P^{RS} leaves the value on these side-inputs unchanged under $< v_1, v_2 >$. Hence $< v_1, v_2 >$ remains a test for P in C_P^{RS}.

Only if part: Only the I-edges of paths in P_I are set to 1 or 0 according to their parities in obtaining C_P^{RS} from C_P^{LD}. By the definition of robust test, each side-input to OR gates along P in C_P^{RS} is at a steady non-controlling(0) value. This implies that the constants 1 or 0 asserted on the I-edge of each path in P_I according to their parities does not propagate to the side-inputs of these OR gates. Otherwise, the side-inputs to the OR gates are not guaranteed to be at non-controlling value under v_2. In other words, none of the transitions on the corresponding I-edges in C_P^{LD} propagate to the side-inputs of OR gates along P. Since the side-inputs of all other gates along P are same in C_P^{LD} and C_P^{RS}, $< v_1, v_2 >$ is a robust test for P in C_P^{LD}. $\square$

Lemma 5.2.13. [13] v is a test for the SA0(resp. SA1) fault on the I-edge of P in the C_P^{RS} for path P if and only if the SIC vector $< \tilde{v}, v >$ is a robust test for the $P \uparrow (P \downarrow)$ in C_P^{LD}.

Proof. **If part:** Let $< \tilde{v}, v >$ be a robust test for the $P \uparrow$ in C_P^{LD}. Then it is also a robust test for the $P \uparrow$ in C_P^{RS} according to Lemma 5.2.12. The output P is 1 when v is applied to the C_P^{RS} for P. Consider what happens in the presence of the SA0 fault on the I-edge of P in C_P^{RS} under vector v. By the definition of robust test, each side-input to the OR gates on P is at a noncontrolling value under v. The I-edge of P is 1 (in the absence of the fault) under vector v in C_P^{RS}. Hence v is a test for the SA0 fault on the I-edge of P in C_P^{RS}.

Only if part: Let v be a test vector for the SA0 fault on the I-edge of P in the C_P^{RS} for P. i is the only input in C_P^{RS} that changes under the vector pair $< \tilde{v}, v >$. Let P_S denote the side-paths to P whose I-edges are not associated with i. Each side-input to P that belongs to a side path in P_S must be at a non-controlling value under v; this is due to the test condition for the SAF. Among the remaining side-inputs, let I_{and} denote the side inputs that meet P at an AND gate, and let I_{or} denote the side-inputs that meet P at an OR gate. I_{or} is empty by construction of the C_P^{RS} because the side-paths to OR gates on P that have i as inputs have been separated to form new inputs during the transformation. Since a SA0 fault is being tested, each side input to P in I_{and} is at value 1 in C_P^{RS} under v (in the absence of the fault). Apply the vector $\tilde{v}$ to the circuit. The side-inputs to P in I_{and} may be at either value 0 or 1. Then, apply the vector v. The side-input to P in I_{and} may change to 1, or they may remain at 1, possibly with dynamic or static hazards, respectively. All the other side-inputs to P remain at hazard-free non-controlling values, since the transition on i under the vector pair cannot propagate to these side-inputs in the C_P^{RS}. The transition that propagates along P is a 0-to-1 transition, which propagates a delay fault at each AND gate along P irrespective of the presence of hazards on the side-inputs, provided the side-inputs have a final value of 1. Since this condition holds, $< \tilde{v}, v >$ is a robust test for the rising transition along P in C_P^{RS}, and hence in C_P^{LD}. $\square$

Lemma 5.2.14. The robust PDF test generation of combinational circuits and the SAF test generation of I-edges of path rising-smooth circuits are equivalent, which are τ-bounded.

Proof. Lemma 5.2.12 and 5.2.13 have proved the relationship between the robust PDF test generation of combinational circuits and the SAF test generation of I-edges of path rising-smooth circuits. Therefore, a two-pattern robust test on a combinational circuit C for $P \uparrow$ (resp. $P \downarrow$) can be performed using SAF test generation by the following procedure.

1. Perform single-path-leaf transformation for P on C and the resulting circuit is C_P^{LD}.

2. Perform path rising-smooth transformation (resp. path falling-smooth transformation) on C_P^{LD} for P and the resulting circuit is C_P^{RS}.

3. Perform SAF test generation of SA0 (resp. SA1) on the I-edge associated with i in C_P^{RS}. Let v denote the test.

4. Transform v into the SIC two-pattern test $< \tilde{v}, v >$.

Let n be the size of a given combinational circuit C. Let T_{SPL}, T_{PRS}, T_C^{sSA} and T_P denote the time complexity of the single-path-leaf transformation, the time complexity of the path rising-smooth transformation, the single SAF combinational test generation and the two-pattern transformation. Let N_{SPL}, N_{PRS}, N_C^{sSA} and N_P denote the problem sizes of the single-path-leaf transformation, the path-rising-smooth transformation, the single SAF combinational test generation and the two-pattern transformation, respectively. According to Lemmas 5.2.6-5.2.8 and for $N_{SPL} = n$, $n \leq N_{PRS} \leq 2n$, $n \leq N_C^{sSA} \leq 2n$ and $N_P \leq n$, the test generation complexity is

$$\begin{aligned}
T_C^{rPD}(n) &= T_{SPL}(N_{SPL}) + T_{PRS}(N_{PRS}) + O(T_C^{sSA}(N_C^{sSA})) + T_P(N_P) \\
&= O(N_{SPL}^2) + O(N_{PRS}^2) + O(\tau(N_C^{sSA})) + O(N_P) \\
&= O(n^2) + O(n^2) + O(\tau(2n)) + O(n) \\
&= O(\tau(n)).
\end{aligned}$$

□

Example 5.2.3. Given a combinational circuit (Figure 5.1(a)) that has PDF $c2367 \uparrow$, it is transformed into the corresponding single-path leaf-dag and then path rising-smooth circuit for $c2367x$. Then combinational test generation is performed to obtain the test patterns. The two-pattern test at primary inputs abc for this example is $< 001, 000 >$.

Theorem 5.2.15. *The robust PDF test generation and the SAF test generation of combinational circuits C are equivalent, which are τ-equivalent.*

Proof. To show that the robust PDF test generation and the SAF test generation of combinational circuits are equivalent, it is needed to prove that the SAF test generation of combinational circuits is equivalent to the robust PDF test generation of a class of combinational circuits in addition to Lemma 5.2.14. Thus, Lemma 5.2.11 and Lemma 5.2.14 prove the theorem. □

5.2.2 Non Robust Testable PDFs

Lemma 5.2.16. v is a test for the SA0(resp. SA1) fault on the I-edge of P in the C_P^{LD} for path P if and only if $< \tilde{v}, v >$ is a non-robust test for the $P \uparrow (P \downarrow)$

in C_P^{LD}.

Proof. **If part:** Let $< \tilde{v}, v >$ be a non-robust test for $P \uparrow$ in C_P^{LD}. The output of P is 1 when v is applied to C_P^{LD}. Consider the case of the presence of SA0 on the I-edge of P in C_P^{LD} under vector v. From the definition of non-robust test, each side-input to P is at a non-controlling value under v. The I-edge of P is 1 under v in the absence of the fault $P \uparrow$. According to test condition, the SA0 is excited and its fault effect propagates to the output in C_P^{LD}. Hence v is a test for the SA0 fault on the I-edge of P in C_P^{LD}.
Only if part: Let v be a test vector for the SA0 on the I-edge of P in C_P^{LD}. i is the input that changes under the vector pair $< \tilde{v}, v >$. Under the test condition, the side inputs along P are at non-controlling value under v. When vector $\tilde{v}$ is applied, the side-inputs to P may be at either value 0 or 1. The side-inputs to P may change to non-controlling value or remain at non-controlling value. According to the definition of non robust test, only the second vector of side-inputs must be non-controlling value. Thus, $< \tilde{v}, v >$ is a non-robust test for $P \uparrow$ in C_P^{LD}. □

Lemma 5.2.17. The non-robust PDF test generation of combinational circuits is equivalent to the SAF test generation of I-edges of single-path leaf dags, which is τ-bounded.

Proof. Based on Lemma 5.2.16, the following procedure generates a two-pattern non-robust test on a combinational circuit using a stuck-at fault test generation.

1. Perform single-path-leaf transformation on C with path P. The resulting circuit is called single path-leaf-dag C_P^{LD} for P.

2. Perform SAF test generation of SA0 (resp. SA1) on the I-edge of P associated with i in C_P^{LD}. Let v denote the test.

3. Transform v into the SIC two-pattern test $< \tilde{v}, v >$.

Let n be the size of a given combinational circuit C. Let T_{SPL}, T_C^{sSA} and T_P denote the time complexity of the single-path-leaf transformation, the single SAF combinational test generation and the two-pattern transformation. Let N_{SPL}, N_C^{sSA} and N_P denote the problem sizes of the single-path-leaf transformation, the single SAF combinational test generation and the two-pattern transformation, respectively. According to Lemmas 5.2.6 and 5.2.8 and also for $N_{SPL} = n$, $n \leq N_C^{sSA} \leq 2n$ and $N_P \leq n$. Therefore, the test generation complexity is

$$\begin{aligned} T_C^{nrPD}(n) &= T_{SPL}(N_{SPL}) + O(T_C^{sSA}(N_C^{sSA})) + T_P(N_P) \\ &= O(N_{SPL}^2) + O(\tau(N_C^{sSA})) + O(N_P) \\ &= O(n^2) + O(\tau(2n)) + O(n) \\ &= O(\tau(n)). \end{aligned}$$

□

Theorem 5.2.18. *The non-robust PDF test generation and the SAF test generation of combinational circuits C are equivalent, which are τ-equivalent.*

Proof. To show that the non-robust PDF test generation and the SAF test generation of combinational circuits are equivalent, it is sufficient to prove that the SAF test generation of combinational circuits is equivalent to the non-robust PDF test generation of a subclass of single-path leaf-dags. Since C_f^δ is a subclass of single-path leaf-dags and robust PDF is a subset of non-robust PDF, Lemma 5.2.11 in addition to Lemma 5.2.17 prove the theorem. □

The combinational test generation complexity for robust and non-robust PDFs is τ-equivalent.

5.3. Segment Delay Fault Test Generation Complexity of the Combinational Circuits

In Section 5.2, it is showed that the combinational test generation complexity for robust and non-robust PDFs is τ-equivalent. In this section, segment delay faults (SDFs) are considered and the test generation complexity of those faults in combinational circuits is considered. The result will be used in Section 5.4 to study the PDF test generation complexity in acyclic sequential circuits.

Definition 5.3.1. An n-bit vector v_l consists of $(v_1^l, v_2^l, ..., v_j^l, ..., v_n^l)$.

Definition 5.3.2. A ***segment-leaf-dag*** C_S^{LD} ***for segment*** S ***(***$S = \{g_1, g_2, ..., g_m\}$***)*** is a combinational circuit such that a fanout and an inverter are only permitted at g_1 of the segment S and the output of an inverter along S is not allowed to have a fanout.

Definition 5.3.3. Let S denote a segment in a given combinational circuit C. C can be transformed into C_S^{LD}, by the ***segment-leaf transformation***:

- S consists of an ordered set of gates $\{g_1, g_2, ..., g_m\}$, where the output of g_1 is the starting point of S and the output of g_m is the ending point of S, respectively. Also, gate g_i is an input to gate g_{i+1} ($1 \leq i \leq m-1$). Let $Pre(g_i)$ denote a set of gates $\{g_2, g_3, ..., g_{i-1}\}$ on S. Traversing from g_m, if a gate g_i has a fanout of two or more, each gate in $Pre(g_i)$ with the connections to its immediate predecessor gates are duplicated once. Let g'_j denote the duplicate of g_j, where $g_j \in Pre(g_i)$. For each g_j in $Pre(g_i)$ and for each immediate successor gate h_{j+1} of g_j, the connection from g_j to h_{j+1} is changed to the connection from g'_j to h_{j+1} if h_{j+1} is not on S. The resulting segment S is free of fanout.

- Starting from g_m along S, all the NAND (resp. NOR) gates on S are changed to the OR (resp. AND) gates using De Morgan's Law.

Note that the segment-leaf-transformation is defined analogously to the single-path-leaf transformation by considering segment S instead of path P.

Definition 5.3.4. The ***S-edge of segment*** S ***with starting point*** s in a segment-leaf-dag C_S^{LD} refers to the first connection of S after the inverter, if it exists. The S-edge is said to be associated with s.

Example 5.3.1. Figure 5.3 shows a combinational circuit (a) and its segment leaf-dag for segment *s45e* (b). Gate *4* is duplicated so that S is free of fanout while Gate *5* and Gate *41* are changed to OR gate and AND gate respectively using De Morgan's law so that the inverters exist only at s along S.

Definition 5.3.5. A ***segment transition-smoother STS***(Figure 5.4) is a circuit with inputs V_0 and V_1 and outputs D_0 and D_1. When $V_0 = V_1$, D_0 and

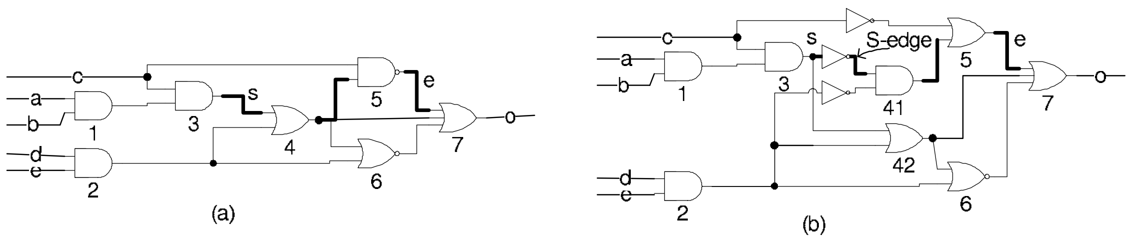


Figure 5.3. (a) a combinational circuit (b) a segment leaf-dag

D_1 have the same value as V_0 and V_1, respectively. When $V_0 \neq V_1$, D_0 is assigned value 0 while D_1 is assigned value 1. The logic function is summarized as follows.

V_0	V_1	D_0	D_1
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

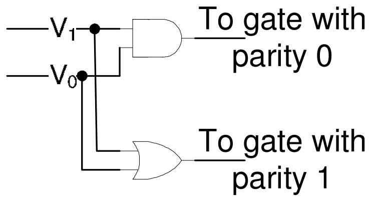


Figure 5.4. segment transition-smoother

Definition 5.3.6. A segment-leaf-dag C_S^{LD} for segment S can be transformed into a segment-rising-smooth circuit C_S^{RS} (resp. segment-falling-smooth circuit C_S^{FS}) for segment S by the segment-rising-smooth (resp. segment falling-smooth) transformation:

1. Let Q_{OR} (resp. Q_{AND}) denote the OR gates (resp. AND gates) along S that have a rising (resp. falling) transition. A gate may have no parity, 0, 1 or both parities. A gate fed to the side-input of an OR gate (resp. AND gate) in Q_{OR} (resp. Q_{AND}) has parity 1 (resp. 0). Perform a reverse topological traversal of the transitive fanout of the primary inputs that are

in the transitive fanin of g_1 on $S(=\{g_1, g_2, ..., g_m\})$ to determine the parity of gates along the side-paths to S. The parity is complemented across a NOT gate. If some fanouts of a gate have parity 1 and others have parity 0, the gate is assigned both parities.

2. Duplicate the gates so that each resulting gate is either nothing, 0, or 1, but not both, depending on its successor gates.

   - Traversing from the ending point g_m on S, for each gate h_j that has a parity (parities) and has a successor gate that is off path and without parity, h_j and the connections to its immediate predecessor gates are duplicated once and its duplicate h'_j has no parity. For each immediate successor gate h_{j+1} of h_j that is off path and has no parity, the connection from h_j to h_{j+1} is changed to the connection from h'_j to h_{j+1}.
   - Traversing from the ending point g_m on S, each gate h_j with both parities and the connections to its immediate predecessor gates are duplicated once and assigned parity 1 while its duplicate h'_j is assigned parity 0. For each immediate successor gate h_{j+1} of h_j that has parity 0 (1 if there is an inversion between h_j and h_{j+1}), the connection from h_j to h_{j+1} is changed to the connection from h'_j to h_{j+1}.

3. Let C_{2P} denote the transformed circuit after the above two steps. C_{2P} is called the second pattern partial circuit. Duplicate the transitive fanin of S-edge of C_{2P} as C_{1P}, which is called the first pattern partial circuit. Each primary input i_{1P} in C_{1P} has a corresponding primary input i_{2P} in C_{2P}. Each gate g_{1P} in C_{1P} has a corresponding g_{2P} in C_{2P}. Let s_{2P} denote the starting point of S_{2P}. Insert a segment transition-smoother STS to a primary input i_{2P} if i_{2P} has an immediate gate with parity 0 or 1 by connecting the inputs i_{1P} of C_{1P} and i_{2P} of C_{2P} to the inputs V_0 and V_1 of STS, respectively and connecting D_0 and D_1 of STS to the immediate gates with parity 0 and 1, respectively.

Let n denote the number of gates of a given combinational circuit and n_s denote the number of gates along S. Let n'_s denote the number of gates along

S that are duplicated where $n'_s \leq n_s$. Note that the size of the resulting circuit after the segment-leaf transformation is $n''' = n + n'_s$ where $n'_s \leq n_s \leq n$. Let n_{sts} denote the number gates resulted from the insertion of STS. During the segment-rising-smooth transformation on the segment leaf-dag with size n''', the gates that are potential to be duplicated in step 2 are gates other than those on S and their duplicates, the number of which is at most $n''' - n_s - n'_s$. Therefore, the size of the resulting circuit after step 2 is at most $2n''' - n_s - n'_s$ or $2n + n'_s - n_s$ for $n''' = n + n'_s$. The number of gates duplicated in step 3 is the number of gates composing the transitive fanin of the starting point of S, which is $n_c \leq 2n - 2n_s$. The number of gates n_{sts} is at most $2n - 2n_s$ as each STS is composed by 2 gates. Therefore, the size of the resulting circuit is $n'''' \leq 2n + n'_s - n_s + n_c + n_{sts} = 6n + n'_s - 5n_s$. The size of the resulting circuit is at most $6n - 5$.

Lemma 5.3.7. Let n denote the size of a given combinational circuit C. The time complexity of the segment-leaf transformation on C is $O(n^2)$.

Proof. Segment-leaf transformation is composed of two steps, which are duplication of each gate along segment S at most once and moving all the inverters on S to the starting point s. The pseudocode is similar to the pseudocode of the single-path-leaf transformation in Lemma 5.2.6 except that segment S from the starting point s to the ending point e is considered instead of path P from the primary input to the primary output. The pseudocode proves the lemma. $\square$

Lemma 5.3.8. Let C denote a segment-leaf-dag and $S \uparrow$ denote a rising SDF in C. The time complexity of the segment rising-smooth transformation on C is $O(n^2)$.

Proof. Let g denote the number of the OR gates on S. Let h denote the total of the gates on S and their duplicates. `SuccessorGate of Gate` denotes the immediate successor gate of `Gate` while `PredecessorGates` denotes the immediate predecessor gates of `Gate`. i_{2P} means a primary input of C while i_{1P} means a primary input of the duplicate of C. $STS(i, j)$ denotes STS with inputs i and j. The following pseudocode proves the lemma.

```
Segment rising-smooth transformation (C, S)                times
                                                          (at most)
 FOR the Gate at the side-input of each ORGate on S         g+1
   IF there is an inversion between Gate and ORGate THEN
     Assign to Gate a parity 0
   ELSE
     Assign to Gate a parity 1
   END IF
 END FOR

 FOR each Gate in the transitive fanout of PIs in          n+1
 the transitive fanin of the starting point s on S
   FOR each SuccessorGate of Gate                          n*n
     IF there is an inversion between Gate and
     SuccessorGate
       Assign to Gate a parity complemented to the
       SuccessorGate's
     ELSE
       Assign to Gate a parity same as the SuccessorGate's
     END IF
   END FOR
 END FOR
```

```
FOR each Gate with a parity                          n-h+1
  IF there is an off-segment SuccessorGate
  without parity THEN
    Duplicate Gate as Gate'
    Connect Gate' to PredecessorGates
    FOR each off-segment SuccessorGate of Gate    (n-h)*(n-h)
      IF SuccessorGate has no parity THEN
        Replace connection Gate to SuccessorGate
        by connection Gate' to SuccessorGate
      END IF
    END FOR
  END IF
END FOR

FOR each Gate with both parities                          n-h+1
  Duplicate Gate as Gate'
  Connect Gate' to PredecessorGates
  Assign to Gate a parity 1
  Assign to Gate' a parity 0
  FOR each SuccessorGate                               (n-h)*(n-h)
    IF the parity is 0 (1 if there is an inversion
    between Gate and SuccessorGate) THEN
      Replace the connection Gate to SuccessorGate
      by Gate' to SuccessorGate
    END IF
  END FOR
END FOR
```

```
Duplicate the circuit C                          2n                 □
FOR each primary input i                        n+1
  FOR each Gate connected to i                  2n*n
    IF the parity of Gate is 1 THEN
      replace the connection i to Gate by
      connection $D_1$ of STS($i_{2P}$, $i_{1P}$) to Gate
    ELSE IF the parity of Gate is 0 THEN
      replace the connection i to Gate by
      connection $D_0$ of STS($i_{2P}$, $i_{1P}$) to Gate
    END IF
  END FOR
END FOR
```

In the following subsection, the SDF test generation problem of a combinational circuit C is showed to be reducible to the double SAF test generation problem of its transformed circuit of C. Therefore, the time complexity of the double SAF test generation is discussed beforehand.

Lemma 5.3.9. The test generation complexity for segment-rising-smooth circuits C_S^{RS} with double SAFs is equivalent to the test generation complexity for combinational circuits with single SAFs, i.e. it is τ-equivalent.

Proof. A circuit with a multiple fault can be represented by a multiple fault model containing an SAF by adding m extra gates, where m is the multiplicity of the faults [13]. In the case of the double SAF test generation, extra two gates are added to the original circuit. Since the fault in the first pattern partial circuit is always at a primary output, the fault is tested if it is excited. Thus, the objective of the double SAF test generation problem represented by its multiple fault model is to generate tests to propagate the fault effect of the corresponding SAF to a primary output in the second pattern partial circuit. Therefore, the double SAF test generation has same complexity as the single SAF test generation. □

5.3.1 Robust Testable SDFs

Lemma 5.3.10. Let $u \sim v$ denote the concatenation of vectors u and v, where u consists of a vector at the second pattern primary inputs I_{2P} and v consists of a vector at the first pattern primary inputs I_{1P} of C_S^{RS}. $< u_1, u_2 >=< v_1 \sim v_1, v_2 \sim v_1 >$ is a robust test for the $S \uparrow$ in the segment rising-smooth circuit C_S^{RS} for S, if and only if $< v_1, v_2 >$ is a robust test for the $S \uparrow$ in the segment leaf-dag C_S^{LD} for S.

Proof. **If part:** $< v_1, v_2 >$ is a robust test for the $S \uparrow$ in C_S^{LD} for S. By the definition of robust test, the side-inputs of the OR gates along S in C_S^{LD} are at a steady non-controlling (0) value with no transitions on the application of v_2 after v_1. Let P_S be a set consisting of all the side-paths to OR gates along S. This means the transitions on each I-edge of $Q \in P_S$ do not propagate under the application of $< v_1, v_2 >$. Let u_x^y denote a vector bit of vector u_y at the input x. With the segment transition-smoother in C_S^{RS}, a constant is assigned to the fanout branch of a second pattern primary input i_{2P} according to the parity of the gate connected to the fanout branch if $u_{i_{2P}}^k \neq u_{i_{1P}}^k$ under vector u_k. Let I_{1P} and I_{2P} denote all first pattern primary inputs and all second pattern primary inputs, respectively. Since $u_{I_{1P}}^2 = u_{I_{2P}}^1 = v_1$, a constant is assigned to the fanout branch of a primary input i_{2P} according to the parity of the gate connected to the fanout branch in C_S^{RS}, which has an STS, when there is a transition at i_{2P} ($u_{i_{2P}}^2 \neq u_{i_{2P}}^1$ which also means $u_{i_{2P}}^2 \neq u_{i_{1P}}^2$) under $< u_1, u_2 >$. Since the second pattern partial circuit C_{2P} of C_S^{RS} is functionally equivalent to C_S^{LD}, the second pattern primary inputs I_{2P} of C_S^{RS} that are assigned with constants under $< u_1, u_2 >$ are corresponding to the primary inputs of C_S^{LD} that have transitions under $< v_1, v_2 >$. Since all the paths from the primary inputs with constants are the paths in P_S in C_S^{RS} and the transitions on the corresponding primary inputs in C_S^{LD} do not propagate to the side-inputs of the OR gates on S, the values of the gates in the transitive fanout of the side-inputs to the OR gates on S_{2P} in C_S^{RS} are same with those on S in C_S^{LS}. Under $u_1 (= v_1 \sim v_1)$, no constants are assigned in C_S^{RS} since $u_{I_{2P}}^1 = u_{I_{1P}}^1$ and thus $S \uparrow$ is initialized. Therefore, $< u_1, u_2 >=< v_1 \sim v_1, v_2 \sim v_1 >$ is a robust test for the $S \uparrow$ in the segment-rising-smooth circuit C_S^{RS} for S.

Only if part: Since $< u_1, u_2 >=< v_1 \sim v_1, v_2 \sim v_1 >$ is a robust test for

S in C_S^{RS}, each side-input to OR gates along S is at a steady non-controlling (0) value. This implies that the constants (assigned according to the parities) asserted on the primary inputs due to $u^1_{i_{2P}} \neq u^2_{i_{2P}}$ do not propagate to the side-inputs of OR gates along S. Since $u^1_{I_{2P}} = v_1$ and $u^2_{I_{2P}} = v_2$ and the second pattern primary inputs I_{2P} in C_S^{RS} that are assigned with constants are corresponding to the primary inputs in C_S^{LD} that have transitions under $< v_1, v_2 >$, the transition on the primary inputs in C_S^{LD} do not propagate to the side-inputs of OR gates along S under $< v_1, v_2 >$. Also, the values of all the gates without parity in the transitive fanout of the side-inputs to the OR gates on S_{2P} in C_S^{RS} and those on S in C_S^{LD} are same. Therefore, $< v_1, v_2 >$ is a robust test for S in C_S^{LD}. □

Lemma 5.3.11. $u_2 = v_2 \sim v_1$ is a test for the double SAF of SA0 on the S-edge of S_{2P} and SA1 on the S-edge of S_{1P} in C_S^{RS} if and only if $< v_1, v_2 >$ is a robust test for $S \uparrow$ in C_S^{LD}.

Proof. **If part:** Let $< v_1, v_2 >$ be a robust test for the $S \uparrow$ in C_S^{LD}. Then, from Lemma 5.3.10, $< u_1, u_2 >=< v_1 \sim v_1, v_2 \sim v_1 >$ is a robust test for the $S \uparrow$ in C_S^{RS}. The starting point s and the ending point e of S_{2P} are 1 when $u_2 (= v_2 \sim v_1)$ is applied to C_S^{RS} in a fault-free case. This means SA0 is activated and propagated to e. Based on condition 4 of the definition of SDF, the difference at e is propagated to a primary output under $v_2 \sim v_1$ in C_S^{RS}. Let P_S denote a set consist of all the side-paths to OR gates along S_{2P}. Since the transitions at the I-edges of the paths in P_S do not propagate to the side inputs of OR gates on S_{2P}, the constants assigned to the corresponding primary inputs of those I-edges do not propagate to the side-inputs of OR gates on S_{2P} and also other part of the circuit. So, $v_2 \sim v_1$ also propagate the fault effect of SA0 at e to the primary output. The partial vector v_1 of $v_1 \sim v_1$ of $< u_1, u_2 >$ assigns to the S-edge of S_{2P} a value 0 in order to initialize $P \uparrow$. This implies v_1 of $v_2 \sim v_1$ assigns to the S-edge of S_{1P} a value 0. This activates and propagate the SA1 to the primary output s_{ed}. Hence, $v_2 \sim v_1$ is a test for the SA0 on the S-edge of S_{2P} and SA1 on the S-edge of S_{1P} in C_S^{RS}.
Only if part: Let $v_2 \sim v_1$ be a test for the SA0 on the S-edge of S_{2P} and SA1 on the S-edge of S_{1P} in C_S^{RS}. The partial vector v_1 of $v_2 \sim v_1$ is a test for SA1, which assigns 0 to the S-edge of S_{1P} under the fault free case while the partial vector v_2 is a test for SA0, which assigns 1 to the S-edge of S_{2P} This

implies the partial vector v_1 of $v_1 \sim v_1$ initializes and launches the $S \uparrow$ under $< u_1, u_2 >$. According to the SAF test condition, all the side-inputs of the gates along S_{2P} are at the non-controlling value and the fault effect of S_{2P} propagate to a primary output under $v_2 \sim v_1$. In order to propagate the fault effect of SA0 on the S-edge of S_{2P}, the constants asserted by the STSs to the primary inputs do not propagate to the side-inputs of the OR gates along S_{2P}. A constant is only asserted to a primary input j if $u^k_{j2P} \neq u^k_{j1P}$. Since $u^2_{j1P} = u^1_{j2P}$, transitions at the primary inputs launched by $< u_1, u_2 >$ do not propagate to the side inputs of OR gates along S. This satisfies the definition of robustly testable SDF. Therefore, $< v_1 \sim v_1, v_2 \sim v_1 >$ is a robust test for $S \uparrow$ in C^{RS}_S and according to Lemma 5.3.10, $< v_1, v_2 >$ is also a robust test for $S \uparrow$ in C^{LD}_S for S. □

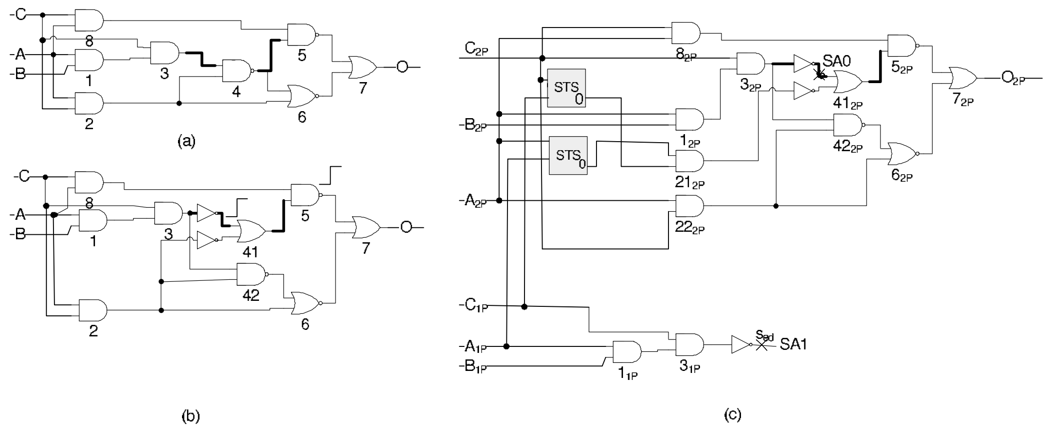


Figure 5.5. (a) a combinational circuit. (b) a segment leaf-dag. (c) a segment rising-smooth circuit.

Lemma 5.3.12. The robust SDF test generation of combinational circuits is equivalent to the SAF test generation of S-edges of segment-rising-smooth circuits, which is τ-bounded.

Proof. Lemma 5.3.10 and Lemma 5.3.11 show the equivalence of the SDF test generation of a given combinational circuit and its segment rising(falling)-smooth circuits. Therefore, SDF test generation of the combinational circuits using SAF test generation can be performed by the following procedure.

1. Perform the segment-leaf transformation for segment S on a given combinational circuit C and the resulting circuit C_S^{LD} is called segment-leaf-dag.

2. Perform the segment-rising-smooth transformation (resp. segment falling-smooth transformation) on C_S^{LD} for segment S and the resulting circuit is C_S^{RS}.

3. Perform the SAF test generation of SA0 on the S-edge of S and SA1 on the S-edge of S_d in C_S^{RS}. Let $v_1 \sim v_2$ denote the test.

4. Transform $v_1 \sim v_2$ into the robust two-pattern test $< v_1, v_2 >$.

Let n be the size of a given combinational circuit C. Let T_{SL}, T_{SRS}, T_C^{mSA} and T_P denote the time complexity of the segment-leaf transformation, segment-rising-smooth transformation, double SAF test generation and the two-pattern test transformation. Let N_{SL}, N_{SRS}, N_C^{mSA} and N_P denote the problem size of the segment-leaf transformation, segment-rising-smooth transformation, the double SAF combinational test generation and the two-pattern test transformation, respectively. According to Lemmas 5.3.7-5.3.9 and for $N_{SL} = n$, $n \leq N_{SRS} \leq 2n$, $2n \leq N_C^{mSA} \leq 6n - 5$ and $N_P \leq n$. Therefore, the test generation complexity is

$$\begin{aligned}
T_C^{rSD}(n) &= T_{SL}(N_{SL}) + T_{SRS}(N_{SRS}) + O(T_C^{mSA}(N_C^{mSA})) + T_P(N_P) \\
&= O(N_{SL}^2) + O(N_{SRS}^2) + O(\tau(N_C^{mSA})) + O(N_P) \\
&\leq O(n^2) + O(4n^2) + O(\tau(6n)) + O(n) \\
&= O(\tau(n))
\end{aligned}$$

□

Example 5.3.2. Figure 5.5 shows a combinational circuit with a SDF 345 $\uparrow$(a), its corresponding segment-leaf-dag for segment 345 (b) and its corresponding segment-rising-smooth circuit for 345. The two-pattern test at the inputs ABC of the original circuit is $< 111, 101 >$.

Theorem 5.3.13. *The robust SDF test generation and the SAF test generation of combinational circuits are equivalent, which are τ-equivalent.*

Proof. A segment-rising-smooth circuit is also a path rising-smooth circuit when the starting point and the ending point of a segment S are also the primary input and the primary output of the circuit, respectively. Therefore path rising-smooth circuits is a subclass of segment-rising-smooth circuits. To prove this theorem, it is sufficient to show that the SAF test generation of combinational circuits can be reduced to the robust PDF test generation of C_f^δ, which is a subclass of path rising-smooth circuits. Lemma 5.2.11 and Lemma 5.3.12 prove the theorem. $\square$

5.3.2 Non-robust testable SDFs

Definition 5.3.14. $C_S^{LD\prime}$ is a transformed circuit of C_S^{LD} obtained by making the S-edge of segment S a primary output.

Lemma 5.3.15. v_2 is a test for the SA0 fault on the S-edge of S in the C_S^{LD} for segment S and v_1 is a test for the SA1 fault on the S-edge of S in the $C_S^{LD\prime}$ if and only if $< v_1, v_2 >$ is a non-robust test for the $S \uparrow$ in C_S^{LD}.

Proof. **If part:** Let $< v_1, v_2 >$ be a non-robust test for $S \uparrow$ in C_S^{LD}. Consider the presence of SA0 on the S-edge of S in C_S^{LD} under vector v_2. From the definition of non-robust test and SDF, each side-input of S is at a non-controlling value under v_2 and the difference at the ending point e of S under the presence of $S \uparrow$ is propagated to a primary output. Under the fault-free case, the S-edge of S is 1 after the application of v_2. According to test condition, the SA0 fault is activated and the fault effect thus propagates to the same output as the difference at e does in the case of SDF in C_S^{LD}. Hence v_2 is a test for SA0 fault on S-edge of S in C_S^{LD}. Since v_1 assigns 0 to the S-edge of C_S^{LD} in order to initialize the $S \uparrow$, v_1 activates the SA1 fault on the S-edge of S in $C_S^{LD\prime}$ and the fault effect propagates to the primary output.

Only if part: $C_S^{LD\prime}$ and C_S^{LD} are functionally and structurally equivalent except that C_S^{LD} has an extra output at the S-edge of S. Let v_2 be a test for the SA0 on the S-edge of S in C_S^{LD} and v_1 be a test vector for the SA1 on the S-edge of S in $C_S^{LD\prime}$. According to the test condition, 1 is assigned to the S-edge of S in C_S^{LD} and all the side inputs of gates along S in C_S^{LD} are at non-controlling value. The fault effect also propagates to a primary output under v_2. This implies that the difference of the transition caused by $S \uparrow$ taking place at the ending point e of S

is observable at the primary output after applying $< v_1, v_2 >$. 0 is assigned to the S-edge of S in $C_S^{LD\prime}$ under v_1. Thus the $S\uparrow$ is initialized. Therefore, $< v_1, v_2 >$ is a non-robust test for $S\uparrow$ in C_S^{LD}. $\square$

Lemma 5.3.16. The non-robust SDF test generation of combinational circuits is equivalent to the SAF test generation of S-edge of segment leaf-dags, which is τ-bounded.

Proof. Different from robust SDF test generation, segment-rising(falling)-smooth transformation is not needed for the non-robust SDF test generation. The following procedures explain the non-robust SDF test generation of the combinational circuits C by using SAF test generation method verified by Lemma 5.3.11 that shows the equivalence of the SDF test generation of a given combinational circuit and its segment leaf-dag:

1. Perform the segment-leaf transformation for SDFs.
2. Generate $C_S^{LD\prime}$ from C_S^{LD}.
3. Generate the second test vector v_2 by the SAF test generation of SA0 (SA1) on the S-edge of S in C_S^{LD}.
4. Generate the first test vector v_1 by the SAF test generation of SA1 (SA0) on the S-edge of S in $C_S^{LD\prime}$.
5. Transform v_1 and v_2 into the non-robust two-pattern test $< v_1, v_2 >$.

Let n be the size of a given combinational circuit C. Let T_{SL}, T_C^{mSA} and T_P denote the time complexity of the segment-leaf transformation, double SAF test generation and the two-pattern test transformation. Let N_{SL}, N_C^{mSA} and N_P denote the problem size of the segment-leaf transformation, the double SAF combinational test generation and the two-pattern test transformation, respectively. According to Lemma 5.3.7 and Lemma 5.3.9, and also for $N_{SL} = n$, $n \le N_C^{mSA} \le 4n$ and

$N_P \leq n$. Therefore, the test generation complexity is

$$\begin{aligned}
T_C^{nrSD}(n) &= T_{SL}(N_{SL}) + O(T_C^{mSA}(N_C^{mSA})) + T_P(N_P) \\
&= O(N_{SL}^2) + O(\tau(N_C^{mSA})) + O(N_P) \\
&\leq O(n^2) + O(\tau(4n)) + O(n) \\
&= O(\tau(n))
\end{aligned}$$

□

Theorem 5.3.17. *The non-robust SDF test generation and the SAF test generation of combinational circuits are equivalent, which are τ-equivalent.*

Proof. In order to show that the non-robust SDF test generation and the SAF test generation of combinational circuits are equivalent, it is needed to prove the conjecture that the SAF test generation of combinational circuits is equivalent to the non-robust SDF test generation of a subclass of segment leaf-dags besides proving Lemma 5.3.16. Since segment leaf-dag is a path leaf-dag when the starting point and the ending point of S are also the primary input and the primary output of the circuit, respectively, segment leaf-dags is a superclass of path leaf-dags. Moreover, a robust testable segment delay fault is also a non-robust testable segment delay fault but the converse is not true. Therefore, Lemma 5.2.11 and Lemma 5.3.16 are sufficient to prove the theorem. □

5.4. Path Delay Fault Test Generation Complexity of the Acyclic Sequential Circuits

Based on the theoretical results in the previous section, the reducibility of the PDF test generation to the SAF test generation of acyclic sequential circuits is addressed in this section.

5.4.1 Path Delay Fault Test Generation Complexity of the Balanced Sequential Circuits

Definition 5.4.1. [16] Let G be a circuit graph of a balanced sequential circuit $\mathcal{S}^{\mathcal{B}}$. The following discussion considers only one output cone circuit $\mathcal{S}^{\mathcal{B}}_{\mathcal{C}}$ of $\mathcal{S}^{\mathcal{B}}$, which is a strongly balanced structure. Let $t(q_i)$ be an integer value which can be assigned to a vertex q_i such that it satisfies the condition of the strongly balanced structure.

$$t(v_i) = t(v_j) + w(a) \ \forall a(v_i, v_j). \tag{5.1}$$

Let t_{max} and t_{min} be the maximum value and the minimum value among the integer values assigned to the vertices of G, respectively. Let I_j $(j = 1, 2, ..., p)$ be a vertex, which corresponds to an input of the output cone circuit $\mathcal{S}^{\mathcal{B}}_{\mathcal{C}}$. A vector $T_I = (\alpha_1, \alpha_2, ..., \alpha_p)$ such that $\alpha_j = t_{max} - t(I_j) + 1$ is said to be an input timing vector of the output cone circuit $\mathcal{S}^{\mathcal{B}}_{\mathcal{C}}$. Let L be $t_{max} - t_{min} + 2$. Let $< v_1, v_2 >$ be a p-bit vector pair where a vector v_l is $(v_1^l, v_2^l, ..., v_j^l, ..., v_p^l)$. A vector sequence $[x_{ij}]$ of length L such that

$$x_{ij} = \begin{cases} v_j^1 & \text{if } i = \alpha_j \\ v_j^2 & \text{if } i = \alpha_j + 1 \\ \text{don't care} & \textit{otherwise} \end{cases} \tag{5.2}$$

is said to be an extended vector sequence of $< v_1, v_2 >$ with respect to T_I. A transformation M which transforms from $< v_1, v_2 >$ into the extended vector sequence with respect to T_I is referred to as sequence transformation with respect to T_I.

Lemma 5.4.2. Let $< v_1, v_2 >$ denote a two-pattern test of a given balanced sequential circuit with size n. The time complexity of the sequence transformation on $< v_1, v_2 >$ is $O(n^2)$.

Proof. Let G be a circuit graph of a balanced sequential circuit with sequential depth $L - 2$ and let $< v_1, v_2 >$ be a pair of input vectors of p bits, where $L \leq n$ and $p \leq n$. Let N_V denote the number of vertices in the circuit graph where $N_V \leq n$.

```
Sequence transformation (G, $v_1$, $v_2$)     times
                                             (at most)
 FOR each vertex u                            $N_V$+1
   Assign to u an integer t so that
 equation 5.1 is satisfied
 END FOR
 Compute input timing vector $T_I$              p
 FOR j from 1 to p                              n
   Compute $\alpha_j$
   Compute i
   Compute $x_{ij}$
 END FOR
```

The pseudocode proves the lemma. □

Theorem 5.4.3. [16] *The test generation problem for PDF list F^B of a balanced sequential circuit $\mathcal{S}^{\mathcal{B}}$ can be reduced to the test generation problem for SDF list F^{CB} of its combinational equivalent $C(\mathcal{S}^{\mathcal{B}})$.*

According to Theorem 5.4.3, the PDF test generation of balanced sequential circuits is equivalent to the SDF test generation of its combinational equivalent. The previous section shows that the SDF test generation of the combinational equivalent can be further reduced to the SAF test generation of its segment-leaf-dag (for non-robust test) or its segment-rising(falling)-smooth circuit(for robust test). The following procedure generates a PDF test for a balanced sequential circuit using SAF test generation.

1. Transform the given circuit into its combinational equivalent $C(\mathcal{S}^{\mathcal{B}})$.
2. Follow the procedures of SDF test generation of the combinational circuits for robust test and non-robust test. Let $< v_1, v_2 >$ denote the resulting two-pattern test.
3. Transform $< v_1, v_2 >$ into the two-pattern test sequence using the sequence transformation [16].

Based on Theorem 5.4.3, the theorems and lemmas of the SDF test generation of the combinational circuits, the following corollary is concluded.

Corollary 5.4.4. The PDF test generation of the balanced sequential circuits under rated clock and slow-fast-slow clock is equivalent to the SAF test generation of combinational circuits, which is τ-equivalent.

5.4.2 Path Delay Fault Test Generation Complexity of the Acyclic Sequential Circuits

In this sub-section, the test generation complexity for the internally balanced sequential circuits and acyclic sequential circuits is discussed. Two clocking schemes are considered here, namely slow-fast-slow clock and rated clock. Slow-fast-slow clocking scheme applies slow clock during justification and propagation while rated clocking scheme requires system clock during all the phases of the test generation. Different from PDF test generation, only rated clocking scheme in SAF test generation is considered. In the case of the acyclic sequential circuits, which are not internally balanced, slow-fast-slow clock is assumed so that each PDF P in S^A corresponds to only one SDF S in its time expansion model (TEM) $C_E(S^A)$.

Definition 5.4.5. [26] Let $\mathcal{S}^{\mathcal{A}}$ be an acyclic sequential circuit, and let $G = (V, A, w)$ be the topology graph of $\mathcal{S}^{\mathcal{A}}$. Let $E = (V_E, A_E, t, l)$ be a TEG of G, and let $C_E(\mathcal{S}^{\mathcal{A}})$ be the TEM of $\mathcal{S}^{\mathcal{A}}$ based on E. Let t_{min} be the minimum value of labels assigned to vertices in E, and let d be the sequential depth of $\mathcal{S}^{\mathcal{A}}$. Let $< v_u^1, v_u^2 >$ be a two-pattern vector at a primary input $u \in V_E$ in $C_E(\mathcal{S}^{\mathcal{A}})$. A procedure transforming $< v_u^1, v_u^2 >$ into the input pattern to the primary input $l(u) \in V$ of $\mathcal{S}^{\mathcal{A}}$ at time k (=0, 1, ..., $d+1$) denoted as $I_{l(u)}(k)$ is said to be the sequence transformation γ. That is, for each u,

$$I_{l(u)}(k) = \begin{cases} v_u^1 & \text{if } k = t(u) - t_{min} \\ v_u^2 & \text{if } k = t(u) - t_{min} + 1 \\ \text{don't care} & \textit{otherwise} \end{cases} \tag{5.3}$$

Such an input sequence with the length $d+2$ is regarded as a two-pattern input sequence.

Lemma 5.4.6. Let x and y denote two different primary inputs of $C_E(\mathcal{S}^{\mathcal{A}})$. To avoid conflicts during sequence transformation γ, $v_x^2 = v_y^1$ if $l(x) = l(y) = z$ and $t(y) - t(x) = 1$.

Proof.

$$I_{l(x)}(k) = \begin{cases} v_x^1 & \text{if } k = t(x) - t_{min} \\ v_x^2 & \text{if } k = t(x) - t_{min} + 1 \\ \text{don't care} & \textit{otherwise} \end{cases} \tag{5.4}$$

$$I_{l(y)}(k) = \begin{cases} v_y^1 & \text{if } k = t(y) - t_{min} \\ v_y^2 & \text{if } k = t(y) - t_{min} + 1 \\ \text{don't care} & \textit{otherwise} \end{cases}$$

Let $l(x) = l(y) = z$ and $t(y) - t(x) = 1$. Then,

$$I_{l(y)}(k) = I_z(k) = v_y^1 \text{ if } k = t(y) - t_{min} = t(x) - t_{min} + 1$$

Since $l_{l(x)}(k) = l_z(k) = v_x^2$ if $k = t(x) - t_{min} + 1$, $v_x^2 = v_y^1$. □

Definition 5.4.7. Let x and y denote two different primary inputs of $C_E((\mathcal{S})^{\mathcal{A}})$. x and y are called pattern-dependency input pair (x, y) if $l(x) = l(y) = z$ and $t(y) - t(x) = 1$.

Definition 5.4.8. Given a segment leaf dag $C_S^{LD}((\mathcal{S})^{\mathcal{A}})$ of a TEM $C_E(\mathcal{S}^{\mathcal{A}})$ of $\mathcal{S}^{\mathcal{A}}$, the circuit can be transformed into a pattern-dependency circuit $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$ by the pattern-dependency transformation.

- In the case of non-robust test generation, duplicate C_S^{LD} where the segment leaf dag C_S^{LD} become the second pattern partial circuit C_{2P} while its duplicate become the first pattern partial circuit C_{1P}. In the case of robust test generation, perform the segment-rising-smooth (resp. segment-falling-smooth) transformation.

- For each pattern-dependency input pair (x, y) of $C_E(\mathcal{S}^{\mathcal{A}})$, connect the corresponding x_{2P} and y_{1P} to form a new primary input called unified input w. The resulting circuit is called pattern-dependency circuit $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$.

The idea of pattern-dependency was introduced in [19].

Definition 5.4.9. Given a pattern-dependency circuit $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$. Let C_{1P} and C_{2P} denote the first pattern partial circuit and the second pattern partial circuit of $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$, respectively. Let $v_1^1, v_2^1, ..., v_m^1$ of an m-bit vector v_1 denote a vector bit at each primary input of C_{1P} of $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$ or the stem of the primary input fanout branches fed to C_{1P} of $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$ if the input is a unified input, respectively where m is the number of vector bits. Let $v_1^2, v_2^2, ..., v_m^2$ of an m-bit vector v_2 denote a vector bit at each primary input of C_{2P} of $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$ or the stem of the primary input fanout branches fed to C_{2P} of $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$ if the input is a unified input, respectively where m is the number of vector bits. $d(v_1, v_2)$ denote the input vector of the pattern-dependency circuit $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$.

In the following Lemmas 5.4.10 and 5.4.11, only rising SDF is discussed. The lemmas and proofs for falling SDF can be derived by considering SA1 at the S-edge of S_{2P} and SA0 at the S-edge of S_{1P} in $C_S^{PD}(S^{\mathcal{A}})$.

Lemma 5.4.10. $< v_1, v_2 >$ is a robust(resp. non-robust) test for the $S \uparrow$ in $C_S^{LD}(\mathcal{S}^{\mathcal{A}})$ if and only if $d(v_1, v_2)$ is a test for SA0 at the S-edge of the corresponding segment S_{2P} and SA1 at the S-edge of the corresponding S_{1P} in $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$ with (resp. without) STS.

Proof. **If part:** Let $d(v_1, v_2)$ be a test for SA0 at the S-edge of S_{2P} and SA1 at the S-edge of S_{1P} in $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$. S-edge of S_{2P} is assigned a value 1 while S-edge of S_{1P} is assigned a value 0 under $d(v_1, v_2)$. Based on the definition of the pattern dependency transformation, this implies a rising transition is launched at the S-edge of S under $< v_1, v_2 >$ in C_S^{LD}. According to the test condition, all the side inputs along S_{2P} are at the non-controlling values under the partial vector v_2. This satisfies condition 1 of the robust test and definition of the non-robust test that the transition launched at the S-edge of S propagates to the ending point of S under $< v_1, v_2 >$. With segment-transition-smoothers (STSs) in the case of the robust test, the constants assigned to the primary inputs of $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$ do not propagate to the side-inputs of the OR gates along S_{2P} under $d(v_1, v_2)$. This means no transitions are at the side inputs of the OR gates along S in $C_S^{LD}(\mathcal{S}^{\mathcal{A}})$ under $< v_1, v_2 >$. Condition 2 of the robust test is satisfied. The fault effect of SA0 at the S-edge of S_{2P} is propagated to a primary output under the partial vector v_2. Thus the difference caused by $S \uparrow$ at the ending point of S propagates to the primary output under $< v_1, v_2 >$. Therefore, $< v_1, v_2 >$ is a robust(resp. non-robust) test for the $S \uparrow$ in $C_S^{LD}(\mathcal{S}^{\mathcal{A}})$.

Only if part: Let $< v_1, v_2 >$ be a robust(resp. non-robust) test for the $S \uparrow$. The S-edge of S is assigned a value 0 under vector v_1 and 1 under vector v_2. Thus the SA0 at the S-edge of S_{2P} and the SA1 at the S-edge of S_{1P} in $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$ are activated under the vector $d(v_1, v_2)$. The fault effect of SA1 at the S-edge of S_{1P} propagates to the primary output s_{ed}. The fault effect of SA0 at the S-edge of S_{2P} propagates to a primary output since all the side-inputs along S_{2P} are at the non-controlling values and the difference at the ending point of S_{2P} propagate to the primary output under $d(v_1, v_2)$. □

Lemma 5.4.11. Let P be a path in a given acyclic sequential circuit $\mathcal{S}^{\mathcal{A}}$ and let S_{2P} and S_{1P} be the corresponding segments in its pattern-dependency circuit $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$. Let (v_1, v_2) be a pattern-dependency input pair. Let $\gamma_L(v_1, v_2)$ denote a two-pattern sequence of length L transformed from the two-pattern vector $< v_1, v_2 >$ by the sequence transformation γ. $\gamma_L(v_1, v_2)$ is a robust(resp. non-robust) two-pattern sequence for the $P \uparrow$ in $\mathcal{S}^{\mathcal{A}}$ with sequential depth $L - 2$ if and only if $d(v_1, v_2)$ is a test for SA0 at the S-edge of S_{2P} and SA1 at the S-edge of S_{1P} in $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$ with (resp. without) STS.

Proof. **If part:** Let $d(v_1, v_2)$ be a test for SA0 at the S-edge of S_{2P} and SA1 at the S-edge of S_{1P} in $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$. From Lemma 5.4.10, $< v_1, v_2 >$ is a robust(resp. non-robust) test for the corresponding $S \uparrow$ in $C_S^{LD}(\mathcal{S}^{\mathcal{A}})$ and thus in $C_E(\mathcal{S}^{\mathcal{A}})$. Let c be the combinational block in $C_S^{LD}(\mathcal{S}^{\mathcal{A}})$ that contains $S \uparrow$ is and let $< v_c^1, v_c^2 >$ be the two-pattern vector at the inputs I_c of c under $< v_1, v_2 >$. $S \uparrow$ is launched and propagates to an output of c under $< v_c^1, v_c^2 >$. Therefore, the fault effect of $S \uparrow$ is at the ending point of S. c in $C_E(\mathcal{S}^{\mathcal{A}})$ and $l(c)$ in $\mathcal{S}^{\mathcal{A}}$ are logically same. According to Lemma 5.4.10, there is no pattern conflict in transforming $< v_1, v_2 >$ into $\gamma_L(v_1, v_2)$ since v_1 and v_2 are generated from the pattern-dependency circuit. According to the sequence transformation γ, $I_{l(c)} = v_c^1$ at time $t(c) - t_{min}$, $I_{l(c)} = v_c^2$ at time $t(c) - t_{min} + 1$. Therefore, the $P \uparrow$ in $\mathcal{S}^{\mathcal{A}}$ which corresponds to the $S \uparrow$ in $C_E(\mathcal{S}^{\mathcal{A}})$ is launched and propagates to the output of $l(c)$. The fault effect of $S \uparrow$ at the output of c propagates to a primary output o under $< v_1, v_2 >$. According to Definition 5.4.5, the fault effect of the $P \uparrow$ propagates to the primary output $l(o)$ at time $t(o) - t_{min} + 1$ under the two-pattern input sequence $\gamma_L(v_1, v_2)$.

Only if part: Let $\gamma_L(v_1, v_2)$ be the two-pattern test sequence for the $P\uparrow$. According to Definition 5.4.5, $\gamma_L(v_1, v_2)$ can be inverse transformed to a two-pattern vector $< v_1, v_2 >$ that activates the corresponding $S\uparrow$ in $C_E(\mathcal{S}^{\mathcal{A}})$ and propagates the fault effect to a primary output by the inverse transformation γ^{-1}. Therefore, $d(v_1, v_2)$ is a test for SA0 at the S-edge of S_{2P} and SA1 at the S-edge of S_{1P} in $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$. □

Theorem 5.4.12. *The PDF test generation of the internally balanced sequential circuits under rated clock and slow-fast-slow clock is equivalent to the SAF test generation of combinational circuits, which is τ-equivalent.*

Proof. The following procedure for path delay fault test generation of an acyclic sequential circuit with sequential depth $d = L - 2$ can be used as shown by Lemmas 5.4.10 and 5.4.11.

1. Generate a TEM of $\mathcal{S}^{\mathcal{A}}$;
2. For each $P\uparrow$ (resp. $P\downarrow$),
   - Perform the segment-leaf transformation on $C_E(\mathcal{S}^{\mathcal{A}})$. The resulting circuit $C_S^{LD}(\mathcal{S}^{\mathcal{A}})$ is called segment-leaf-dag.
   - Perform the pattern-dependency circuit transformation. The resulting circuit $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$ is called pattern-dependency circuit.
   - Perform the SAF test generation of SA0 (resp. SA1) at the S-edge of S_{2P} and SA1 (resp. SA0) at the S-edge of S_{1P} in the pattern-dependency circuit $C_S^{PD}(\mathcal{S}^{\mathcal{A}})$. Let $d(v_2, v_1)$ denote the test obtained respectively.
   - Split $d(v_1, v_2)$ into v_1 and v_2 according to the definition of $d(v_1, v_2)$. Transform $< v_1, v_2 >$ into the two-pattern sequence $\gamma_L(v_1, v_2)$ based on Definition 5.4.5.

Let T_{SL}, T_{PD}, T_C^{mSA} and T_P denote the time complexity of the segment-leaf transformation, the pattern-dependency circuit transformation, the double SAF test generation and the two-pattern sequence transformation, respectively. Let N_{SL}, N_{PD}, N_C^{mSA} and N_P denote the problem size of the segment-leaf transformation, the pattern-dependency circuit transformation, the double SAF test generation and the two-pattern sequence transformation, respectively. According to Lemmas 5.3.7-5.3.9 and for $N_{SL} = n$, $n \leq N_{PD} \leq 2n$, $2n \leq N_C^{mSA} \leq 6n-5$, $N_P \leq n$. Therefore, the PDF test generation complexity is

$$\begin{aligned} T_{IB}^{PD}(n) &= T_{SL}(N_{SL}) + T_{PD}(N_{PD}) + O(T_C^{mSA}(N_C^{mSA})) + T_P(N_P) \\ &= O(N_{SL}^2) + O(N_{PD}^2) + O(\tau(N_C^{mSA})) + O(N_P^2) \\ &\leq O(n^2) + O(4n^2) + O(\tau(6n)) + O(n^2) \\ &= O(\tau(n)) \end{aligned}$$

Therefore, the PDF test generation of internally balanced sequential circuits is equivalent to the SAF test generation of S-edges of pattern dependency circuits, which is τ-bounded. To show that the PDF test generation of internally balanced sequential circuits is τ-equivalent, it is needed to prove that the SAF test generation of combinational circuits is equivalent to a subclass of internally balanced sequential circuits. Since PDF test generation of balanced sequential circuits is τ-equivalent and PDF test generation of internally balanced sequential circuits is shown τ-bounded in the above-mentioned, the PDF test generation of internally balanced sequential circuits is equivalent to the SAF test generation of combinational circuits, which is τ-equivalent. □

Theorem 5.4.13. *The PDF test generation of the acyclic sequential circuits is τ^2-bounded under slow-fast-slow clock.*

Proof. Let T_{SL}, T_{PD}, T_C^{mSA} and T_P denote the time complexity of the segment-leaf transformation, the pattern-dependency circuit transformation, the double SAF test generation and the two-pattern sequence transformation, respectively. Let N_{SL}, N_{PD}, N_C^{mSA} and N_P denote the problem size of the segment-leaf transformation, the pattern-dependency circuit transformation, the double SAF test generation and the two-pattern sequence transformation, respectively. Let $\mathcal{S}^{\mathcal{A}}$ denote a given acyclic sequential circuit. The size of its TEM $C_E(\mathcal{S}^{\mathcal{A}})$ is $(d+1)n$ where d is the sequential depth. [11,12] illustrated an acyclic sequential circuit of size n that goes through duplications and the size increases up to $\Theta(n^2)$ under TEM representation. Lemmas 5.4.10 and 5.4.11 shows the equivalence of the PDF test generation of a given acyclic sequential circuit and the SAF test generation of

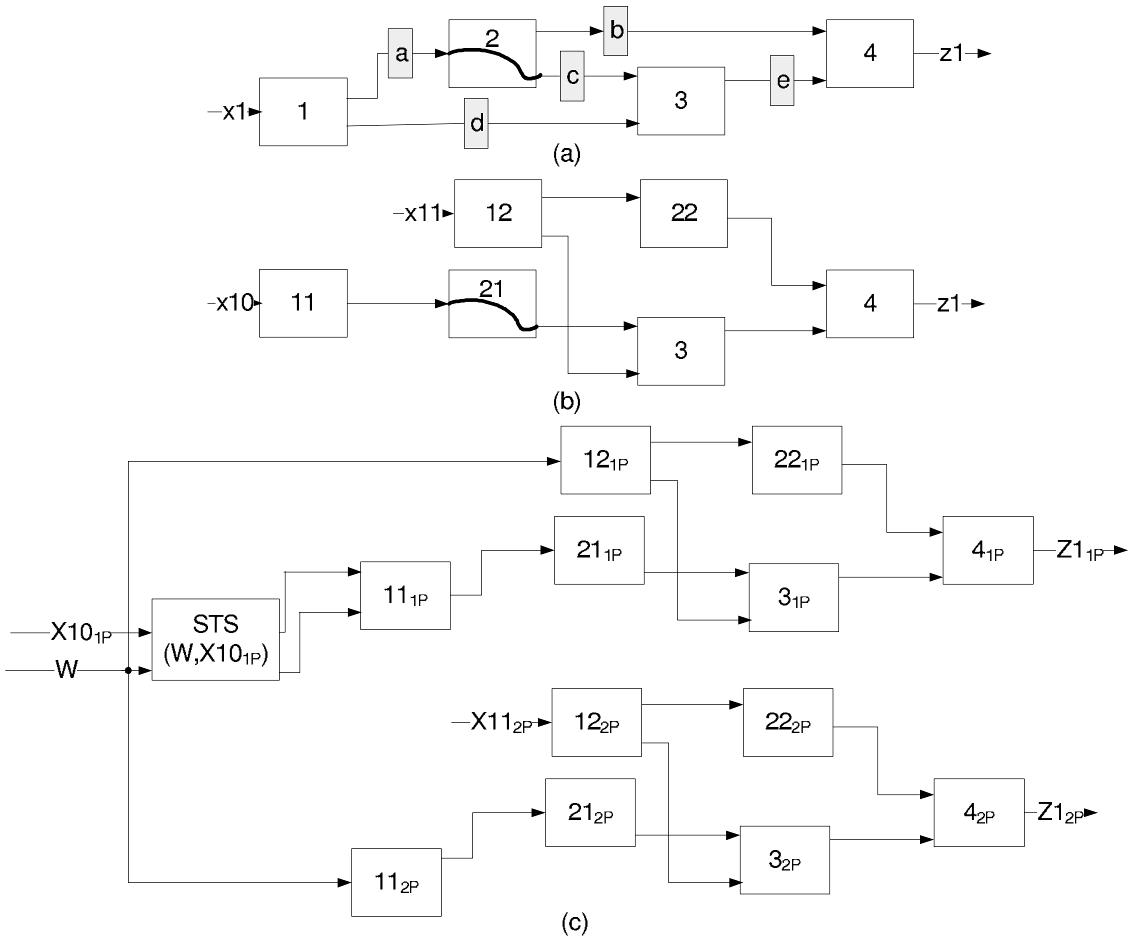


Figure 5.6. (a) An acyclic sequential circuit. (b) Its time expansion model. (c) Its pattern-dependency circuit.

its pattern dependency circuit. For $n \leq N_{SL} \leq (d+1)n$, $n \leq N_{PD} \leq 2((d+1)n)$, $2n \leq N_C^{mSA} \leq 6((d+1)n)$, $N_P \leq n$. Therefore, the PDF test generation complexity is

$$\begin{aligned} T_A^{PD}(n) &= T_{SL}(N_{SL}) + T_{PD}(N_{PD}) + T_C^{mSA}(N_C^{mSA}) + T_P(N_P) \\ &= O(N_{SL}^2) + O(N_{PD}^2) + O(\tau(N_C^{mSA})) + O(N_P^2) \\ &= O(d^2n^2) + O(4d^2n^2) + O(\tau(6dn)) + O(n^2) \\ &= O(\tau^2(n)) \text{ for } d = O(n) \end{aligned}$$

If $d = 0$, the PDF test generation of these acyclic sequential circuits is τ-equivalent. As shown in [11,12], the SAF test generation of acyclic sequential circuits is τ^2-bounded. Therefore, the PDF test generation and SAF test generation of acyclic sequential circuits are τ^2-bounded. □

[11,12] stated that the SAF test generation of acyclic sequential circuits is not τ-equivalent under TEM. In other words, the following is still an open problem.

Conjecture 5.4.1. [11, 12] *The SAF test generation of acyclic sequential circuits is not τ-equivalent.*

By using the same example of acyclic sequential circuit in [1], the following theorem is proved.

Theorem 5.4.14. *The PDF test generation of acyclic sequential circuits is not τ-equivalent under TEM with slow-fast-slow clock.*

For PDF test generation, there might be also other PDF test generation models for acyclic sequential circuits besides TEM. Consequently, the following conjecture is concluded.

Conjecture 5.4.2. *The PDF test generation of acyclic sequential circuits is not τ-equivalent.*

Example 5.4.1. Figure 5.6 shows the transformations of an acyclic sequential circuit to represent its PDF test generation problem based on SAF test generation. Note that only one PDF is considered, that is in block 21, under slow-fast-slow clock.

Besides the PDF test generation under slow-fast-slow clock, there are analogous issues for the case under rated clock testing and they remain open problems.

Open Problem 1. Is the PDF test generation of acyclic sequential circuits τ^2-bounded under rated clock?

Open Problem 2. Is the PDF test generation of acyclic sequential circuits not τ-equivalent under TEM with rated clock?

Open Problem 3. Is the PDF test generation of acyclic sequential circuits not τ-equivalent under rated clock?

Open Problem 4. Is the PDF test generation of acyclic sequential circuits under rated clock equivalent to the SAF test generation of acyclic sequential circuits?

5.5. PDF Test Generation Complexity of Cyclic Sequential Circuits

Generally, PDF test generation (Figure 5.7) for a cyclic sequential circuit involves the following three phases.

1. **Initialization.** In this phase, an input sequence is derived so that a signal transition is produced at the origin of the path.
2. **PDF excitation.** PDF excitation means the transition is propagated through the path. A two-pattern vector is derived to activate a PDF.
3. **Error propagation.** In this phase, an input sequence is derived to allow observation of the value at the destination flip-flop of the path (if the destination is a primary output, this step is not needed.)

Definition 5.5.1. Let P and P' denote a path in a given cyclic sequential circuit $\mathcal{S}^{\mathcal{C}}$ and the corresponding path in its combinational part c. A duplex combinational circuit $C_P^D(\mathcal{S}^{\mathcal{C}})$ for P (Figure 5.8) of a cyclic sequential circuit $\mathcal{S}$ can be obtained by the following transformation:

- Perform the single-path-leaf transformation for P' on c. The inputs of c corresponding to the outputs of the flip-flops in $\mathcal{S}^{\mathcal{C}}$ are called the pseudo inputs while the outputs of c corresponding to the inputs of the flip-flops in $\mathcal{S}^{\mathcal{C}}$ are called the pseudo outputs. The resulting single-path leaf-dag is denoted by $c_{P'}^{LD}$.

- Duplicate $c_{P'}^{LD}$. The single-path leaf-dag $c_{P'}^{LD}$ and its duplicate are named as the first partial circuit c_1 and the second partial circuit c_2.

- Connect the pseudo outputs of c_1 to the corresponding pseudo inputs of c_2 to form an iterative logic array of single-path leaf-dag of size 2. The new connections between c_1 and c_2 are called pseudo interconnections and a pseudo interconnection is labeled as QD_i while a resulting pseudo input and pseudo output are labeled as Q_i and D_i respectively, corresponding to flip-flop i in $\mathcal{S}^{\mathcal{C}}$. Note that the path P in $\mathcal{S}^{\mathcal{C}}$ corresponds to two segments S_{c1} and S_{c2} in $C_P^D(\mathcal{S}^{\mathcal{C}})$. A primary input and a primary output of c_1 is denoted by I_{1j} and O_{1k}, respectively while a primary input and a primary output of c_2 is denoted by I_{2j} and O_{2k}.

Definition 5.5.2. Let $P\uparrow$ denote a rising PDF in a given cyclic sequential circuit $\mathcal{S}^{\mathcal{C}}$. A duplex combinational circuit $C_P^D(\mathcal{S}^{\mathcal{C}})$ for P can be transformed into a path-rising-smooth duplex circuit(resp. path-falling-smooth duplex circuit) $C_S^{PRS}(\mathcal{S}^{\mathcal{C}})$ (resp. $C_S^{PFS}(\mathcal{S}^{\mathcal{C}})$) for the corresponding segment S_{c2} by the following procedure:

- Let Q_{OR} (resp. Q_{AND}) denote the OR gates (resp. AND gates) along S_{c2} corresponding to $P\uparrow$ (resp. $P\downarrow$). A gate may have no parity, 0, 1 or both parities. A side-input to an OR gate (resp. AND gate) in Q_{OR} (resp. Q_{AND}) has parity 1 (resp. 0). Perform a reverse topological traversal from the transitive fanout of all pseudo interconnections, to determine the parity of all gates along the side-paths to S_{c2}. The parity is complemented across a NOT gate. If some fanouts of a gate have parity 1 and others have parity 0, the gate is assigned both parities.

- Duplicate gates so that each resulting gate has parity of either nothing, 0 or 1 but not both.

  - Traversing from pseudo output or primary output g_m on P, for each gate h_j with a parity (parities) and with a successor gate that is off path and without parity, h_j and the connections of its immediate predecessor gates are duplicated once and its duplicate h'_j has no parity. For each immediate successor gate h_{j+1} of h_j that has no parity, the connection from h_j to h_{j+1} is replaced by the connection from h'_j to h_{j+1}.
  - Traversing from pseudo output or primary output g_m, each gate h_j with both parities and the connections to its immediate predecessor gates are duplicated once and assigned parity 1 while its duplicate h'_j is assigned parity 0. For each immediate successor gate h_{j+1} of h_j that has parity 0 (1 if there is an inversion between h_j and h_{j+1}), the connection from h_j to h_{j+1} is replaced by the connection from h'_j to h_{j+1}.

- Insert to the fanout branch of a second circuit primary input I_{2j} a segment-transition-smoother STS(I_{2j}, I_{1j}) if the fanout branch has an immediate gate with parity 0 or 1. At the fanout branch of a pseudo interconnection QD_i, insert a segment transition-smoother STS(QD_i, Q_i) if the QD_i has an immediate gate with parity 0 or 1.

Lemma 5.5.3. $< v_1, v_2 >$ is an input sequence that robustly excites a PDF $P \uparrow$ (resp. $P \downarrow$) of a sequential circuit $\mathcal{S}^{\mathcal{C}}$ in present state s_1 if and only if $s_1 \sim v_1 \sim v_2$ is a test for SA0 at the S-edge of S_{c2} with an input constraint of 0 at the S-edge of S_{c1} in the corresponding duplex combinational circuit $C_S^{PRS}(\mathcal{S}^{\mathcal{C}})$.

Proof. **If part:** $s_1 \sim v_1 \sim v_2$ is a test for SA0 at the S-edge of S_{c2} with an input constraint 0 at the S-edge of S_{c1}. In the fault free case, S-edge of S_{c1} and S-edge of S_{c2} have different values under $s_1 \sim v_1 \sim v_2$. This means $< v_1, v_2 >$ initializes the corresponding $P \uparrow$. The value at the ending points of S_{c2} and S_{c1} are 0 and 1 respectively under $s_1 \sim v_1 \sim v_2$. All the side-inputs of the gates along S_{c2} are at the non-controlling values. Thus, all the side-input of gates along P are at the non-controlling value under v_2. Segment transition-smoother guarantees that the transition at a flip-flop or a primary input does not propagate to the side-inputs of OR gates along P under $< v_1, v_2 >$. Therefore, $< v_1, v_2 >$ robustly excites

$P \uparrow$.

Only If part: $< v_1, v_2 >$ robustly excites $P \uparrow$. All side inputs of gates along P are at the non-controlling values. The value at the S-edge of S_{c2} is different from that at the S-edge of S_{c1}. This excites the SA0 at the S-edge of S_{c2} and put a constraint 0 to the S-edge of S_{c1}. According to the definition of robust PDF, the side inputs of gates along S_{c2} are at the non-controlling value. This satisfies the test condition. Therefore, $s_1 \sim v_1 \sim v_2$ is a test for SA0 at the S-edge of S_{c2} with an input constraint of 0 at the S-edge of S_{c1}. $\square$

Lemma 5.5.4. $< v_1, v_2 >$ is an input sequence that non-robustly excites a PDF $P \uparrow$ (resp. $P \downarrow$) of a sequential circuit $\mathcal{S}^{\mathcal{C}}$ in present state s_1 if and only if $s_1 \sim v_1 \sim v_2$ is a test for SA0 at the S-edge of S_{c2} with an input constraint of 0 at the S-edge of S_{c1} in the corresponding path-rising-smooth duplex circuit $C_S^{PRS}(\mathcal{S}^{\mathcal{C}})$.

Proof. **If part:** $s_1 \sim v_1 \sim v_2$ is a test for SA0 at the S-edge of S_{c2} with an input constraint 0 at the S-edge of S_{c1}. In the fault free case, S-edge of S_{c2} and S-edge of S_{c1} have the different values under $s_1 \sim v_1 \sim v_2$. This initializes the $P \uparrow$. The value at the ending points of S_{c2} and S_{c1} are 0 and 1 respectively under $s_1 \sim v_1 \sim v_2$. All the side-input of gates along P are at the non-controlling value. Therefore, $< v_1, v_2 >$ non-robustly excites $P \uparrow$.

Only If part: $< v_1, v_2 >$ non-robustly excites $P \uparrow$. All the side inputs of gates along P are at the non-controlling values. The value at the S-edge of S_{c2} is different from that at the S-edge of S_{c1}, which is 1 and 0 respectively. This excites the SA0 at the S-edge of S_{c2} and put a constraint 0 to the S-edge of S_{c1}. According to the definition of PDF, the side inputs of gates along S_{c2} are at non-controlling value. This satisfies the test condition. Therefore, $s_1 \sim v_1 \sim v_2$ is a test for SA0 at the S-edge of S_{c2} with an input constraint of 0 at the S-edge of S_{c1}. $\square$

Lemma 5.5.5. The PDF excitation is equivalent to the SAF test generation of S-edges of path-rising-smooth duplex circuits, which is τ-bounded.

Proof. Pseudo-transformation transforms the kernel of a given cyclic sequential circuit so that the PDF activation can be done by combinational test generation. To activate a PDF in a given sequential circuit $\mathcal{S}^{\mathcal{C}}$, for each PDF

1. Derive the duplex combinational circuit $C^D(\mathcal{S}^\mathcal{C})$.
2. Derive the path-rising-smooth duplex circuit $C_S^{PRS}(\mathcal{S}^\mathcal{C})$ (for robust test).
3. Perform the SAF test generation of SA0 at the S-edge of S_{c2} with an input constraint 0 at the S-edge of S_{c1}. Let $s_1 \sim v_1 \sim v_2$ denote the test obtained.
4. Transform $s_1 \sim v_1 \sim v_2$ into an input sequence $< v_1, v_2 >$ for the PDF activation.

Based on Lemmas 5.5.3-5.5.4, the PDF excitation can be modeled by the SAF test generation. Let $T_D(N_D)$, $T_{PRS}(N_{PRS})$, $T_c^{SAF}(N_c^{SAF})$ and $T_P(N_P)$ denote the time complexity of the duplex combinational circuit derivation, the path-rising-smooth duplex circuit derivation, the SAF test generation and the two-pattern test transformation. Let n denote the size of a given cyclic sequential circuit $\mathcal{S}^\mathcal{C}$. To derive a duplex combinational circuit, it takes $O(n^2)$ according to Definition 5.5.1. After the derivation, the size of the resulting circuit is at most $4n$. The procedures of transforming a C^D into a C_S^{PRS} involve only the first partial circuit C_1 of C^D. The process takes $O(n^2)$ as explained in Definition 5.5.2. The size of the resulting circuit is at most $8n$. Therefore, the time complexity of the PDF excitation is

$$\begin{aligned} T_E^{PD} &= T_D(N_D) + T_{PRS}(N_{PRS}) + O(T_c^{SAF}(N_c^{SAF})) + T_P(N_P) \\ &= O(n^2) + O(4n^2) + O(\tau(8n)) + O(n) \\ &= O(\tau(n)). \end{aligned}$$

□

Definition 5.5.6. A combinational circuit C with a SAF f can be transformed into a cyclic sequential circuit $S_f^{c\delta}$ (Figure 5.9) by ***cyclic δ transformation***:

S1. Let $o_1, ..., o_p$ denote the primary outputs of C. Let $c_1, ..., c_p$ denote the XOR function of each primary output of C and the corresponding primary output of the faulty circuit C_f. Let $G(C, C_f)$ be the circuit realizing c_1 OR ...OR c_p.

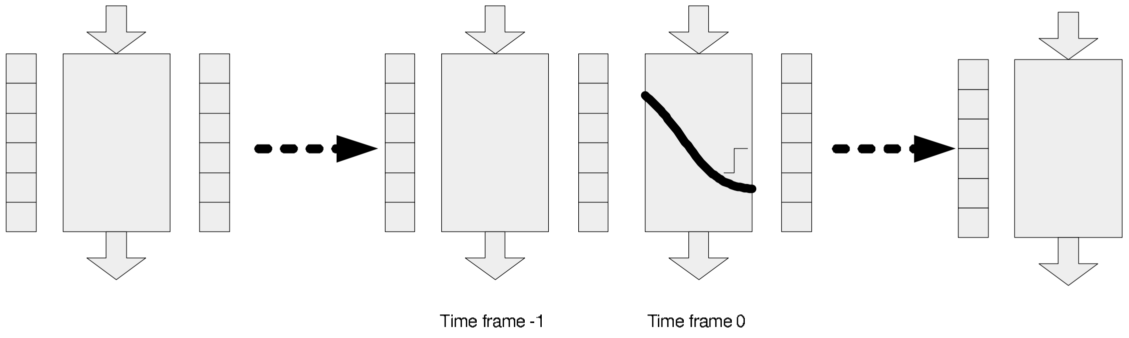


Figure 5.7. PDF test generation for cyclic sequential circuit.

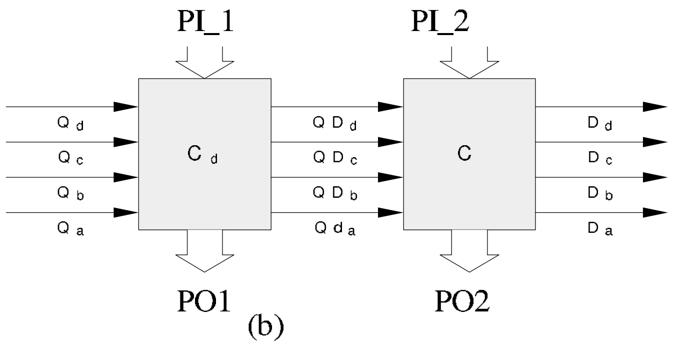


Figure 5.8. A duplex combinational circuit.

S2. Connect the output of $G(C, C_f)$ to a two-input AND gate. The output O of the AND gate A is fed back to the AND gate through an inverter I.

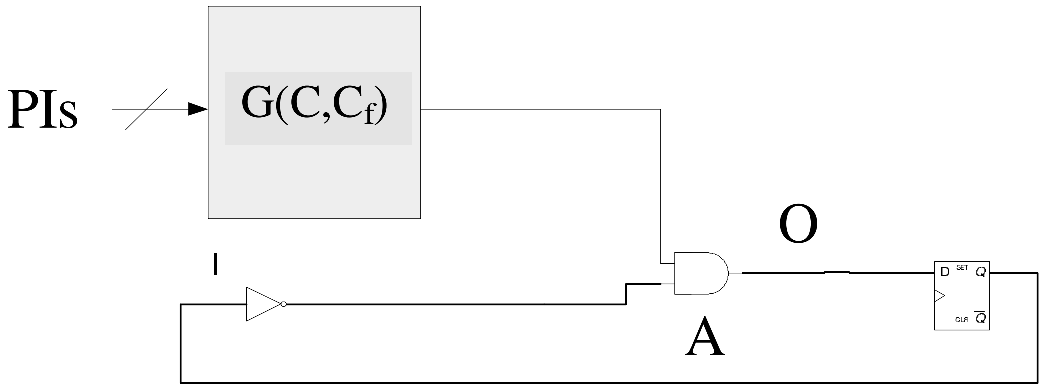


Figure 5.9. A cyclic sequential circuit $S_f^{c\delta}$.

Lemma 5.5.7. v is a test for SAF f in a combinational circuit C if and only if $< v, v >$ excites the PDF $IAO \uparrow$ or $IAO \downarrow$ in the corresponding cyclic sequential circuit $S_f^{c\delta}$.

Proof. Under $< v, v >$, all the side-inputs are at stable non-controlling values. Therefore, the transition along IAO can be propagated to the flip-flop and thus the PDF is excited. □

Theorem 5.5.8. *The PDF excitation is τ-equivalent.*

Proof. To prove that the PDF excitation is τ-equivalent, in addition to Lemma 5.5.5, it is sufficient to prove that the SAF test generation of combinational circuits can be transformed into the PDF excitation of cyclic sequential circuits. Therefore, Lemmas 5.5.5 and 5.5.7 prove it. □

For easily testable sequential circuits, the PDF test generation complexity depends on the time complexity of the justification and the state differentiation. If the time complexity is less than or equivalent to $\tau(n)$, the cyclic sequential circuits is τ-equivalent. Therefore, the following corollary is concluded.

Corollary 5.5.9. The PDF test generation under rated clock and slow-fast-slow clock and SAF test generation of t-time-bounded testable circuits are τ-equivalent (resp. τ^2-bounded) if $t = \tau(n)$ (resp. $t = \tau^2(n)$).

Corollary 5.5.10. The PDF test generation under rated clock and slow-fast-slow clock and SAF test generation of l-length-bounded testable circuits are τ^2-bounded if $l = O(n)$.

5.5.1 PDF Test Generation and SAF Test Generation of the State-Shiftable Finite-State-Machine Realizations

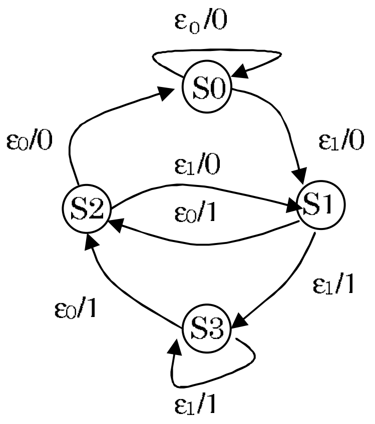
Formally, a finite state machine FSM is defined as a 5-tuple (I, ST, O, DEL, GAM) where I is a set of input symbols, ST is a set of states, O is a set of output symbols, $DEL : I \times ST \rightarrow ST$ is the next-state function, and GAM is the output function[35]. A ***state-shiftable finite state machine*** [23] is a machine that possesses

1. transfer sequences of length at most $[\log_2 m]$ to carry the machine from state s_0 to state s_i for all i, and
2. distinguishing sequences of length $[\log_2 m]$, which are arbitrary input sequences consisting of 2 input symbols, where m denotes the number of states.

Figures 5.10(a) and (b) show the state diagrams and state tables for a distributive SSFSM.

A sequential circuit that is realized from the SSFSM is called SSFSM realization. In this section, two easily testable classes of state-shiftable finite-state-machine (SSFSM) realizations, namely two-column SSFSM realizations with observable shifting logic (2COS-SSFSM) and two-column distributive SSFSM realizations are introduced. The latter class contain a submachine called distributive SSFSM.

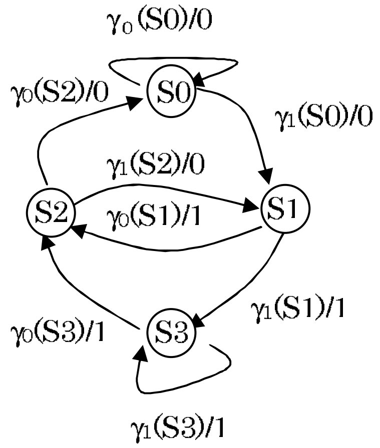
Definition 5.5.11. ***Distributive SSFSM*** is a two-column SSFSM realization with different pairs of input symbols for each state. Let the input symbols of two-column for state s_j be denoted by $\gamma_0(s_j)$ and $\gamma_1(s_j)$, respectively. Let ϵ_0 and



(a) State diagram of SSFSM.

IS / PS	γ_0…………γ_i	ε_0	ε_1
S0	S3………S1	S0	S1
S1	S1………S2	S2	S3
S2	S0………S1	S0	S1
S3	S2………S3	S2	S3

(b) State table that has two-column corresponding to SSFSM.



(c) State diagram of distributive SSFSM.

IS / PS	I_0	I_1	I_2	I_3	ε_0	ε_1
S0	$\boxed{S0}$	S3	S3	$\boxed{S1}$	S0	S1
S1	S0	$\boxed{S2}$	S1	$\boxed{S3}$	S2	S3
S2	S3	$\boxed{S0}$	$\boxed{S1}$	S2	S0	S1
S3	$\boxed{S2}$	$\boxed{S3}$	S1	S1	S2	S3

(d) State table that has two-column SSFSM with different pairs of input symbols for each present state where $\gamma_0(S_0)=I_0$, $\gamma_1(S_0)=I_3$, $\gamma_0(S1)=I_1$, $\gamma_1(S1)=I_3$, $\gamma_0(S2)=I_1$, $\gamma_1(S2)=I_2$, $\gamma_0(S3)=I_0$, $\gamma_1(S3)=I_1$

Figure 5.10. State diagrams and state tables for SSFSM and distributive SSFSM.

ϵ_1 denote the input symbols of a two-column SSFSM, which has same degree with the distributive SSFSM. For each j, the next state function δ is such that $\delta(\epsilon_0, s_j) = \delta(\gamma_0(s_j), s_j)$ and $\delta(\epsilon_1, s_j) = \delta(\gamma_1(s_j), s_j)$.

Figures 5.10(c) and (d) show the state diagrams and state tables for a distributive SSFSM.

The test generation complexity of these classes and the relationship of their PDF test generation and SAF test generation are addressed. Only the slow-fast-slow clock is considered and thus the circuit under test is fault-free during justification and differentiation. By this assumption, the PDF test generation is expected to be easier or equivalent to the SAF test generation for these two classes.

Definition 5.5.12. ***Two-column SSFSM realizations with observable shifting logic*** (2COS-SSFSM) is an SSFSM realizations that satisfies the following conditions:

C1. There exists a two-column submachine of SSFSM of degree m, where $m = O(n)$ and n is the size of the 2COS-SSFSM. Let the the input symbols of the two-column be denoted by ϵ_0 and ϵ_1.

C2. Let C_0 and C_1 denote the input combinations (cubes) of ϵ_0 and ϵ_1, respectively, after the input assignments. The logic that is equivalent to C_0 *OR* C_1 is called shifting logic and is a fanout of each next state function D_i and the output O_{ss} of SSFSM, where $1 \le i \le m-1$. The shifting logic S_L is made observable.
$S_L = C_0$ OR C_1
$D_0 = C_0$ OR F_0
$\vdots$
$D_i = (C_0$ OR $C_1)$ AND Q_{i-1} OR F_i
$\vdots$
$D_{m-1} = (C_0$ OR $C_1)$ AND Q_{m-2} OR F_{m-1}
$O_{ss} = (C_0 + C_1)$ AND Q_{m-1} OR F_{Oss}
Q_i (resp. D_i) is an output (resp. input) of flip-flop bit i. Gate sharing is permitted during the synthesis. A OR B (resp. A AND B) means both

logics are ORed (resp. ANDed) together. Figure 5.11 shows the general block diagram of a 2COS-SSFSM.

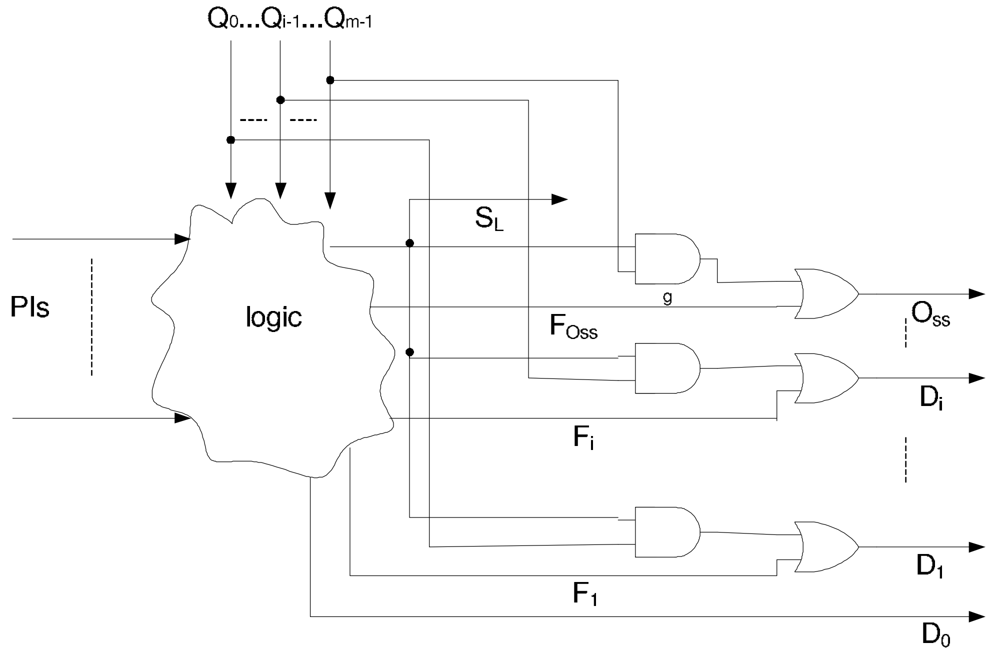


Figure 5.11. General block diagram of a two-column SSFSM realization with observable shifting logic.

To generate an SAF test of a 2COS-SSFSM, a constraint is put on the derivation of excitation state, that is the input vectors of C_0 and C_1 as the test vectors are prioritized.

Theorem 5.5.13. *The PDF test generation under slow-fast-slow clock and SAF test generation of 2COS-SSFSM is equivalent, which are τ-equivalent.*

Proof. **PDF test generation:** The process consists of three steps, including PDF activation, justification and differentiation. In PDF activation, the input sequence $< v_1, v_2 >$ is derived. Let the excitation state be denoted by s_e. According to Theorem 5.5.8, the running time of PDF activation is $\Theta(\tau(n))$, which is reducible to the SAF activation (the derivation of excitation state for an SAF). Since slow-fast-slow clock is used, there is no fault effect during justification and differentiation. Therefore, any excitation state is justifiable by an input sequence consisting of ϵ_0 and ϵ_1 with length at most $m-1$ while a pair of faulty and fault-free next states after the PDF activation can be differentiated by any input sequence consisting of ϵ_0 and ϵ_1 with length at most $m-1$, where m is the degree of SSFSM. Obviously the PDF test generation is τ-equivalent.

SAF test generation: In the following, several cases of faults are discussed. For each case, the justification sequence is an input sequence consisting of ϵ_0 and ϵ_1 with length at most $m-1$. If the fault is activated and propagates to one of the flip-flops before the circuit reaches the excitation state, the current state is made an excitation state. Since shifting logic S_L is observable, all the faults propagate to S_L are detectable. For each testable fault f in the fanout-free region, it is guaranteed to be detectable without fault masking during the shifting operation once the fault effect propagates to a flip-flop. Let the fault effect denoted by $\bar{d}$ that propagates to Q_h after SAF activation where $0 \leq h \leq m-1$.

$$D_{h+1} \;=\; S_L Q_h + F_{h+1}$$

For $S_L = 1$ during shifting operation,

$$D_{h+1} = \bar{d} + F_{h+1}$$

Note that no other fault effects propagate to F_{h+1} as f is in the fanout-free region and F_{h+1} has a fault-free value 0 during the shifting operation ($S_L = 1$). Therefore $D_{h+1} = \bar{d}$

The fault effect is propagated to $D_{h+2}, D_{h+3}, ..., D_{m-1}$ and then output O_{ss} for m is the degree of SSFSM. For example, the output of AND gate g is in fanout-free region (Figure 5.11).

In generating tests for other faults which are in fanout region, we can assume the shifting operation is always working for S_L is observable. Let's discuss the faults in the fanout region other than those at the stem of Q_k for $0 \leq k \leq m-1$. During the phase of SAF activation, any input cubes contained by C_0 or C_1 are prioritized as the input pattern to excite the fault f. If one of them is a test, the fault effect that propagates to a flip-flop must be $\bar{d}$ as discussed in the following.

$$D_i \;=\; (C_0 + C_1) \text{ AND } Q_{i-1} \text{ OR } F_i$$

F_i has fault-free value 0 during shifting operation. In addition, shifting logic S_L and Q_{i-1} are fault free. Therefore the fault effect propagates to F_i must be $\bar{d}$ after SAF activation if there exists an input cube that is contained by C_0 or C_1 and excites the fault f. During differentiation,

$$\begin{aligned} Q_i &= \bar{d} \\ D_{i+1} &= \bar{d} + F_{i+1} \end{aligned}$$

Since this is done by shifting operation where $S_L = 1$ and each stem of Q_k for $0 \le k \le m-1$ is fault-free, if the fault f is activated again during differentiation, it will propagate to F_{i+1} as $\bar{d}$, which has the same polarity as the fault effect during SAF activation. Therefore, fault masking does not happen during differentiation. For the faults at the stems of Q_k where $0 \le k \le m-1$ and let $Q_k = Q_{i-1}$, boolean difference is used to prove that the polarity of fault effect is also always same during shifting operation.

$$\begin{aligned} D_i &= (C_0 + C_1)Q_{i-1} + F_i \\ &= (C_0 + C_1)Q_{i-1} + R\bar{Q}_{i-1}G_i + SQ_{i-1}H_i + F_i' \end{aligned}$$

F_i' does not contain literal Q_{i-1}. G_i and H_i are the disjunctions of the cubes that contain literals Q_j for $j \neq i-1$. R and S are the disjunctions of input cubes that do not contain and are not contained by C_0 or C_1. For Q_{i-1} s-a-0,

$$\begin{aligned} h &= Q_{i-1} \\ D_i &= (C_0 + C_1)h + R\bar{h}G_i + ShH_i + F_i' \\ h\frac{dD_i}{dh} &= (h(C_0 + C_1 + SH_i) + hF_i') \oplus (hRG_i + hF_i') \\ &= Q_{i-1}(C_0 + C_1) + Q_{i-1}H_iS + Q_{i-1}G_iR \text{ by redundance law} \end{aligned}$$

The similar proof can be done for the case where Q_{i-1} s-a-1. Note that the fault effect of s-a-0 (resp. s-a-1) d (resp. $\bar{d}$) propagates to D_i as d (resp. $\bar{d}$) if the vectors contained by $Q_{i-1}(C_0 + C_1)$ or $Q_{i-1}H_iS$ is applied while propagates to D_i as $\bar{d}$ (resp. d) if the vectors contained by $Q_{i-1}G_iR$ is applied. During SAF activation, the input cubes contained by C_0 or C_1 is prioritized. Therefore, if there exists an input pattern contained by C_0 or C_1, the fault effect that propagates to a flip-flop during SAF activation is d, i.e. $D_i = d$. Let the fault effect propagate to Q_{i-1} after SAF activation of a fault f in the fanout region and let the input cube that activates the fault denoted by X_c. During the phase of differentiation, the fault effect being shifted is not masked because only the fault effect of the same polarity may appear at F_j for $j \geq i$ under input sequence consisting of X_c. Therefore, by giving the input cubes contained by C_0 and C_1 a priority to be an input vector to excite the fault, fault masking does not happen during state differentiation. In the case where no input cubes that are contained by either C_0 or C_1 excite the fault during SAF activation, the fault is not excited during the phase of differentiation and thus the fault masking does not occur. Let T_E, T_J and T_D denote the running time of SAF excitation, justification and differentiation respectively.

$$\begin{aligned}
T^{SA}_{2COS}(n) &= T_E + T_J + T_D \\
&= \tau(n) + O(m-1) + O(m-1) \\
&= \tau(n) + O(n) + O(n) \text{ for m=O(n)} \\
&= \Theta(\tau(n))
\end{aligned}$$

□

However, it is still unsolved for the case under rated clock.

Open Problem 5. Are the PDF test generation under rated clock and the SAF test generation of two-column SSFSM realizations with observable shifting logic equivalent?

Definition 5.5.14. ***Two-column distributive SSFSM*** (2CD-SSFSM) is SS-FSM realizations that fulfills the following conditions:

C1. There exists a two-column submachine of SSFSM of degree m, where $m = O(n)$ and n is the size of the 2CD-SSFSM. Let the the input symbols of the two-column be denoted by ϵ_0 and ϵ_1.

C2. There exists a distributive submachine of SSFSM over the columns other than those of ϵ_0 and ϵ_1. The number of columns is $2^{N-1} + 2$ where N is an integer.

C3. There exists an input symbol c of the distributed submachine of SSFSM and a state s_j for $0 \leq j \leq m-1$ such that $s_k = \delta(c, s_j)$ and $s_k \neq \delta(\epsilon_0, s_j)$, $s_k \neq \delta(\epsilon_1, s_j)$ for $k \neq j$ and $0 \leq k \leq m-1$.

C4. Let C_0 and C_1 denote the input combinations of ϵ_0 and ϵ_1 respectively after the input assignment. C_0 and C_1 are two-bit assignments, one bit of which is different to each other. E.g. $C_0 = ab'$ and $C_1 = ab$ where a and b are variables for primary inputs. The similar bit of C_0 and C_1 is complement to other input combinations.
$SHIFT = C_0 + C_1$ $D_0 = C_0$ OR F_0
$\vdots$
$D_i = (C_0 + C_1)$ AND Q_{i-1} OR F_i
$\vdots$
$D_{m-1} = (C_0 + C_1)$ AND Q_{m-2} OR F_{m-1}
$O_{ss} = (C_0 + C_1)$ AND Q_{m-1} OR F_{Oss}
Q_i (resp. D_i) is an output (resp. input) of flip-flop bit i. Gate sharing is permitted during the synthesis. A OR B (resp. A AND B) means both implicants are ORed (resp. ANDed) together.

Figure 5.12 shows the general block diagram of a 2CD-SSFSM.

Theorem 5.5.15. *The PDF test generation under slow-fast-slow of the two-column distributive SSFSM is τ-equivalent while the SAF test generation complexity of the same class is τ^2-bounded.*

Proof. **PDF test generation:** Same as the proof of PDF test generation in Theorem 5.5.13
SAF test generation: In the following, several cases of faults are discussed. For each case, the justification sequence is an input sequence consisting of ϵ_0 and ϵ_1 with length at most $m-1$. If the fault is activated and propagates to one of the flip-flops before the circuit reaches the excitation state, the current state is made an excitation state. The realization of logic C_0+C_1 is eventually an input, namely SHIFT that is ANDed with Q_{i-1}. Let's first discuss about the faults at SHIFT. Shifting operation fails when a s-a-0 occurs at SHIFT or when a s-a-0 occurs at the output of (SHIFT AND Q_{i-1}). The latter is always activated and differentiated

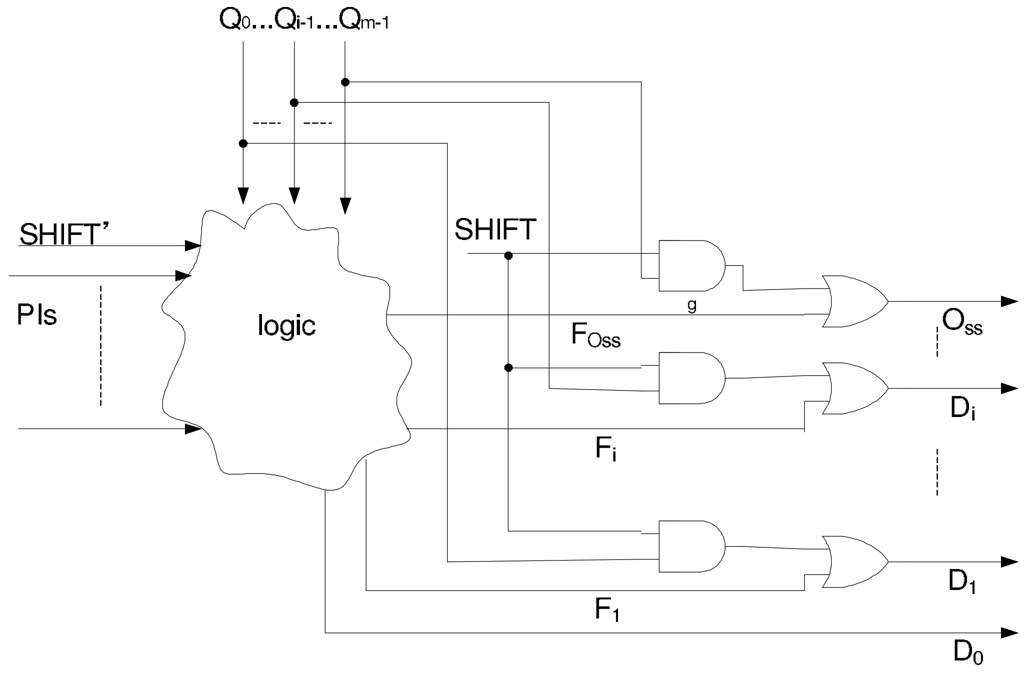


Figure 5.12. General block diagram of a two-column distributive SSFSM realization.

in shifting operation since the fault location is in the fanout-free region and the shifting operation of all the more significant flip-flops are still working. Looking in the former case where the fault is at the stem of SHIFT, the shifting operation of the circuit fails. However, the distributive shifting operation is intact. First, the SAF activation is performed at running time $\tau(n)$. To differentiate a pair of fault-free and faulty states after SAF activation, the fault effect at a flip-flop is propagated to the output O_{ss} by searching the input sequence on its iterative logic array of size at most $m-1$. Since $m = O(n)$, the running time of the differentiation is $O(\tau^2(n))$. For s-a-1 at the stem of SHIFT, it is obviously easy to test since during differentiation (SHIFT=1), the fault is not activated again.
Let's discuss about the faults in the fanout-free region. For each testable fault f in the fanout-free region, it is guaranteed to be detectable without fault masking during the shifting operation once the fault effect propagates to a flip-flop. Let the fault effect denoted by $\bar{d}$ that propagates to Q_h after SAF activation where $0 \leq h \leq m-1$.

$$D_{h+1} \quad = \quad (SHIFT)Q_h + F_{h+1}$$

For $SHIFT = 1$ during shifting operation,

$$D_{h+1} = \bar{d} + F_{h+1}$$

Note that no other fault effect propagates to F_{h+1} as f is in the fanout-free region and F_{h+1} has a fault-free value 0 during the shifting operation ($SHIFT = 1$). Therefore, $D_{h+1} = \bar{d}$
The fault effect is propagated to $D_{h+2}, D_{h+3}, ..., D_{m-1}$ and then output O_{ss} for m is the degree of SSFSM. For example, the output of AND gate g is in fanout-free region (Figure 5.12).
Let's discuss the faults in the fanout region other than those at the stem of Q_k for $0 \leq k \leq m-1$. During the phase of SAF activation, any input cubes contained by SHIFT are prioritized as an input pattern to excite the fault f. If one of them is a test, the fault effect that propagates to a flip-flop must be $\bar{d}$ as discussed in

the following.

$$D_i \;=\; (SHIFT) \text{ AND } Q_{i-1} \text{ OR } F_i$$

F_i has fault-free value 0 during shifting operation. In addition, shifting logic $SHIFT$ and Q_{i-1} are fault free. Therefore the fault effect propagates to F_i must be $\bar{d}$ after SAF activation if there exists an input vector that is contained by SHIFT and excite the fault f. During differentiation,

$$\begin{aligned} Q_i &= \bar{d} \\ D_{i+1} &= \bar{d} + F_{i+1} \end{aligned}$$

Since this is done by shifting operation where $SHIFT = 1$ and each stem of Q_k for $0 \leq k \leq m-1$ is fault-free, if the fault f is activated again during differentiation, it will propagate to F_{i+1} as $\bar{d}$, which has the same polarity as the fault effect during SAF activation. Therefore, fault masking does not happen during differentiation. For the faults at the stems of Q_k where $0 \leq k \leq m-1$ and let $Q_k = Q_{i-1}$, boolean difference is used to prove that the polarity of fault effect is also always same during shifting operation.

$$\begin{aligned} D_i &= (SHIFT)Q_{i-1} + F_i \\ &= (SHIFT)Q_{i-1} + R\bar{Q}_{i-1}G_i + SQ_{i-1}H_i + F_i' \end{aligned}$$

F_i' does not contain literal Q_{i-1}. G_i and H_i are the disjunctions of the cubes that contain literals Q_j for $j \neq i-1$. R and S are the disjunctions of input cubes that do not contain and are not contained by SHIFT. For Q_{i-1} s-a-0,

$$\begin{aligned} h &= Q_{i-1} \\ D_i &= (SHIFT)h + R\bar{h}G_i + ShH_i + F_i' \\ h\frac{dD_i}{dh} &= (h(SHIFT + SH_i) + hF_i') \oplus (hRG_i + hF_i') \\ &= Q_{i-1}(SHIFT) + Q_{i-1}H_iS + Q_{i-1}G_iR \text{ by redundance law} \end{aligned}$$

The similar proof can be done for the case where Q_{i-1} s-a-1. Note that the fault effect of s-a-0 (resp. s-a-1) d (resp. $\bar{d}$) propagates to D_i as d (resp. $\bar{d}$) if the vectors contained by $Q_{i-1}(SHIFT)$ or $Q_{i-1}H_iS$ is applied while propagates to D_i as $\bar{d}$ (resp. d) if the vectors contained by $Q_{i-1}G_iR$ is applied. During SAF activation, the input cubes contained by SHIFT is prioritized. Therefore, if there exists an input pattern contained by SHIFT, the fault effect that propagates to a flip-flop during SAF activation is d, i.e. $D_i = d$. Let the fault effect propagate to Q_{i-1} after SAF activation of a fault f in the fanout region and let the input cube that activates the fault denoted by X_c. During the phase of differentiation, the fault effect being shifted is not masked because only the fault effect of the same polarity may appear at F_j for $j \geq i$ under input sequence consisting of X_c. Therefore, by giving the input cubes contained by SHIFT a priority to be an input vector to excite the fault, fault masking does not happen during state differentiation. In the case where no input vectors contained by either SHIFT excite the fault f, the fault is not excited during the phase of differentiation and thus the fault masking does not occur. Let T_E, T_J and T_D denote the running time of SAF excitation, justification and differentiation. Therefore,

$$\begin{aligned}
T^{SA}_{2CD}(n) &= T_E(n) + T_J + T_D \\
&= \tau(n) + O(m-1) + O(\tau((m-1)n)) \\
&= \tau(n) + O(n) + O(\tau^2(n)) \text{ for } m = O(n) \\
&= O(\tau^2(n))
\end{aligned}$$

□

However, the work cannot conclude that the SAF test generation complexity of the two-column distributive SSFSM realizations is not τ-equivalent although it seems to be correct.

Conjecture 5.5.1. *The SAF test generation of the two-column distributive SS-FSM realizations is not τ-equivalent.*

The PDF test generation complexity of the two-column distributive SS-FSM realizations under rated clock is still an open problem.

Open Problem 6. Is the PDF test generation of the two-column distributive SSFSM realizations τ-equivalent under rated clock?

5.6. Conclusion

The relationships between the PDF test generation and SAF test generation of several existing classes of circuits have been described in this chapter. The PDF test generation of internally balanced sequential circuits under rated clock and slow-fast-slow clock is equivalent to the SAF test generation of combinational circuits. On the other hand, the PDF test generation of the acyclic sequential circuits are τ^2-bounded under slow-fast-slow clock. It is shown that under TEM with slow-fast-slow clock the PDF test generation is not τ-equivalent. The PDF test generation under slow-fast-slow clock and the SAF test generation of two-column SSFSM realizations with observable shifting logic are equivalent to the SAF test generation of the combinational circuits while for two-column distributive SSFSM realizations, its PDF test generation under slow-fast-slow clock is τ-equivalent but its SAF test generation is τ^2-bounded. The PDF test generation of acyclic sequential circuits and cyclic sequential circuits is discussed under the assumption of slow-fast-slow clock. The discussion of PDF test generation of sequential circuits under rated clock remains an open problem. The conjectures and the open problems are summarized as follows. Note that slow-fast-slow clcok and rated clock are simplified as sfs and $rated$, respectively.
Conjecture 5.4.2 $T_A^{PD,sfs}(n) \neq \tau(n)$
Open problem 1 $T_A^{PD,rated}(n) = O(\tau^2(n))$?
Open problem 2 $T_{A,TEM}^{PD,rated}(n) \neq \tau(n)$?
Open problem 3 $T_A^{PD,rated}(n) \neq \tau(n)$?
Open problem 4 $T_A^{PD,rated}(n) = T_A^{SA}(n)$?
Open problem 5 $T_{2COS}^{PD,rated}(n) = T_{2COS}^{SA}(n) = T_C^{sSA} = \tau(n)$?
Conjecture 5.5.1 $T_{2CD}^{SA}(n) \neq \tau(n)$
Open problem 6 $T_{2CD}^{PD,rated}(n) = \tau(n)$?

Chapter 6

Classification of Sequential Circuits Based on Acyclic Test Generation Complexity

6.1. Introduction

The test generation of acyclic sequential circuits has been shown to be τ^2-bounded [11,12] using time expansion model (TEM) [18]. In other words, the test generation complexity is at most the square of combinational test generation complexity, which is regarded as not difficult. This chapter introduces a new class of sequential circuits with acyclic test generation complexity. The new class is called acyclically testable sequential circuits whose test generation complexity of the new class is bounded by a circuit property called thru function. [33] has introduced a class of circuits called partially strong testable circuits based on thru function but the target circuit is datapath only and test generation complexity was not discussed explicitly. [34] also considered existing thru functions in a scan technique but those thru functions are activated by primary inputs only. The new class that is defined in this chapter covers some sequential circuits that are cyclic. In an acyclically testable sequential circuit, the signals that activate a thru function are either the signals at primary inputs or the signals at registers. Based on the properties of input dependency, thru tree dependency and the depth of thru trees, two subclasses of acyclically testable sequential circuits, namely acyclically testable sequential circuits type A and acyclically testable sequential circuits type B [28], are identified. By introducing stronger conditions, a subclass of acyclically testable sequential circuits type B is defined[29]. It is named acyclically testable sequential circuits type C. This chapter also introduces a test generation procedure and an analysis of the test generation complexity of acyclically testable sequential circuits. This is followed by a design-for-testability (DFT) method to augment an arbitrary sequential circuit into an acyclically testable sequential circuit. An experiment on benchmark circuits is conducted to show the effectiveness of the DFT method. Finally, the chapter is concluded.

6.2. Acyclically Testable Sequential Circuits

This section defines a circuit representation called R-graph. Using R-graph, new concepts of circuit properties are introduced. These circuit properties include thru function, thru tree, thru tree dependency, input dependency and k-consistency. Based on these properties, the class of acyclically testable sequential circuits whose test generation is equivalent to the test generation of acyclic sequential circuits is defined. The relationship between the class of acyclically testable sequential circuits and acyclic sequential circuits are shown in Figure 6.1. Furthermore, three classes of sequential circuits are categorized as the subclasses of acyclically testable sequential circuits based on varying circuit properties.

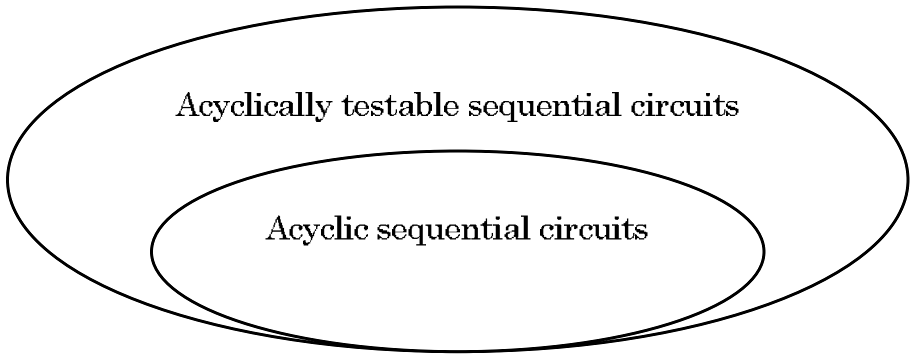

Acyclically testable sequential circuits
Acyclic sequential circuits

Figure 6.1. The relationship between acyclic sequential circuits and acyclically testable sequential circuits.

Definition 6.2.1. Let X, Y and Z be a set of boolean variables respectively in a circuit where $X \cap Y \cap Z = \emptyset$. A ***thru function*** $t_{X \to Y}$ is a boolean formula in conjunctive normal form such that

- the boolean connectives of the formula consist of $\wedge$ (AND), $\vee$ (OR) and $\neg$ (NOT);
- the boolean variables Z of the formula and X consist of register outputs and primary inputs while Y consists of register inputs and primary outputs;
- the signals at X transfer to Y if Z has an assignment that makes the thru function 'true' or active ($t_{X \to Y} = 1$);

Example 6.2.1. Figure 6.2(a) shows that without depending on the signals at the output X of feedback register r, Y can be justified by only U with thru function $t_{U \to Y}$ active. Figure 6.2(b) presents another example circuit with a multiplexer MUX. Signals at I transfer to L when $K = 0$.

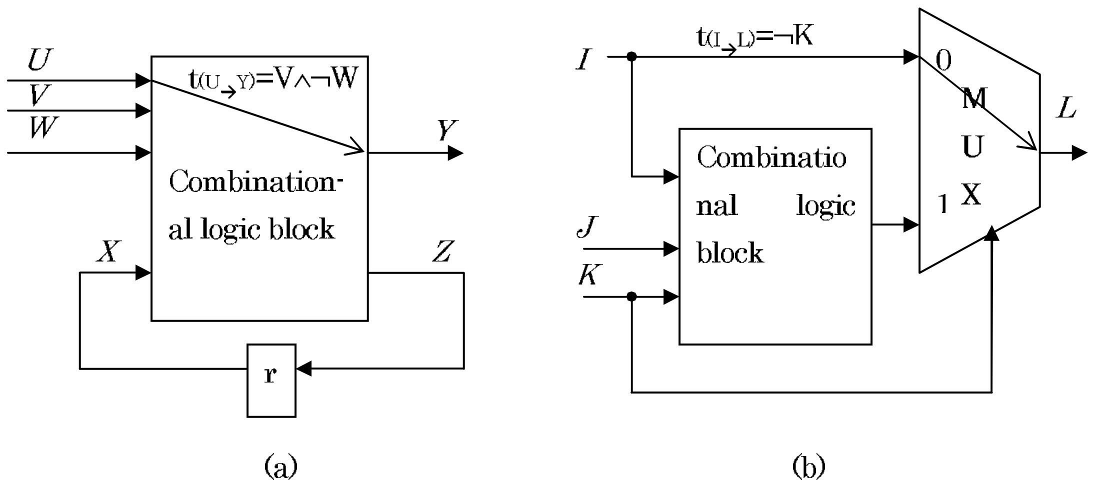


Figure 6.2. The use of thru functions

Definition 6.2.2. Two thru functions $t_{i \to j}$ and $t_{l \to m}$ are said to be ***dependent*** if they cannot be active at the same time.

Example 6.2.2. Figure 6.3 shows two functions $t_{I1 \rightarrow O1}$ and $t_{I3 \rightarrow O1}$ that are not dependent. In other words, thru functions t_1 can be active at the same time. Figure 6.4(a) shows two functions $t_{I1 \rightarrow O1}$ and $t_{I3 \rightarrow O1}$ that are dependent because signals at $I1$ and $I3$ do not transfer to $O1$ simultaneously. Figure 6.4(b) illustrates another example circuit that consists of three multiplexers. Thru function $t_{i \rightarrow o} = \neg p \wedge \neg q$ and thru funtion $t_{k \rightarrow o} = \neg p \wedge q$ are dependent as shown by the boolean formula in each thru function.

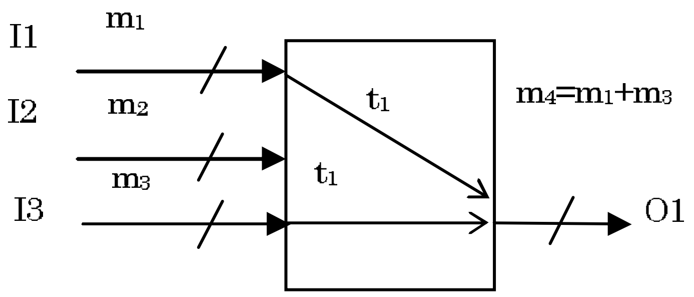


Figure 6.3. Not dependent thru functions.

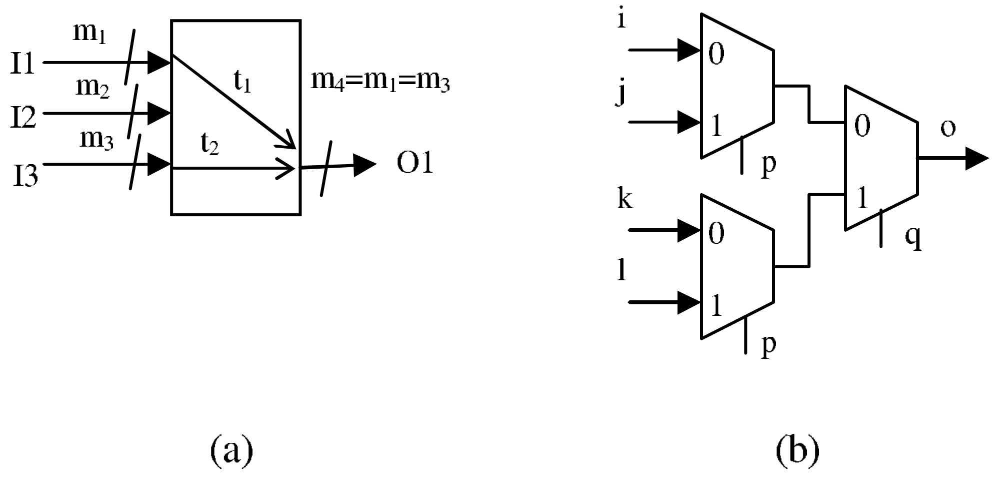


Figure 6.4. Dependent thru functions.

R-graph represents the topology of circuits by grouping flip-flops (FFs) into registers and including the information about the thru functions available in the logic. R-graph is used to introduce the new concepts of the circuit properties including thru function, thru tree, thru tree dependency and input dependency.

Definition 6.2.3. A circuit representation called ***R-graph*** is a directed graph $G = (V, A, w, r, t)$ that has the following properties.

1. Let FF_i denote a flip-flop. Let $pre(FF_i) = \{FF_j | FF_j \xrightarrow{c} FF_i\}$ (resp. $suc(FF_i) = \{FF_j | FF_i \xrightarrow{c} FF_j\}$) where c is a combinational path. $v \in V$ is a primary input or primary output or register that consists of a maximal set of flip-flops such that $pre(FF_p) = pre(FF_q)$ and $suc(FF_p) = suc(FF_q)$ for all FF_p, FF_q in the set of flip-flops;

2. $(v_i, v_j) \in A$ denotes an arc if there exists a combinational path from the register corresponding to v_i to the register corresponding to v_j;

3. $w : V \rightarrow Z^+$ (the set of positive integers) defines the number of flip-flops in each register corresponding to a vertex;

4. $r : V \rightarrow \{h, \emptyset\}$ defines type of a register where the register is a hold register v if $r(v) = h$. Else, it is a regular register. Note that $r(w) = \emptyset$ if w corresponds to a primary input or primary output;

5. $t : A \rightarrow T \bigcup \{\emptyset, 1\}$ (T is a set of thru functions) where $t(u, v) = \emptyset$ if there is no thru function for $(u, v) \in A$ and $t(u, v)$ is a thru function that transfer signals from the output of register u or primary input u to the input of register v or primary output v. If $t(u, v) = 1$ (also called identity thru function), the signal values are transferred from u to v through a wire logic (not a gate logic) directly. Note that identity thru function is always active.

Example 6.2.3. Figure 6.6 shows the R-graph of the sequential circuit S1 of Figure 6.5. The notation CLB in Figure 6.5 means combinational logic block that include the information of logic connection in the block. Black registers are registers with hold functions while others are regular registers. Register $R2$ is a hold register. The thru functions $t_{(I \rightarrow K)}$, $t_{(L \rightarrow N)}$, $t_{(O \rightarrow P)}$ and $t_{(Q \rightarrow S)}$ which are the thru functions extracted from the high level netlist of S1, are included in its R-graph. According to the R-graph, $R1$, $R2$ and $R3$ form a loop while $R5$ forms a self-loop.

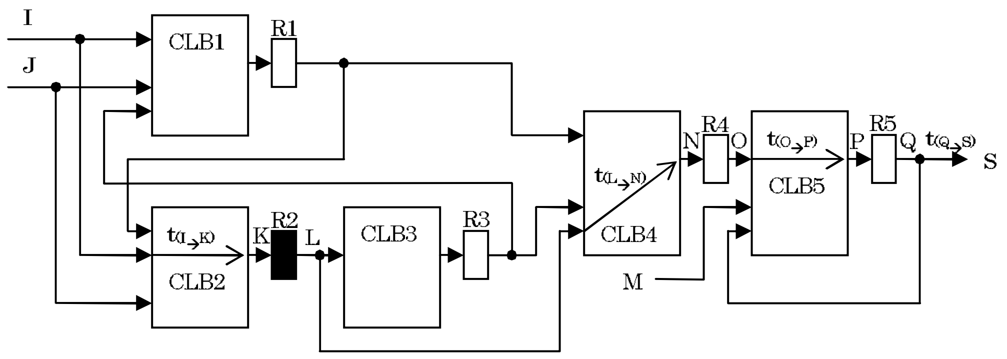


Figure 6.5. Sequential circuit $S1$.

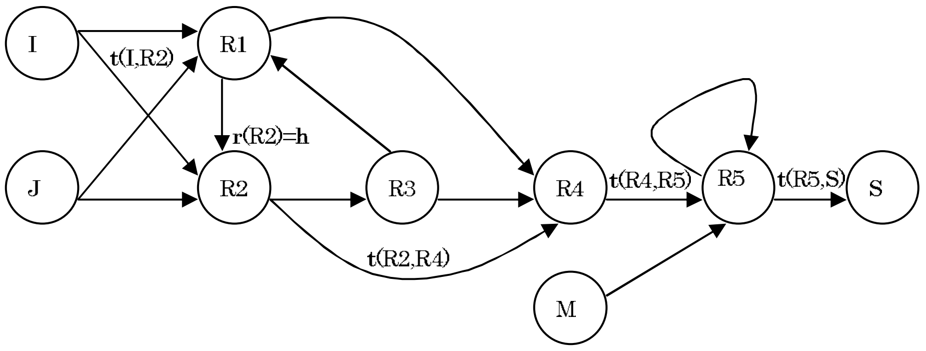


Figure 6.6. R-graph of $S1$.

If a thru function transfers signal from a register outside a loop or a primary input to a register inside the loop, the thru function can be used to justify the register in a loop with a signal from the register or the primary input outside the loop within one clock cycle but without depending on any signal in the loop. In other words, the loop is broken logically by the thru function. However, a thru function is not sufficient to guarantee that the register in the loop can be justified with any signal within the signal range of normal operation. The following shows an example where a thru function cannot justify a signal to a register in a loop.

Example 6.2.4. Let E (resp. G) denote a 4-bit variables consiting of bits e_3, e_2, e_1 and e_0(resp. g_3, g_2, g_1 and g_0). Figure 6.7 shows a sequential circuit that has two 4-bit adder where $C = A + B$ and $F = D + I$ and a stuck-at-1 fault at i_3. There is a thru function $t_{E\to G} = \neg SEL$ that transfers signals from E to G

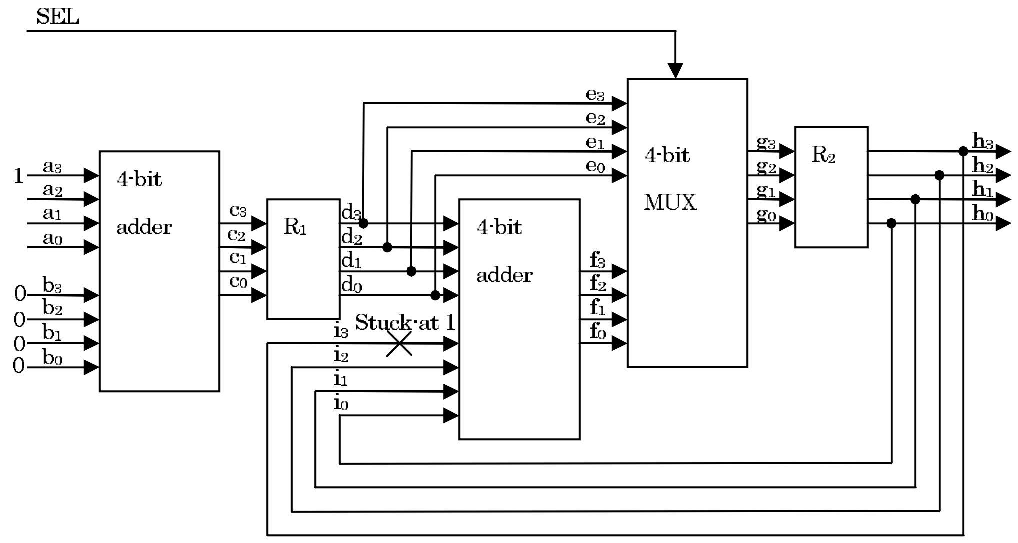


Figure 6.7. The limitation of thru function in justification.

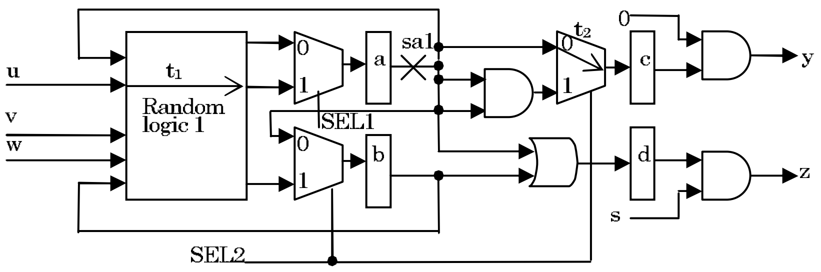


Figure 6.8. The limitation of thru function in propagation.

when $SEL = 0$ within one clock cycle without depending on the signals at the output of feedback register R_2 at I. However, if the range of the signals that are justifiable by the thru function is studied carefully, it is obvious that the thru function cannot justify some signals that are in the range of the signals at G in normal operation. In this example, the signal range at E in normal operation is from 8 to 15. If the thru function is used to justify G, the justifiable signal range at G is from 8 to 15. Thus, the stuck-at-1 cannot be activated. However, if the following steps are done, the stuck-at-1 can be activated after four clock cycles.

- First, D is assigned 8 and then tranfered to G through the thru function with $SEL = 0$;

- At the next clock cycle, D is assigned 9 with $SEL = 1$. Note that the signal G is now 1;
- After one clock pulse at R_2, I is justified with 1 and the stuck-at fault is activated.

Thru function does not guarantee the justification of every signal within the range of normal operation. Similarly, thru function does not guarantee the propagation of every signal within the range of normal operation.

Example 6.2.5. Figure 6.8 shows a circuit where the stuck-at-1 fault (sa1) can be activated by assigning signal 0 from primary input u through thru function t_1 with $SEL1 = 1$. When the fault effect propagates through thru function t_2 to the output of flip-flop c after one clock cycle, the fault effect disappears as it is masked by the constraint 0 at the input of AND gate y. However, if the fault effect first propagates to the output of flip-flop b, the fault effect can be observed at z without depending on any thru function.

Therefore, a concept called thru tree is introduced in this chapter. Thru tree consists of a set of thru functions connected in a form of rooted tree that starts from primary inputs and ends at a primary output.

Definition 6.2.4. Let R-graph $G_R = (V, A, w, r, t)$ represent a given sequential circuit S. A ***thru tree*** is a subgraph of the R-graph such that

1. it is a rooted tree;
2. there is only one sink (root), which is corresponding to a primary output;
3. the sources are vertices that correspond to primary inputs;
4. each arc is labeled with a thru function.

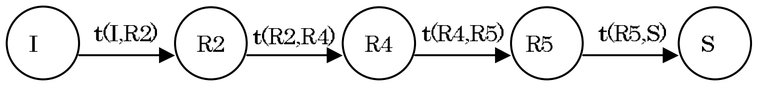


Figure 6.9. The only thru tree of S1.

Figure 6.9 shows the only thru tree of S1, which is also a path whose arcs are labelled with thru functions. In the thru tree, each register is justifiable from a primary input and is observable at a primary output through a series of thru functions if each thru function in the tree is activated by a signal whose corresponding vertex is not in the thru tree. If a thru function in a thru tree is activated by a signal of the vertex which is also in the same thru tree, the thru function does not guarantee the justification and propagation. The following shows an example where a contradiction takes place.

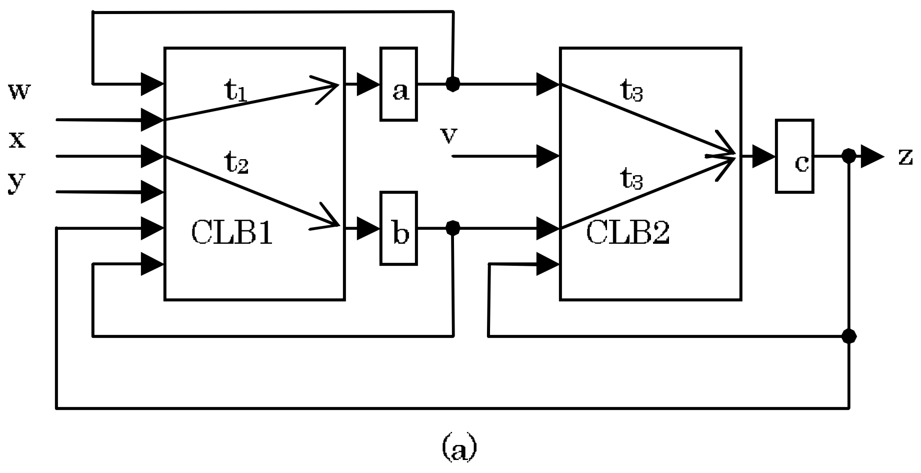
Example 6.2.6. Figure 6.10 shows another example circuit S2 and its R-graph. Figure 6.11 shows a thru tree of S2. Let $t_1 = y$ and $t_2 = c$ and $t_3 = \neg v$. In order to justify register c using the thru tree, all the thru functions t_1, t_2 and t_3 must be active. However, t_2 depends on c to become active while c is depending on thru function t_2 for justification from primary inputs x and w without depending on feedback a and c. The interdependency between justifying register c and activating thru function t_2 occurs. Therefore, justification of register c is not guaranteed.

To avoid the interdependency, it is essential to define the concept of the dependency between a thru tree and the register that activates a thru function in the thru tree as well as the dependency between two thru trees. Then, the concept is based in the definition of acyclically testable sequential circuits.

Definition 6.2.5. If V_{ti} is a set of vertices that activate a thru function t_i in a thru tree T_j, T_j is said to be ***dependent*** on V_{ti}. Furthermore, if V_{ti} includes a vertex in a thru tree T_k, T_j is said to be ***dependent*** on T_k.

Example 6.2.7. Figure 6.12 are a sequential circuit S3, its R-graph and its thru trees $T1$ and $T2$. From its R-graph and thru trees, $T1$ is dependent on $I3$ while $T2$ is dependent on $R2$ and $I1$. Furthermore, $T2$ is dependent on $T1$ since $R2$ and $I1$ are included in $T2$.

Another issue to be discussed here is input dependency. While a thru function $t_{x \to y}$ is being used to justify y, x is fixed at the signals that are needed to justify y. If x is needed to justify y and another signal, for example z, at the same time, z may not be justified since x is fixed to justify y through the thru function. Again, justification is not guaranteed.



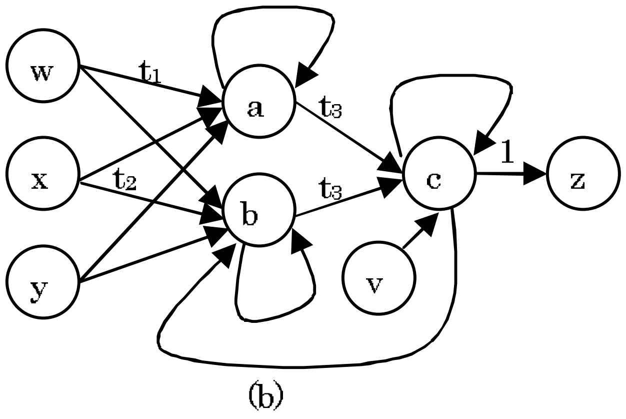


Figure 6.10. S2 (a) and its R-graph (b).

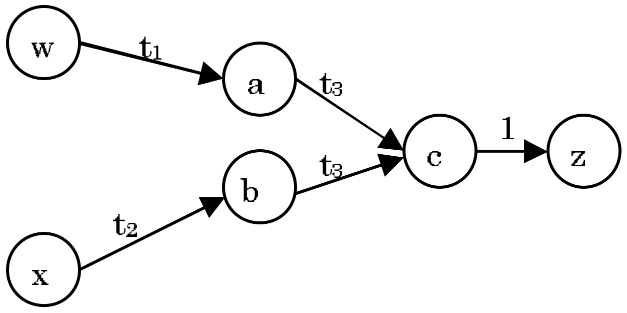
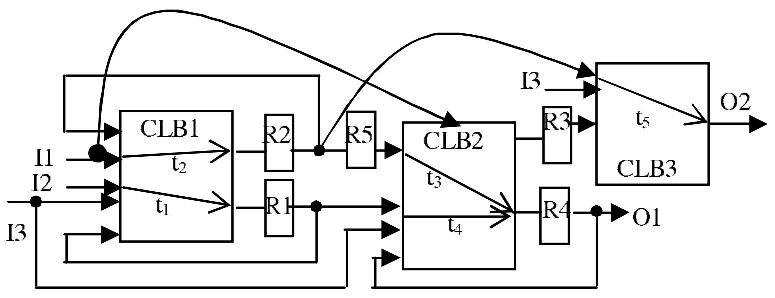
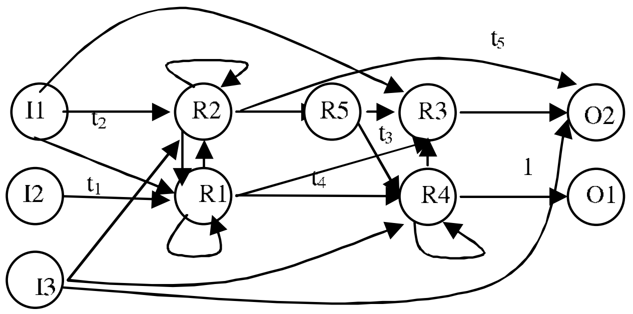


Figure 6.11. A thru tree of S2.

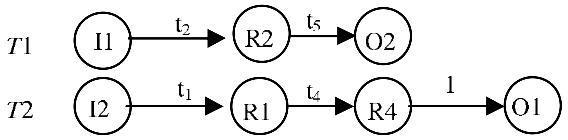


(a) Sequential circuit S3



t_1= (R2) (I1); t_2= I3; t_3= I3; t_4= (R2) (I1); t_5= I3
r(R2)=h, r(R1)=h, r(R4)=h

(b) R-graph of S3



(c) Thru trees *T*1 and *T*2

Figure 6.12. Sequential circuit S3.

Definition 6.2.6. Let G_R be the R-graph of a sequential circuit S, and let B be a set of thru trees in G_R. Let (u, v) be a set of all paths starting at u and ending at v. Two distinct paths $p_1, p_2 \in (u, v)$ have ***input dependency*** if the following conditions are satisfied.

i. the first arc of one of the paths is different from the first arc of another path;

ii. the first arc of at least one of the paths is labeled with a thru function in a thru tree in B;

iii. each path contains at most one cycle that starts from and ends at v;

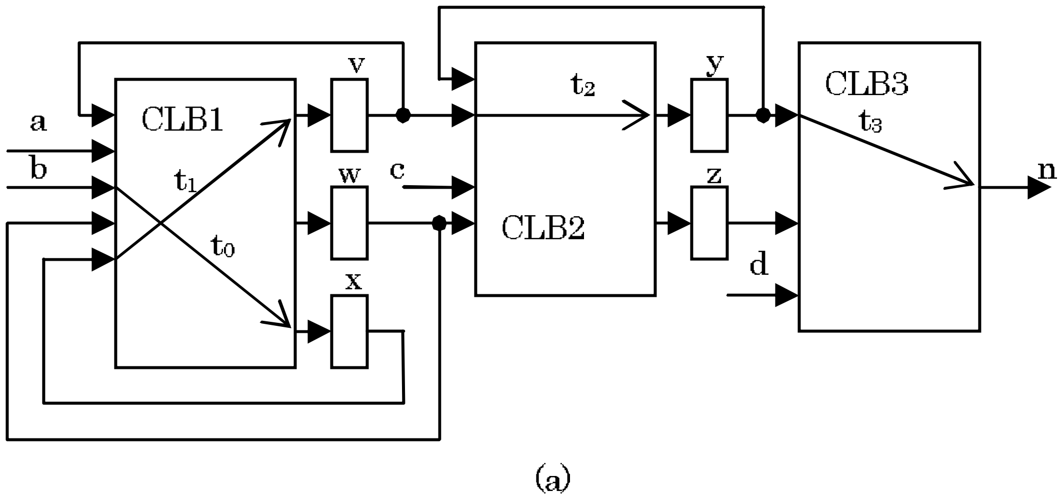
iv. if the first arc of a path p_1 (resp. p_2) does not have a thru function in a thru tree in B, all vertices except the first vertex and the last vertex are not included in any thru tree in B. Else if the first arc of a path p_1 (resp. p_2) has a thru function in a thru tree in B, all vertices except the first vertex, the second vertex and the last vertex are not included in any thru tree in B;

v. p_1 and p_2 have same length;

Example 6.2.8. Figure 6.13 shows an example circuit $S4$ with thru functions t_0, t_1, t_2 and t_3 and its R-graph. Figure 6.14 shows two of the paths from R-graph where the first arc of path $x \to v \to v$ is labeled with t_1 and both paths are of same length. Suppose v and w have to be justified 1 respectively in order to excite a fault in CLB2. To justify v, x has to be assigned 1 one clock cycle before. However, if x needs to be 0 in order to generate 1 at w using the whole logic in CLB1, a conflict takes place. Thus, the fault cannot be excited using thru functions t_0 and t_1 for justification of v.

Input dependency can be resolved by hold registers with certain conditions. The following shows one example how a hold register resolves an input dependency.

Example 6.2.9. Circuit $S5$ in Figure 6.15 has an input dependency between two paths, $R1 \to R4 \to PO$ and $R1 \to R5 \to PO$. The two paths with input



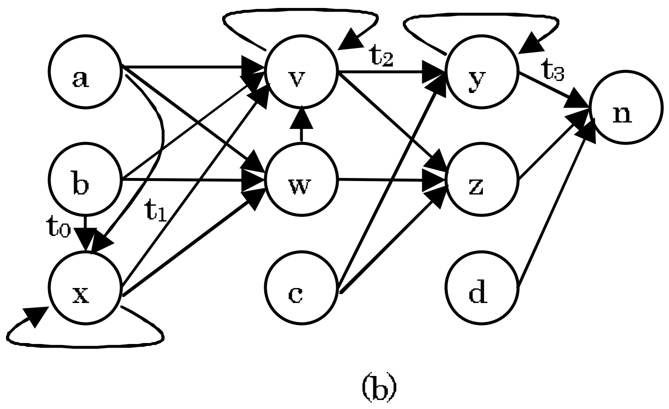


Figure 6.13. S4 (a) and its R-graph (b).

dependency is shown in the aspects of graph and time expansion model in Figure 6.16. Circuit $S6$ in Figure 6.17 is same as $S5$ except register $R5$ of $S6$ is a hold register. By holding $R5$ at time $T2$, input dependency between paths $R1 \rightarrow R4 \rightarrow PO$ and $R1 \rightarrow R5 \rightarrow PO$ can be resolved. This is illustrated in Figure 6.18.

Thru tree dependency has to be resolved in test generation process. Thru tree dependency takes place when the signal of a register is used to activate a thru function and justify another register simultaneously.

Definition 6.2.7. Let G_R be the R-graph of a sequential circuit S, and let B be

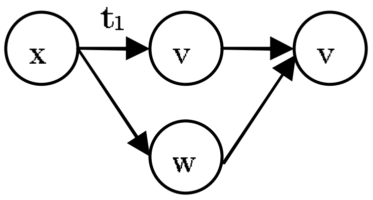


Figure 6.14. Two paths that have input dependency.

a set of thru trees in G_R and let $a_{ti} \in G_R$ be an arc with thru function t_i. For each pair of paths p_m and p_n, p_m and p_n have ***thru tree dependency*** if

i. p_m is a path that starts from the sink vertex u of arc a_{ti} and ends at a vertex v;

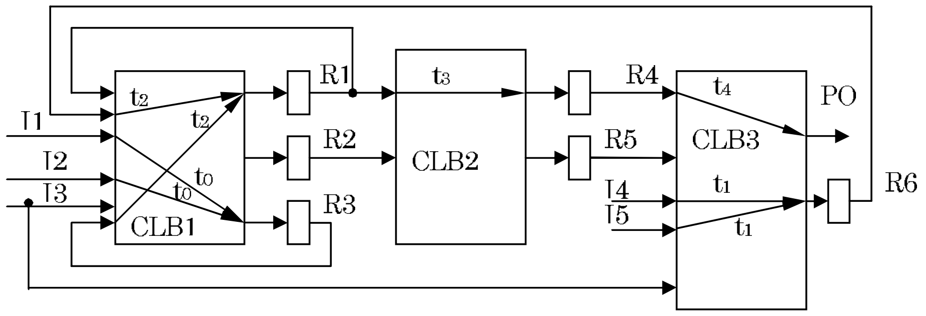
ii. p_n is a path that starts from a vertex w in V_{ti} and ends at v;

iii. p_m and p_n are the paths where each path is either a simple path or a path that contains a cycle starting from and ending at v, and each vertex, except the first vertex and the last vertex, is not included in any thru tree in B;

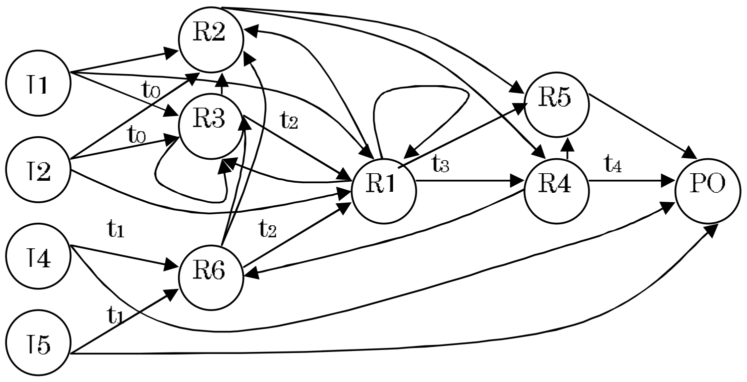
iv. $|p_m| < |p_n|$.

Example 6.2.10. Figure 6.19 is a sequential circuit with hold registers u, w, x and z. Note that signal at register z at time 4 is used to justify $CLB1$ and at the same time activate thru function t_0 at time 4 in Figure 6.20. This is because there is a thru tree dependency between paths $x \rightarrow x$ and $z \rightarrow u \rightarrow x$.

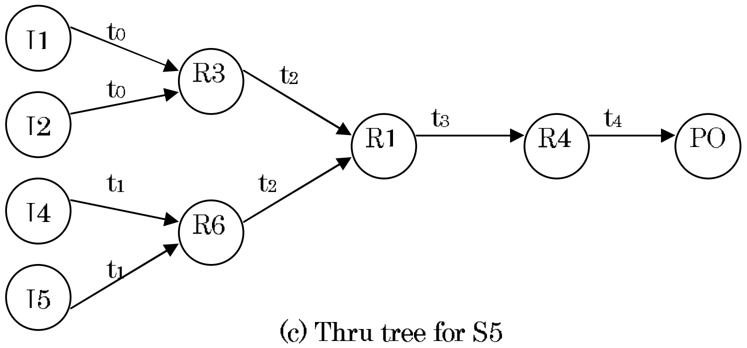
Thru tree dependency is also resolvable by hold registers.



(a) S5

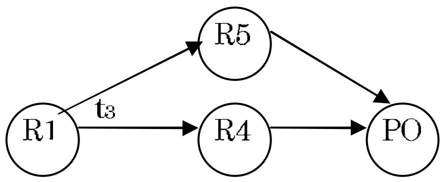


(b) R-graph for S5

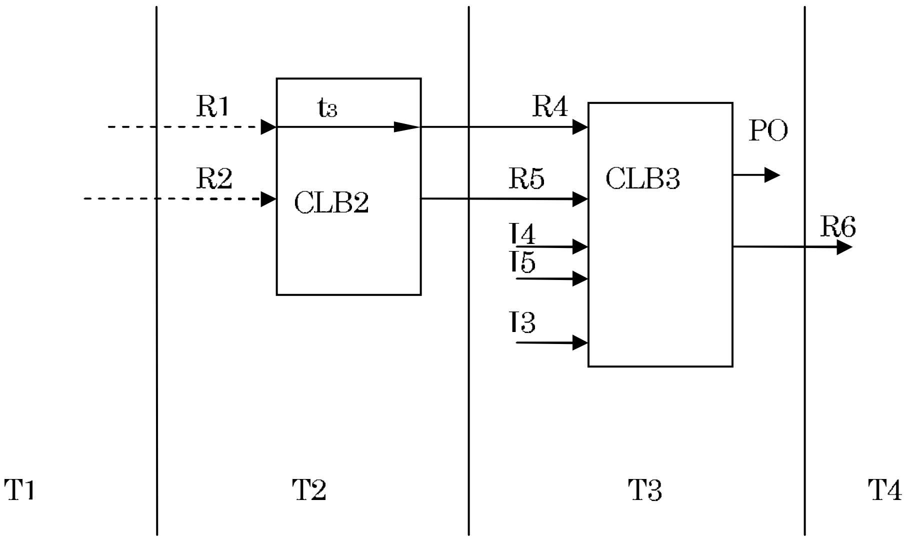


(c) Thru tree for S5

Figure 6.15. Sequential circuit S5.

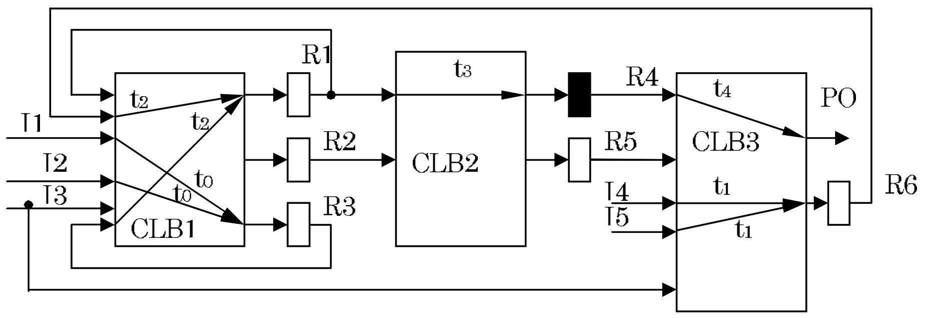


(a) Input dependency

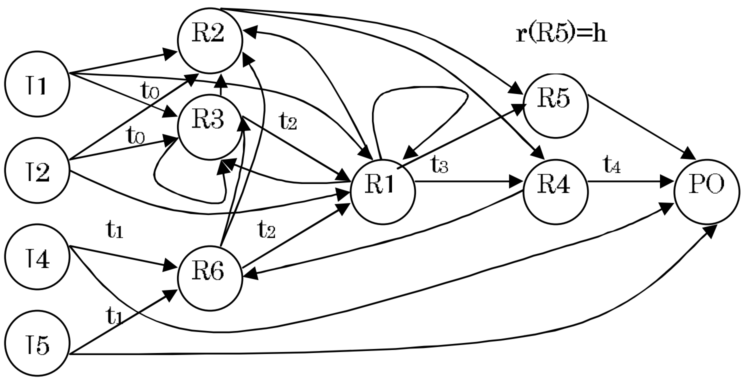


(b) Input dependency between path R1→R4→PO and path R1→R5→PO.

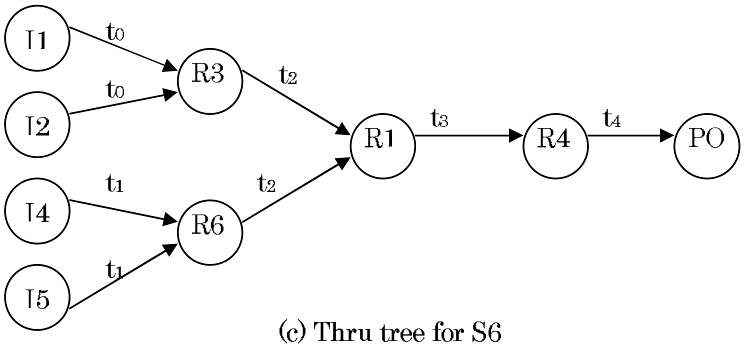
Figure 6.16. Input dependency in S5.



(a) S6

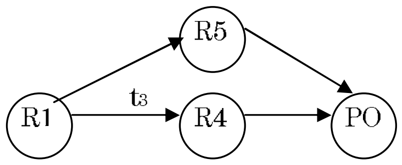


(b) R-graph for S6

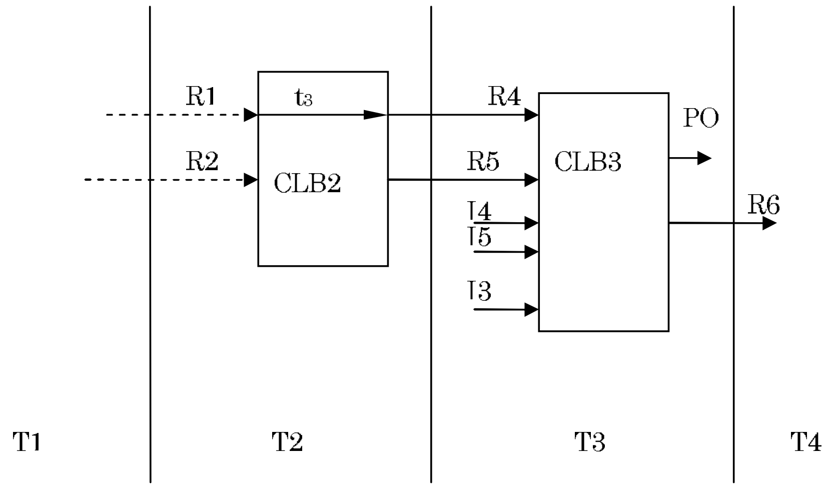


(c) Thru tree for S6

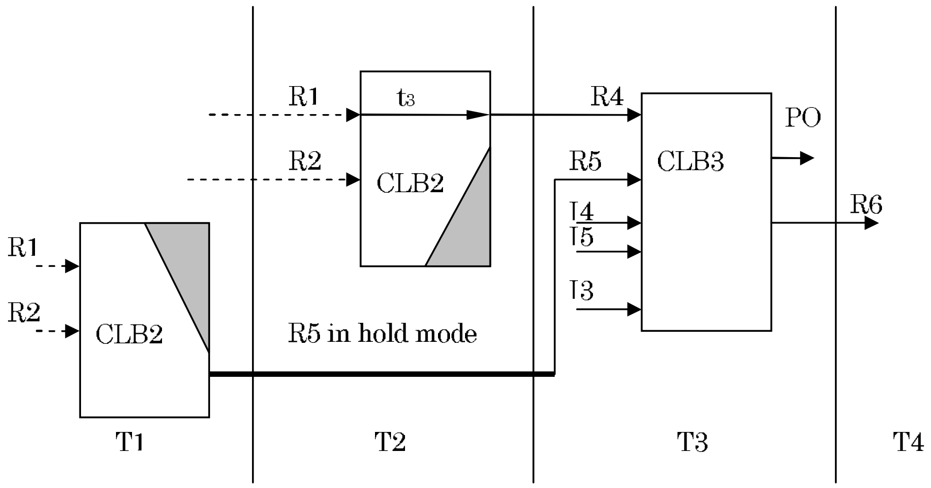
Figure 6.17. Sequential circuit S6 with hold register R5.



(a) Input dependency

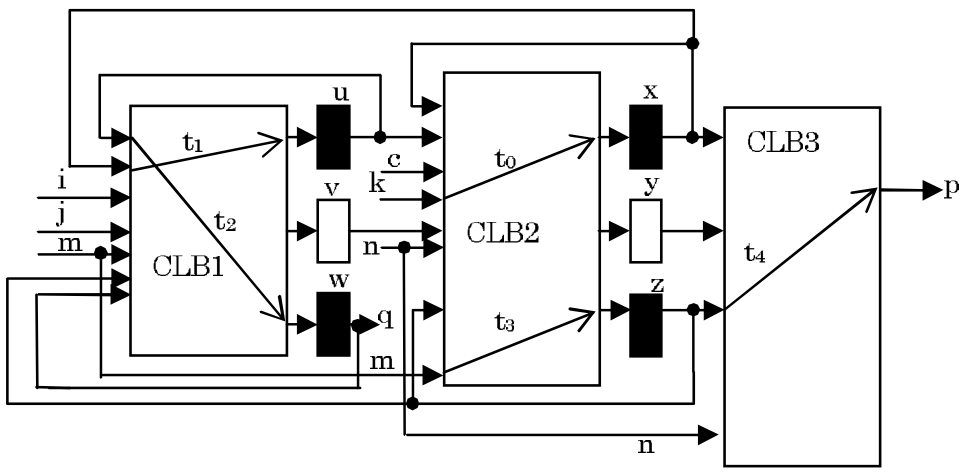


(b) Input dependency between path R1→R4→PO and path R1→R5→PO.

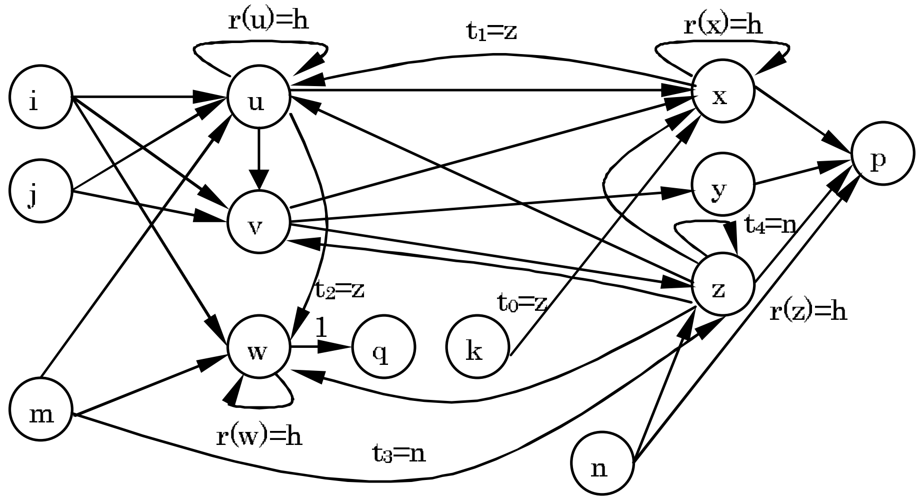


(c) Input dependency between path R1→R4→PO and path R1→R5→PO is resoved by holding R5.

Figure 6.18. Resolution of the Input dependency in S6.



(a) S7



(b)R-graph for S7

$$\begin{tikzcd}
c \arrow[r, "t_0=z"] & x \arrow[r, "t_1=z"] & u \arrow[r, "t_2=z"] & w \arrow[r, "1"] & q
\end{tikzcd}$$

$$\begin{tikzcd}
d \arrow[r, "t_3=n"] & z \arrow[r, "t_4=n"] & p
\end{tikzcd}$$

(c) Thru trees for S7

Figure 6.19. S7 (a) and its R-graph (b).

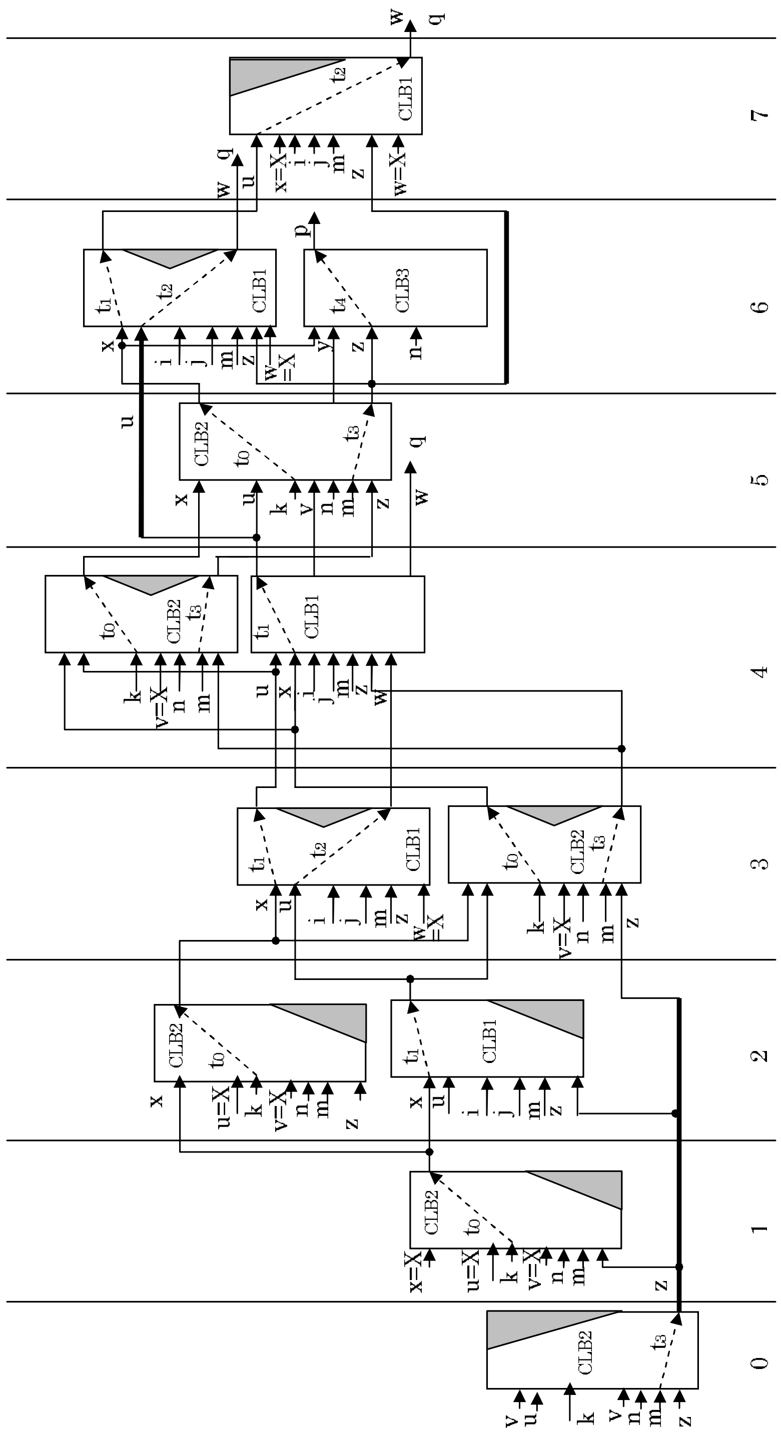


Figure 6.20. Time expansion model of S7.

The number of thru trees that depend on each other in an acyclically testable sequential circuit is one of the factors that bound its test generation complexity to τ^2-bounded. Therefore, a dependency graph is introduced to represent the property of the dependency and the number of thru trees that depend on each other.

Definition 6.2.8. Let G_R be the R-graph of a sequential circuit S, and let B be a set of thru trees in G_R. The ***dependency graph*** of B is a directed graph $G_D = (V_D, A_D)$ such that

i. vertex $v \in V_D$ is a thru tree in B;

ii. $(v_i, v_j) \in A_D$ denotes an arc if there exists a vertex (of G_R) in thru tree v_i that activates a thru function in thru tree v_j;

Example 6.2.11. Figure 6.21 shows the dependency graph of $T1$ and $T2$ of S3.

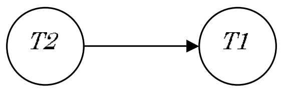


Figure 6.21. Dependency graph of $T1$ and $T2$ of S3.

Based on the concepts of thru function, thru tree, thru tree dependency and input dependency, three classes of acyclically testable sequential circuits are detailed in the following subsections.

6.2.1 Acyclically Testable Sequential Circuits Type A

Acyclically testable sequential circuits type A is a class of acyclically testable sequential circuits that does not have input dependency as well as thru tree dependency [28].

Definition 6.2.9. Let R-graph $G_R = (V, A, w, r, t)$ represent a given sequential circuit S. S is called to be ***acyclically testable*** if G_R contains a set of disjoint thru trees such that the following conditions are satisfied.

1. There exists a set TT of thru trees that cover all the vertices of a feedback vertex set;

2. Let V_{ti} be a set of all vertices that activate a thru function t_i. For any thru function t_i in each thru tree T_j in TT, the following conditions are satisfied.

   i. V_{ti} does not include any vertex of T_j;

   ii. V_{ti} does not include any register vertex that is not included in any thru tree. Note that V_{ti} can include an input vertex or output vertex that is not included in any thru tree;

   iii. All the register vertices in T_j and V_{ti} are hold registers;

3. Let T_i and T_j be two different trees in TT.

   i. For each pair of thru function t_i in T_i and t_j in T_j, V_{ti} and V_{tj} are disjoint;

   ii. If T_i (resp. T_j) is dependent on T_j (resp. T_i), T_j (resp. T_i) is not dependent on T_i (resp. T_j), and

   iii. If T_i is dependent on T_j, $|p_i| \geq |p_j|$ for each pair of p_i and p_j such that p_i (resp. p_j) is a path starting from vertex u_i in T_i (resp. vertex u_j in T_j) and ending at vertex v with at most one cycle starting from and ending at v, where $|p_i|$ (resp. $|p_j|$) denotes the length of path p_i (resp. p_j);

4. For each pair of reconvergent paths p_1 and p_2, p_1 and p_2 does not have input dependency;

5. For each pair of paths p_1 and p_2, p_1 and p_2 does not have thru tree dependency.

6.2.2 Acyclically Testable Sequential Circuits Type B

Different from acyclically testable sequential circuits type A, acyclically testable sequential circuits type B are allowed to have input dependency and thru tree dependency with conditions that there exists a set of hold registers that can resolve the input dependency and thru tree dependency [28]. As a tradeoff between the input dependency (resp. thru tree dependency) and test generation complexity, the multiplication of the maximum length of paths in the dependency graph and the maximum depth of thru trees in the sequential circuit is to be bounded by a constant.

Definition 6.2.10. Let R-graph $G_R = (V, A, w, r, t)$ represent a given sequential circuit S. S is called to be ***acyclically testable*** if G_R contains a set of disjoint thru trees B such that the following conditions are satisfied.

1. B is a set of thru trees that satisfies the following conditions.
   - i. The thru trees in B cover all the vertices of a feedback vertex set; and
   - ii Let the maximum depth of thru trees in B be D_{max}. Let the maximum length of paths in the dependency graph of B be L_{max}. $D_{max} \times L_{max}$ is bounded by $O(1)$;
2. Let V_{ti} be a set of all vertices that activate a thru function t_i. Let $a_{ti} \in A$ be an arc with thru function t_i. For any thru function t_i in each thru tree T_j in B, the following conditions are satisfied.
   - i. V_{ti} does not include any vertex of T_j;
   - ii. V_{ti} does not include any register vertex that is not included in any thru tree in B. Note that V_{ti} can include an input vertex or output vertex that is not included in any thru tree in B;
   - iii. The sink vertex of arc a_{ti} in T_j and the register vertices in V_{ti} are corresponding to hold registers;
   - iv. For each pair of paths p_m and p_n, if p_m and p_n have thru tree dependency then there exists a hold register vertex x ($r(x) = h$) that satisfies either Condition A or Condition B.
     - (A) i. x is on p_m but not p_n, and $x \neq u$; and
       - ii. Let p_k be a path that starts from x and ends at v. Let p_p be the subpath of p_m that starts from x and ends at v. $|p_p| \geq |p_k|$ for all p_k.

(B) i. x is on p_n but not p_m, and $x \neq w$; and

ii. Let p_k be a path that starts from x and ends at v. Let p_p denote the subpath of p_n that starts from x and ends at v. $|p_p| \geq |p_k|$ for all p_k.

$|p_m|$ (resp. $|p_n|$, $|p_k|$) denotes the length of path p_m (resp. p_n, p_k);

3. Let T_i and T_j be two different trees in B.

   i. For each pair of thru function t_i in T_i and t_j in T_j, V_{ti} and V_{tj} are disjoint;

   ii. If T_i (resp. T_j) is dependent on T_j (resp. T_i), T_j (resp. T_i) is not dependent on T_i (resp. T_j), and

4. For each pair of reconvergent paths p_1 and p_2 that start from u and end at v, there exists a hold register vertex w on p_1 such that the length of the subpath of p_1 that starts from w and ends at v is equal or longer than the length of p_k for all p_k if p_1 and p_2 have input dependency where p_k denotes a path that starts from w and ends at v.

6.2.3 Acyclically Testable Sequential Circuits Type C

In [29], a subclass of acyclically testable sequential circuits type B is introduced. This class is called acyclically testable sequential circuits type C, which has stronger conditions. A property called k-consistency is introduced. Then, the class of acyclically testable sequential circuits type C is defined based on k-consistency.

Definition 6.2.11. Let R-graph $G_R = (V, A, w, r, t)$ represent a given sequential circuit S. A set of thru tree B in G_R is said to be ***k-consistent*** with G_R if the following conditions are satisfied.

i. The dependency graph of B is acyclic;

ii. All thru trees in B are disjoint;

iii. Let the maximum depth of thru trees in B be D_{max}. Let the maximum length of paths in the dependency graph of B be L_{max}. $D_{max} \times L_{max}$ is bounded by k;

iv. Any vertex that activates a thru tree T_i in B is either an input vertex or a hold register vertex in B, and activates no other thru tree T_j in B;

v. For each pair of reconvergent paths p_1 and p_2 that start from u and end at v, there exists a hold register vertex w on p_1 but not on p_2 such that w is not the second vertex x of p_1 and the length of the subpath $w \rightarrow v$ of p_1 is equal to or longer than the length of any other path p_k that starts from w and ends at v if all vertices on p_1 and p_2 except u, v and x are not included in any thru tree in B and either of the following Conditions a and b is satisfied.

   a. p_1 and p_2 are of equal length and the first arc (u, x) on p_1 is labeled with a thru function of a thru tree in B; or

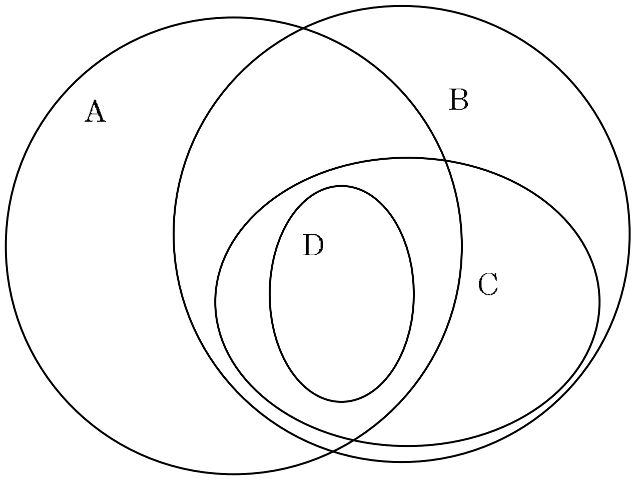
   b. p_1 is equal to or shorter than p_2 and the first arc (u, x) on p_1 activates the thru function coming to the vertex x.

Definition 6.2.12. A sequential circuit S is said to be ***k-acyclically testable*** if the R-graph G_R of S contains a set of thru trees B that is k-consistent with G_R and covers all the vertices of a feedback vertex set of G_R. A sequential circuit S is said to be ***acyclically testable*** if S is k-acyclically testable for some constant k.

Example 6.2.12. S3 is an acyclically testable sequential circuit. Its ATEG will be showed in the following subsection.

Since an acyclic sequential circuit is an acyclically testable sequential circuit with empty feedback vertex set according to definitions of acyclically testable sequential circuits, a sequential circuit is acyclically testable if it is acyclic but the converse is not correct. Therefore, the following theorem is concluded.

Theorem 6.2.13. *The class of acyclically testable sequential circuits is a proper superset of the class of acyclic sequential circuits. (Figure 6.22)*



A: Acyclically Testable Sequential Circuits type A.
B: Acyclically Testable Sequential Circuits type B.
C: Acyclically Testable Sequential Circuits type C.
D: Acyclic Sequential Circuits

Figure 6.22. Relationship between acyclically testable sequential circuits and acyclic sequential circuits.

6.3. Time Expansion Model

Time expansion model (TEM) has been introduced in [30] as a test generation model for acyclic sequential circuits based on time expansion graph (TEG). A topology graph is a directed graph of circuit representation where a vertex v denotes a combinational logic block while an arc (u, v) represents a connection from combinational logic block u to combinational logic block v. The authors defined time expansion graph (TEG) for the topology graph of a given acyclic sequential circuit. To facilitate the discussion of test generation model for acyclically testable sequential circuits, the time expansion graph (TEG) that is used to derive a time expansion model for a given acyclic sequential circuit represented by R-graph is redefined.

Definition 6.3.1. Let S be an acyclic sequential circuit and let $G_R = (V, A, w, h, t)$

be the R-graph of S. Let $G_T = (V_E, A_E, T, l)$ be a directed graph, where V_E is a set of vertices, A_E is a set of arcs, T is a mapping from V_E to a set of integer and l is a mapping from V_E to the set of vertices in R. If graph G_T satisfies the following five conditions, graph G_T is said to be a ***time-expansion graph (TEG)*** of G_R.

C1 (Input/Output and register preservation): The mapping l is a surjective, i.e., $\forall v \in V, \exists u \in V_E, s.t. v = l(u)$.

C2 (Logic preservation) Let u be a vertex in G_T. For any direct predecessor $v(\in pre(l(u)))$ of $l(u)$ in G_R where $v \neq l(u)$, there exists a vertex u' in G_T such that $l(u') = v$ and $u' \in pre(u)$. Here, $pre(v)$ denotes the set of direct predecessors of v.

C3 (Time consistency) For any arc (u, v) $(\in A_E)$, there exists an arc $(l(u), l(v))$ such that $T(v) - T(u) = 1$ if $l(u)$ corresponds to a register or a primary input and $l(v)$ corresponds to a register. $T(v) - T(u) = 0$ if $l(u)$ corresponds to a register and $l(v)$ corresponds to a primary output.

C4 (Time uniqueness) For any pair of vertices u, v $(\in V_E)$, if $T(u) = T(v)$ and if $l(u) = l(v)$, then the vertices u and v are identical, i.e., $u = v$.

C5 (Hold consistency): For any arc (u, v) in G_T, if $T(v) - T(u) = 1$ and $l(v) = l(u) = w$, w is a hold register $(r(w) = h)$ that is in hold mode at $T(u)$ and the number of predecessors of v is one.

Definition 6.3.2. Let S be an acyclic sequential circuit, let $G_R = (V, A, w, h, t)$ be the R-graph of S, and let $G_T = (V_E, A_E, T, l)$ be a TEG of G_R. The combinational equivalent $C_E(S)$ obtained by the following procedure is said to be the ***time expansion model (TEM)*** of S based on G_T.

1. For each time frame, replace each vertex with a connection without a register and replace each arc with the combinational logic block where the corresponding combinational path (represented by the arc) is located. Each combinational logic block appears at most once at each time frame.

2. A logic gate in each logic block is removed if it is not reachable to any input of other logic blocks.

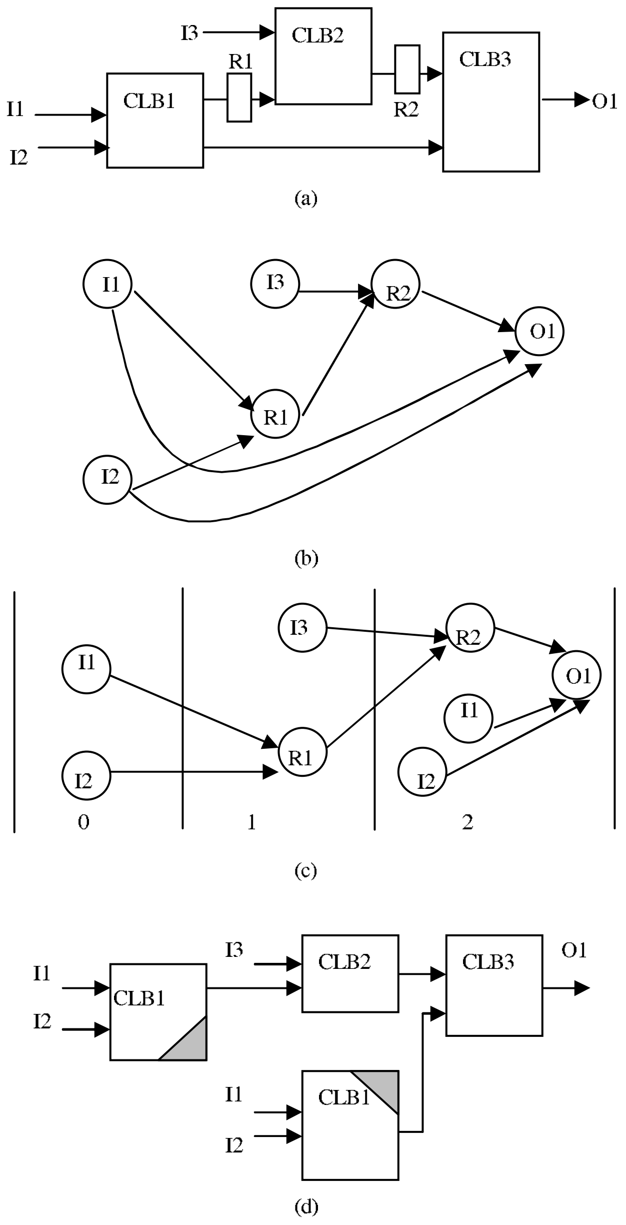


Figure 6.23. Example of time expansion model.(a) Acyclic sequential circuit S8. (b) R-graph of S8. (c) Time expansion graph of S8. (d) Time expansion model of S8.

Example 6.3.1. Figure 6.23(b) shows the R-graph of one of the acyclic sequential circuit S8 in Figure 6.23(a). Its time expansion graph (TEG) and its time expansion model (TEM) are derived in Figure 6.23(c) and 6.23(d).

6.4. Acyclically-Extended Time Expansion Model

This section introduces a test generation model called acyclically-extended time expansion model (ATEM) to perform the test generation on acyclically testable sequential circuits. The procedure of test generation is also described. In the following text, the vertex that corresponds to a primary input (resp. primary output) is called input vertex (resp. output vertex) while the vertex that corresponds to a register (resp. flip-flop) is called register vertex (resp. flip-flop vertex). First, acyclically-extended time expansion graph (ATEG) is defined. Some properties of ATEG are introduced. Based on ATEG and the properties, ATEM is redefined.

Definition 6.4.1. Let S be an acyclically testable sequential circuit with acyclic test thru trees B and let $G_R = (V, A, w, h, t)$ be the R-graph of S. The ***acyclically-extended time expansion graph (ATEG)*** $G_A = (V_A, A_A, T, l)$ with respect to B is a directed graph that satisfies the following conditions.

C1 (Input/Output and register preservation): The mapping l is a surjective, i.e., $\forall v \in V$, $\exists u \in V_A$, s.t. $v = l(u)$.

C2 (Logic preservation for fault excitation phase): Let u be a vertex in G_R. For any direct predecessor $v(\in pre(u))$ of u in G_R, there exists vertices w and x in G_A such that $l(w) = u$, $l(x) = v$, $x \in pre(w)$ and $|pre(w)| = |pre(u)|$. Here, $pre(w)$ (resp. $pre(u)$) denotes the set of direct predecessors of w (resp. u) and $|pre(w)|$ (resp. $|pre(u)|$) denotes the number of all direct predecessors of w (resp. u).

C3 (Thru tree for justification and propagation): Let u be a vertex in a thru tree T_i in B in G_R. Let $W \subset pre(u)$ be a set of all direct predecessors of u in T_i. Let t_j be a thru function on all incoming arcs of u in T_i and V_{tj} be a set of vertices that activate t_j. For each u in T_i in B in G_R, there exists a vertex v in G_A which satisfies the following conditions.

i $l(v) = u$;

ii For each vertex x in $pre(v)$, the following conditions are satisfied.

a. If there exists a vertex w' in W such that $l(x) = w'$ then $x \notin pre(z)$ for any z where $l(z)$ is a vertex included in any other thru tree T_k except T_i and $x \notin pre(y)$ such that $l(y) = l(x)$;

b. Let T_k be a thru tree that is activated by $l(x)$. If $l(x) = l(v)$, then $|pre(v)| = 1$ and $x \notin pre(z)$ for any z where $l(z) \neq l(v)$ and $l(z)$ is a vertex that is not included in thru tree T_k;

c. If $l(x) \in V_{tj}$, then $x \notin pre(z)$ for any z where $l(z) \neq l(x)$ and $l(z)$ is a vertex that is not included in thru tree T_i.

$|pre(v)|$ is the number of vertices in $pre(v)$.

C4 (Time consistency): For any arc (u, v) ($\in A_A$), there exists an arc $(l(u), l(v))$ such that $T(v) - T(u) = 1$ if $l(u)$ corresponds to a register or a primary input and $l(v)$ corresponds to a register. $T(v) - T(u) = 0$ if $l(u)$ corresponds to a register and $l(v)$ corresponds to a primary output.

C5 (Time uniqueness): For any pair of vertices u, v ($\in V_A$), if $T(u) = T(v)$ and if $l(u) = l(v)$, then the vertices u and v are identical, i.e., $u = v$.

C6 (Hold consistency): Let u be a vertex in G_A. Let v ($\in pre(u)$) be a predecessor of u. If $|pre(u)| < |pre(l(u))|$ and $l(u) = l(v) = w$, then $r(w) = h$ and $|pre(u)| = 1$.

C7 (Input Independency): Let u, v be two vertices in G_A. Let p_i and p_j be a pair of reconvergent paths that start from u and end at v. Let w be a vertex on p_i such that $u \in pre(w)$. Let x be a vertex on p_j such that $u \in pre(x)$. For each pair of paths p_i, p_j where $w \neq x$, $|pre(w)| = |pre(l(w))|$ and $|pre(x)| = |pre(l(x))|$.

The following three examples are used to explain condition C3.

Example 6.4.1. Figure 6.24(a) shows the time expansion models of S3 that does not satisfy condition C3(ii)(a). Condition C3(ii)(a) tells that if a vertex w' at time m is used to justify another vertex u at time $m + 1$, then w' cannot be used simultaneously to activate a thru function at time m. In Figure 6.24(a), $I1$ at time 3 (corresponding to w' at time m) is used to justify $R2$ at time 4 (corresponding u at time $m+1$) but at the same time $I1$ is used to activate thru functions t_1 and t_4. This violates Condition C3(ii)(a).

Example 6.4.2. Condition C3(ii)(b) tells that if a vertex $r = l(x)$ is in HOLD mode at time m, r cannot be used to justify signal for any thru tree through a thru function at time m. But r can be used to activate a thru function in a thru tree at time m. In Figure 6.24(b), $R2$ is in hold mode at time 2 (corresponding to r at m). At time 2, $R2$ is used to activate thru functions t_1. At the same time, $R2$ is used to justify the value of $O2$ at 3. This violates Condition C3(ii)(b).

Example 6.4.3. Condition C3(ii)(c) tells that when a vertex $r = l(x)$ at time m is used to activate a thru function going to vertex u at time $m+1$, r cannot be used to justify signal for any thru tree through a thru function at time m. But r can be in HOLD mode. For example, Figure 6.24(c) shows that $R2$ (corresponding to r) is activating thru function t_1 at time 3 and at the same time $R2$ is justifying $R5$. But these two events are not allowed to happen at the same time.

As the first step of designing a test generation procedure for acyclically testable sequential circuits, the logic for hold function is assumed fault free. The tests for these faults can be generated separately and the test generation procedure for these faults will be considered in the future works. To guarantee the test generation for faults in thru functions, each register in the feedback vertex set are regarded as having reset function.

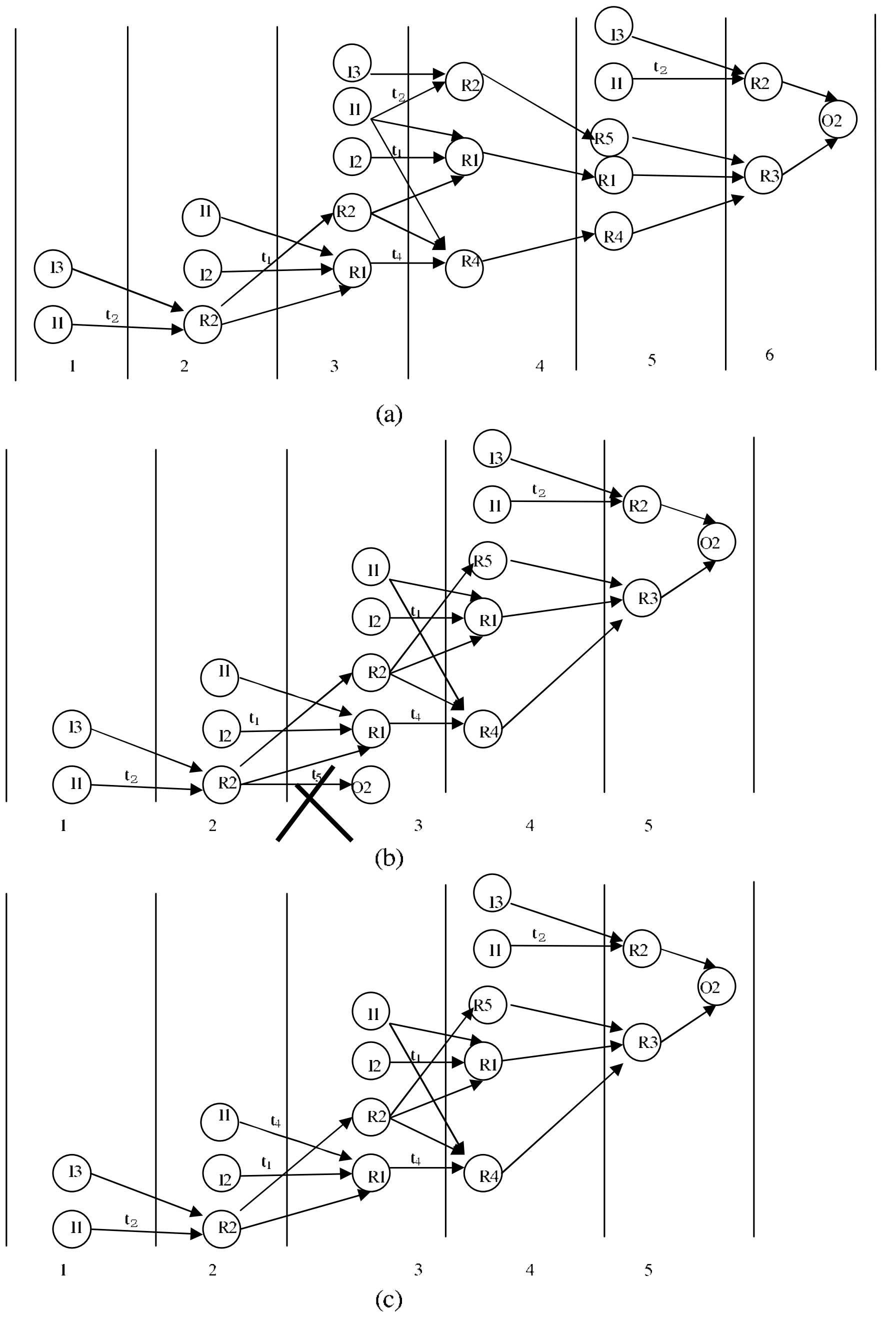


Figure 6.24. Time expansion models of S3 that violates C3.

Definition 6.4.2. Let S be a given acyclically testable sequential circuit. The ***acyclically-extended time expansion model (ATEM)*** of S is the combinational equivalent obtained by the following procedure.

1. For each time frame, replace each vertex with a connection without a register and replace each arc with the combinational logic block where the corresponding combinational path (represented by the arc) is located. Each combinational logic block appears at most once at each time frame.

2. A logic gate in each logic block is removed if it is not reachable to any input of other logic blocks.

3. Each input that corresponds to an output of a register is assigned don't care value.

Example 6.4.4. Figure 6.25 shows the ATEM of S3 with respect to output O2.

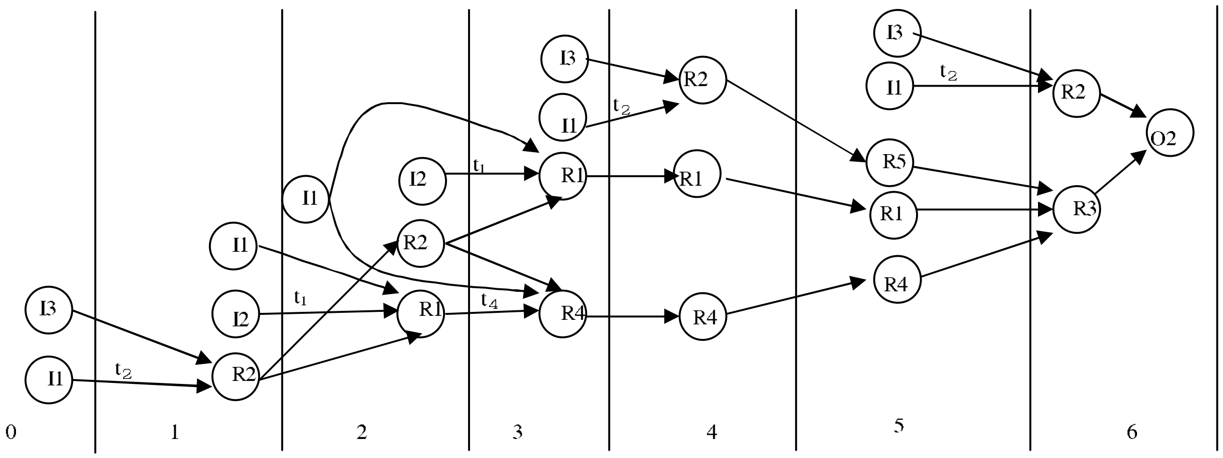


Figure 6.25. ATEM for S3.

6.5. Test Generation Procedure

For each stuck-at fault in a given acyclically testable sequential circuit, the test generation process is done as follows using ATEM test generation algorithm. Multiple fault modeling of [31] is considered.

Step 1 Generate an acyclically-extended time expansion model (ATEM) of the sequential circuit.

Step 2 Transform the combinational equivalent ATEM into its multiple fault model.

Step 3 Apply combinational ATPG on the multiple fault model.

Step 4 Derive the test sequence from the test pattern obtained from the test generation on the multiple fault model of the ATEM.

Theorem 6.5.1. *The ATEM test generation algorithm can identify redundancy and all testable faults.*

Theorem 6.5.2. *The test generation complexity of the acyclically testable sequential circuits is τ^2-bounded.*

6.6. Design for Testability Method

In this section, a design for testability(DFT) method to augment a given sequential circuit into an acyclically testable sequential circuit is introduced. The DFT method performs some operations on R-graph and it is designed to induce minimum area overhead. The procedure consists of the following three steps.

Step 1 Identify the vertices of minimum feedback vertex set (MFVS).

Step 2 Identify existing thru trees.

Step 3 Group the vertices of MFVS into two groups, G1, G2 and G3 as follows.

3.1 Group a vertex u into G1 if it corresponds to a register or input/output that activate a thru function. If the vertex is in an existing thru tree T_i, group all the vertices in T_i in G1. If G1 has only input/output, G1 is made empty.

3.2 Group the remaining register vertices in MFVS into G2.

3.3 Group the remaining input/output vertices into G3.

Step 4 For each group of G1 and G2, the following is done.

4.1 Check that at least one input vertex and one output vertex exist in the group. If the group does not have input vertex (resp. output vertex), one input vertex (resp. output vertex) is taken from G3. If G3 does not have one, a new vertex is added into the group.

4.2 Group each vertex (except output vertex) into a group called potential source if the vertex does not have an outgoing arc labeled with a thru function.

4.3 Group each vertex (except input vertex) into a group called potential destination if the register vertex does not have an incoming arc labeled with a thru function.

4.4 For each vertex u in the group of potential source, introduce a new outgoing arc labeled with a new thru function t_{new} to connect u to a vertex v in the group of potential destination. u and v are taken out from the groups of potential sources and potential destination, respectively.

4.5 Repeat 4.4 until the group of potential destination is empty or the group of potential desitination has only output vertices.

4.6 For each vertex u in the group of potential source, introduce a new outgoing arc labeled with a new thru function t_{new} to connect u to an output vertex v that does not have an incoming arc labeled with thru functions. If the group does not have one, an output vertex is taken from G3 to the group. If G3 does not have one, a new output vertex is introduced to the group.

Step 5 If G1 is not empty, each register in G1 and G2 is augmented into a hold register. For other register vertices in MFVS, each register is augmented into a register with reset function.

Step 1 is done by using an exact algorightm for selecting partial scan flip-flops introduced in [32]. All the new thru functions t_{new} introduced in the DFT method are same. For example $t_{new} = r$ means the new thru function is activated when $r = 1$ where r can be an existing primary input or a new primary input.

6.7. Case Studies

In the case studies, experiments are conducted on RTL benchmark circuits, which are datapaths of varying bit width. Our DFT method is applied on the datapaths of GCD, LWF, JWF, and MPEG and compare the area overhead of the augmented circuits with that of the full scanned circuits and the partial scanned circuits. Partial scanned circuits are the circuits whose minimum feedback set of flip-flops are scanned so that the augmented circuits are acyclic. Thus, the circuits modified with partial scan and with our DFT method have same test generation complexity. Table 6.1 presents the characteristics of the benchmark circuit. Table 6.2 shows the fault coverage and fault efficiency of each benchmark circuit. Each fault testable in the partial scan designed circuits is also testable in the corresponding circuit augmented by our DFT method, and vice versa. Table 6.3 shows the area overhead where one unit of area corresponds to the size of an inverter and pin overhead. It shows that the area overhead of the benchmark circuits augmented by our method is less than that of the full scanned circuits and the partial scanned circuits. The pin overhead in our method comes from the reset function and extra input to control the new thru functions. Table 6.4 tells that the test generation time for the original circuits is large while the test generation time for the partial scan designed circuits as well as the acyclically testable sequential circuits is small. Table 6.4 also gives the information that the test application time of the circuits under our augmentation is more than the original circuits' but less than the partial scan.

Table 6.1. Characteristics

B/mark	Original			
	#Flip-flops	Area	#Primary inputs	#Primary outputs
GCD	48	1383	40	19
LWF	80	1763	39	32
JWF	224	5925	106	80
MPEG	1928	46772	499	128

Table 6.2. Number of faults, fault efficiency and fault coverage

	Original		Full scan		Partial scan		Our method	
B/mark	FC(%)	FE(%)	FC(%)	FE(%)	FC(%)	FE(%)	FC(%)	FE(%)
GCD	99.75	99.75	100	100	100	100	100	100
LWF	99.94	99.94	100	100	100	100	100	100
JWF	98.70	98.70	100	100	100	100	100	100
MPEG	84.80	84.80	100	100	100	100	100	100

Table 6.3. Area and pin overhead

	Full scan		Partial scan		Our method	
B/mark	Area(OH%)	Pin OH	Area(OH%)	Pin OH	Area(OH%)	Pin OH
GCD	1719(24.30)	3	1495(8.10)	3	1415(2.31)	1
LWF	2323(31.76)	3	1875(6.36)	3	1798(1.99)	2
JWF	7493(26.46)	3	6485(9.45)	3	5957(0.54)	2
MPEG	60268(28.85)	3	47612(1.80)	4	47556(1.68)	2

6.8. Conclusion

A new class called acyclically testable sequential circuits has been introduced. The test generation complexity of the acyclically testable sequential circuits is τ^2-bounded. On the other hand, acyclically testable sequential circuits are at-speed testable. The DFT method to augment an arbitrary sequential circuit into an acyclically testable sequential circuit has been introduced. Experimental results showed that the area overhead of the resulting augmented circuits is less compared to the partial scan designed circuits. Complete fault efficiency is also achieved and the test generation time is low. Moreover, the test application time is less than the test application time of the full scanned circuits and partial scanned circuits.

Table 6.4. Test generation time and test application time

B/mark	Test generation time(s)				Test application time (clock cycles)			
	Original	Full scan	Partial scan	Our method	Original	Full scan	Partial scan	Our method
GCD	87.19	0.02	0.19	0.43	159	6124	3334	815
LWF	49.02	0.02	0.06	0.40	59	4049	1444	196
JWF	1689.14	0.08	0.50	13.48	103	17100	12488	1648
MPEG	2646.42	0.18	12.05	33.91	114	162035	31822	9690

Chapter 7

Conclusion and Future Works

7.1. Conclusion

τ^k notation has been introduced in order to clarify the test generation complexity. Based on this notation, balanced sequential circuits, strongly balanced sequential circuits, internally balanced sequential circuits have been shown to be τ-equivalent while the class of acyclic sequential circuits has been shown to be τ^2-bounded. FSR scan technique was introduced. The test generation complexity for FSR scan designed circuits and scan designed circuits were shown to be equivalent to the circuit kernels of each design. A case study for which FSR scan design has lower test application time. We introduced several classes of easily testable cyclic sequential circuits including l-length-bounded testable circuits and l-length-bounded validity-identifiable circuits with $l = O(n)$, t-time-bounded testable circuits and t-time-bounded validity-identifiable circuits with $t = \tau(n)$ or $\tau^2(n)$, state-shiftable FSM realizations and counter-cycle FSM realizations. The case studies indicate that state-shiftable FSM realization can be better than its corresponding counter-cycle FSM realization and its corresponding full scan designed circuit in certain cases while full scan designed circuit has better result than its corresponding counter-cycle FSM realization in certain cases.

The relationships between the PDF test generation and SAF test generation of several existing classes of circuits have been described in this paper. The PDF test generation of internally balanced sequential circuits under rated clock and slow-fast-slow clock is equivalent to the SAF test generation of combinational circuits. On the other hand, the PDF test generation of the acyclic sequential circuits are τ^2-bounded under slow-fast-slow clock. It is shown that under TEM with slow-fast-slow clock the PDF test generation is not τ-equivalent. The PDF test generation under slow-fast-slow clock and the SAF test generation of two-column SSFSM realizations with observable shifting logic are equivalent to the SAF test generation of the combinational circuits while for two-column distributive SSFSM realizations, its PDF test generation under slow-fast-slow clock is τ-equivalent but its SAF test generation is τ^2-bounded.

A new class called acyclically testable sequential circuits has been introduced. The test generation complexity of the acyclically testable sequential circuits is τ^2-bounded. On the other hand, acyclically testable sequential circuits are at-speed testable. The DFT method to augment an arbitrary sequential circuit into an acyclically testable sequential circuit has been introduced. Experimental results showed that the area overhead of the resulting augmented circuits is less compared to the partial scan designed circuits. Complete fault efficiency is also achieved and the test generation time is low. Moreover, the test application time is less than the test application time of the full scanned circuits and partial scanned circuits.

7.2. Future Works

Several future works are identified. In Chapter 4, synthesis for testability methods were done by hand on five mcnc benchmark circuits. Automation is foreseen as a task that can show the effectiveness of the SFT methods on the larger circuits.

In Chapter 5, several classes of easily testable sequential circuits have been identified and discussed theoretically but experiment is not done on benchmarks circuits. A design for testability method can be designed based on the definitions of the classes of easily testable sequential circuits. On the other hand, only the test generation complexity for sequential circuits with robust and non-robust faults was studied. The work can be extended to cover functional sensitizable and functional unsensitizable path delay faults, as well as transition faults.

The PDF test generation of acyclic sequential circuits and cyclic sequential circuits was discussed under the assumption of slow-fast-slow clock. The discussion of PDF test generation of sequential circuits under rated clock remains an open problem. The conjectures are summarized as follows.

1. Proving that the class of acyclic sequential circuits with stuck-at faults is not τ-equivalent.

2. Proving that the class of acyclic sequential circuits with path delay faults is not τ-equivalent.

3. Proving that the SAF test generation of the two-column distributive SSFSM realizations is not τ-equivalent.

Regarding 1, the test generation complexity for acyclic sequential circuits with stuck-at faults and path delay faults was proved to be τ^2-bounded under time expansion model. It has not yet been proved to be not τ-equivalent. The following are the open problems identified in Chapter 5.

1. Is the PDF test generation of acyclic sequential circuits τ^2-bounded under rated clock?

2. Is the PDF test generation of acyclic sequential circuits not τ-equivalent under TEM with rated clock?

3. Is the PDF test generation of acyclic sequential circuits not τ-equivalent under rated clock?

4. Is the PDF test generation of acyclic sequential circuits under rated clock equivalent to the SAF test generation of acyclic sequential circuits?

5. Are the PDF test generation under rated clock and the SAF test generation of two-column SSFSM realizations with observable shifting logic equivalent?

6. Is the PDF test generation of the two-column distributive SSFSM realizations τ-equivalent under rated clock?

In Chapter 6, a test generation procedure is done on the acyclically testable sequential circuits under the assumption that the logic related to the hold function of a register is fault free. As a future work, a test generation method should be designed to generate tests for the faults related to the hold function. Furthermore, it is currently assumed that hold functions are activated only by primary inputs. But this also should be relaxed so that hold functions are activated by PIs and registers, in the same way as activating thru functions. The case study was only conducted on the datapath circuits. It should be extended to the circuits consisting of datapath and controller.

Acknowledgements

This dissertation is the result of three years of work whereby I have been accompanied and supported by many people. I would like to take the opportunity to express my gratitude to all of them.

The first person I would like to thank is my supervisor Professor Hideo Fujiwara. I have been under his supervision since 2003 when I started my doctoral course. During these years, Professor Hideo Fujiwara always provided a motivating, enthusiastic, and critical atmosphere during the many discussions we had. It was a great pleasure to me to conduct this dissertation under his supervision. I owe him lots of gratitude for many constructive comments, guidance and support. He could not even realize how much I have learned from him.

I would like to thank my co-supervisors, Professor Hiroyuki Seki and Associate Professor Michiko Inoue who monitored my work and took effort in reading and providing me with valuable comments on earlier versions of this dissertation. I thank you all.

I am indebted to Assistant Professors Satoshi Ohtake and Tomokazu Yoneda for their friendly discussion and continuous cooperation.

I am highly thankful to Dr. Thomas Clouqueur for his constant guidance and encouragement throughout the course of my doctoral research. He gave lots of simulating suggestions and constructive comments to my research.

I am also grateful to Professor Tomoo Inoue, Associate Professor Hideyuki Ichihara and Mr. Nobuya Oka, Hiroshima City University, for their valuable comments during our research collaboration.

The members and ex-members of Fujiwara laboratory gave me the feeling of being at home at work. Dr. Tsuyoshi Iwagaki, Dr. Kazuko Kambe, Dr. Virendra Singh, Dr. Zhiqiang You, Mr. Masato Nakazato, Mr. Yuuki Yoshikawa, Mr. Vorayos Thongtan, many thanks for being my colleagues.

My doctoral course has been supported and funded by Universiti Teknologi Malaysia, Malaysia. I thank the university for their confidence in me.

I am also grateful to my family members for their love and moral support they gave during these three years.

References

[1] H. Fujiwara, S. Toida, “The complexity of fault detection: an approach to design for testability,” Proc. of the 12th Int. Symp. on Fault Tolerant Computing, pp. 101-108, Jun. 1982.

[2] P. Goel, “Test generation cost analysis and projections,” Proc. 17th DAC, pp. 77-84, Jun. 1980.

[3] M.R. Prasad, P. Chong and K. Keutzer, “Why is ATPG easy?” Proc. 36th DAC, pp.22-28, Jun. 1999.

[4] E. Gizdarski and H. Fujiwara, “SPIRIT: A highly robust combinational test generation algorithm,” IEEE Trans. Comput-Aided Des. Integra. Circuit Syst., Vol. 21, No. 12, pp. 1446-1458, Dec. 2002.

[5] R. Gupta and M. A. Breuer, “The BALLAST methodology for structured partial scan design,” IEEE Trans. on Computers, Vol. C-39, No. 4, pp. 538-544, Apr. 1990.

[6] A. Balakrishnan and S.T. Chakradhar, ”Sequential circuits with combinational test generation complexity,” Proc. IEEE Int. Conf. VLSI Design, pp. 111-117, Jan. 1996.

[7] H. Fujiwara, “A new class of sequential circuits with combinational test generation complexity,” IEEE Trans. on Computers, Vol. 49, No. 9, pp. 895-905, Sept. 2000.

[8] R. Gupta and M.A. Breuer, “Testability properties of acyclic structures and applications to partial scan design,” Proc. IEEE VLSI Test Symposium, pp. 49-54, 1992.

[9] R. Gupta and M.A. Breuer, “Partial scan design of register transfer level circuits,” JETTA, Vol. 7, pp. 25-46, 1995.

[10] M. Inoue, C. Jinno and H. Fujiwara, “An extended class of sequential circuits with combinational test generation complexity,” Proc. 20th Int. Conf. on Computer Design, pp. 200-205, Sept. 2002.

[11] C. Y. Ooi and H. Fujiwara, "Classification of sequential circuits based on τ^k-Notation," Proc. of ATS, pp. 348-353, Nov. 2004.

[12] C. Y. Ooi, T. Clouqueur and H. Fujiwara, "Classification of sequential circuits based on τ^k notation and its applications," IEICE Trans. on Information and Systems, pp. 2738-2747, Dec. 2005.

[13] A. Saldanha, R. K. Brayton, A. L. Sangiovanni-Vincentelli, "Equivalence of robust delay-fault and single stuck-fault test generation," Proc. 29th DAC, pp. 173-176, 1992.

[14] M. A. Gharaybeh, M. L. Bushnell and V. D. Agrawal, "Classification and test generation for path-delay faults using single struck-at fault tests," JETTA, pp. 55-67, 1997.

[15] S. Ohtake, K. Ohtani and H. Fujiwara, "A method of test generation for path delay faults using stuck-at fault test generation algorithms," Proc. of DATE, pp. 310-315, 2003.

[16] S. Ohtake, S. Miwa and H. Fujiwara, "A method of test generation for path delay faults in balanced sequential circuits," Proc. 20th VTS, pp. 321-327, 2002.

[17] S. Majumder, B. B. Bhattacharya, V. D. Agrawal, M. L. Bushnell, "A complete characterization of path delay faults through stuck-at faults," 12th Int. Conf. on VLSI Design, pp. 492-497, Jan. 1999.

[18] T. Inoue, T. Hosokawa, T. Mihara and H. Fujiwara, "An optimal time expansion model based on combinational ATPG for RTL circuits," Proc. 7th ATS, pp. 190-197, Dec. 1998.

[19] T. Iwagaki, S. Ohtake and H. Fujiwara, "Acceleration of transition test generation for acyclic sequential circuit utilizing constrained combinational stuck-at test generation," Proc. 10th ETS, pp. 48-53, 2005.

[20] A. Krstic and K.-T. Cheng, Delay Fault Testing for VLSI Circuits, Kluwer Academic Publishers, 1998.

[21] K. Heragu, J. H. Patel and V. D. Agrawal, "Segment delay faults: A new fault model," Proc. of 14th VTS, pp. 32-39, 1996.

[22] T.H. Cormen, C.E. Leiserson and R.L. Rivest, Introduction of Algorithms, The MIT Press, 1990.

[23] H. Fujiwara and K. Kinoshita, "Easily testable sequential machines," Technology Reports of The Osaka University, Vol. 24, No. 1214, 1974.

[24] A. Ghosh, S. Devadas, and A.R. Newton, Sequential Logic Testing and Verification, Kluwer Academic Publishers, Boston/Dordrecht/London, 1992.

[25] S. Yang, "Logic synthesis and optimization benchmarks user guide version 3.0," Report of the Microelectronics Center of North Carolina, 1991.

[26] T. Iwagaki, S. Ohtake and H. Fujiwara, "A path delay test generation method for sequential circuits based on reducibility to combinational test generation," Digest of Papers 8th IEEE ETW, pp. 307-312, May 2003.

[27] C. Y. Ooi, T. Clouquer and H. Fujiwara, "Test generation complexity for path delay faults based on τ^k-notation," Digest of Papers 6th WRTLT, pp. 61-72, 2005.

[28] C. Y. Ooi and H. Fujiwara, "Acyclically testable sequential circuits type A and B," NAIST Technical Report, NAIST-IS-TR2006004, July 2006.

[29] C. Y. Ooi and H. Fujiwara, "A new class of sequential circuits with acyclic test generation complexity," 24th IEEE ICCD, October 2006 (To appear).

[30] T. Inoue, D. K. Das, C. Sano, T. Mihara, H. Fujiwara, "Test generation for acyclic sequential circuits with hold registers," Proc. 18th Int. Conf. on Computer Design, pp. 550-556, 2000.

[31] C. Y. Ooi, T. Clouqueur and H. Fujiwara, "Test generation complexity for stuck-at and path delay faults based on τ^k-notation," NAIST Technical Report, NAIST-IS-TR2005003, May 2005.

[32] S. T. Chakradhar, A. Balakrishnan, V. D. Agrawal, "An exact algorithm for selecting partial scan flip-flops," JETTA, pp. 83-93, 1995.

[33] H. Iwata, T. Yoneda, S. Ohtake and H. Fujiwara, “A DFT method for RTL data paths based on partially strong testability to guarantee complete fault efficiency,” Proc. IEEE the 14th ATS, pp. 306-311, Dec. 2005.

[34] C. Lin, M. Marek-Sadowska, M.T. Lee and K. Chen, “Cost-free scan: a low-overhead scan path design,” IEEE Trans. CAD Integra. Circuit and Sys., Vol. 17, No. 19, pp. 852-861, Sept. 1998.

[35] A. Ghosh, S. Devadas and A. R. Newton, Sequential Logic Testing and Verification, Kluwer Academic Publishers, 1992.