PAPER

# Classification of Sequential Circuits Based on $\tau^k$ Notation and Its Applications

**Chia Yee OOI**[†a)], ***Student Member***, **Thomas CLOUQUEUR**[†], ***Nonmember***, ***and*** **Hideo FUJIWARA**[†], ***Fellow***

**SUMMARY** This paper introduces $\tau^k$ notation to be used to assess test generation complexity of classes of sequential circuits. Using $\tau^k$ notation, we reconsider and restate the time complexity of test generation for existing classes of acyclic sequential circuits. We also introduce a new DFT method called feedback shift register (FSR) scan design technique, which is extended from the scan design technique. Therefore, for a given sequential circuit, the corresponding FSR scan designed circuit has always equal or lower area overhead and test application time than the corresponding scan designed circuit. Furthermore, we identify some new classes of sequential circuits that contain some cyclic sequential circuits, which are $\tau$-equivalent and $\tau^2$-bounded. These classes are the $l$-length-bounded testable circuits, $l$-length-bounded validity-identifiable circuits, $t$-time-bounded testable circuits and $t$-time-bounded validity-identifiable circuits. In addition, we provide two examples of circuits belonging to these classes, namely counter-cycle finite state machine realizations and state-shiftable finite state machine realizations. Instead of using a DFT method, a given sequential circuit described at the finite state machine (FSM) level can be synthesized using another test methodology called synthesis for testability (SFT) into a circuit that belongs to one of the easily testable classes of cyclic sequential circuits.

***key words:*** *test generation, easily testable sequential circuits, complexity, design for testability, synthesis for testability*

## 1. Introduction

It has been known for more than two decades that the test generation problem for single stuck-at faults is NP-complete. However, empirical observations show that the combinational test generation complexity seems to be $O(n^r)$ for some constant $r$, where $n$ is the size of the circuit [1], [2]. For example, the ATPG tool named SPIRIT [16] can achieve 100% fault efficiency for benchmark circuits ITC'99, surpassing the existing commercial ATPGs. Consequently, studies have been conducted to search for classes of sequential circuits with combinational test generation complexity.

Classes of sequential circuits with combinational test generation complexity include balanced sequential circuits [3], strongly balanced sequential circuits [4], internally balanced sequential circuits [6], switched balanced sequential circuits [7] and switched internally balanced sequential circuits [8], all of which are acyclic. Also, a test generation model (TGM) was proposed to transform any acyclic sequential circuit into its combinational equivalent with logic duplicates for at most $d$ time frames where $d$ is the sequential depth [5]. On the other hand, the test generation problem for general sequential circuits, which is modeled by an iterative logic array, has greater time complexity than for acyclic sequential circuits. To ease test generation, a given sequential circuit can be augmented into a circuit with a kernel belonging to one of the classes of acyclic sequential circuits using a design for testability (DFT) method. In our discussion, we assume combinational test generation achieves complete fault efficiency in general based on the work in [16]. So, the test generation of these augmented circuits also result in complete fault efficiency. Therefore, combinational test generation complexity is fundamental in the discussion on the time complexity of test generation of sequential circuits. As more and more classes are introduced, a general notation would be useful in facilitating the discussion on classification. Therefore, prior to introducing some larger classes of easily testable sequential circuits, we define a test generation notation that we call $\tau^k$ notation based on combinational test generation complexity. In our discussion, we assume the combinational test generation complexity is $\Theta(n^r)$ for some constant $r > 2$ and it is denoted by $\tau(n)$, where $n$ is the size of the circuit.

As mentioned, previously defined classes of sequential circuits with combinational test generation complexity are acyclic [6]–[8]. In this paper, we show that some cyclic sequential circuits also have combinational test generation complexity. 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. Rather than DFT, we use synthesis for testability (SFT) methods to augment and synthesize a given sequential circuit into a circuit belonging to one of these classes.

In Sect. 2, using asymptotic notation, we define $\tau^k$ notation. The section also discusses a test generation model called time expansion model (TEM). In Sect. 3, we reconsider the time complexity of test generation problem for the existing classes of acyclic sequential circuits using the $\tau^k$ notation. Section 4 presents a new DFT method called the FSR scan design. In Sect. 5, several *$\tau$-equivalent* and *$\tau^2$-bounded* classes of sequential circuits, which include some cyclic sequential circuits, are introduced. Section 6 discusses the need for SFT method to augment and synthesize a given sequential circuit at FSM level into an easily testable realization or circuit. Conclusion is presented in the final section.

Manuscript received December 10, 2004.
Manuscript revised July 14, 2005.

[†] The authors are with the Graduate School of Information Science, Nara Institute of Science and Technology (NAIST), Ikoma-shi, 630–0192 Japan.
a) E-mail: chiaye-o@is.naist.jp

DOI: 10.1093/ietisy/e88–d.12.2738

## 2. Preliminaries

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 $f(n)$ [9].

To facilitate our discussion, we define the time complexity of test generation problem 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 $S$?

$P_\alpha$: **Class $\alpha$ Test Generation Problem**

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

**Definition 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 $P_\alpha$, 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 $\alpha$, respectively.

In our discussion, we have the following assumption about the combinational test generation complexity.

**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, we define the $\tau^k$ notation. We consider $T_C(n)$ 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 2:** $T(n)$ is *$\tau^k$-equivalent* if and only if $T(n) = \Theta(\tau^k(n))$ and *$\tau^k$-bounded* if and only if $T(n) = O(\tau^k(n))$, where $k > 0$.

**Definition 3:** Class $\alpha$ is *$\tau^k$-equivalent* if and only if $T_\alpha(n) = \Theta(\tau^k(n))$ and *$\tau^k$-bounded* if and only if $T_\alpha(n) = O(\tau^k(n))$, where $k > 0$.

In the following section, we discuss the test generation complexity of acyclic sequential circuits using a test generation model called time expansion model or TEM [11]. To make the discussion clear, we quote the definition of TEM from [11].

**Definition 4:** 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 5:** Let $S$ be an acyclic sequential circuit and let $G = (V, A, r)$ be the topology graph of $S$. 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 6:** Let $S$ be an acyclic sequential circuit, let $G = (V, A, r)$ be the topology graph of $S$, and let $E = (V_E, A_E, t, l)$ be a TEG of $G$. The combinational circuit $C_E(S)$ obtained by the following procedure is said to be the time expansion model (TEM) of $S$ 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.

The following section reconsiders and restates the test generation complexity of the existing classes of acyclic sequential circuits based on $\tau^k$ notation.

## 3. Existing Classes of Acyclic Sequential Circuits

The test generation problem for the existing classes of acyclic circuits in terms of $\tau^k$ notation gives a clearer picture of the test generation complexity.

A sequential circuit is said to be a *balanced sequential circuit* if, for any pair of primary input and primary output, all paths between them have the same number of flip-flops. A subclass of balanced sequential circuits, which is called strongly balanced sequential circuits, is defined as follows. A sequential circuit is a *strongly balanced sequential circuit* if it is balanced and in addition, all paths between a node and all reachable PIs in its fan-in cone have the same number of flip-flops. A larger class of sequential circuits with combinational test generation complexity is internally balanced sequential circuits. A sequential circuit is an *internally balanced sequential circuit* if a circuit resulting from operation 1 of the extended combinational transformation in [6] on an acyclic sequential circuit is a balanced sequential circuit. It has been shown in previous studies that a sequential circuit belonging to any of these three classes can be transformed into its combinational equivalent [3], [5], [6], the test patterns of which can be transformed back to the test sequences of the original sequential circuit. Thus, we have the following theorem based on $\tau^k$ notation.

**Theorem 1:** The classes of internally balanced sequential circuits, balanced sequential circuits and strongly balanced sequential circuits are $\tau$-equivalent.

An *acyclic sequential circuit* is a sequential circuit without feedback. We show that the test generation complexity for this class is not $\tau$-equivalent if the test generation model is TEM [11].

**Lemma 1:** 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 [12],

$$t(u'_j) = t(v') - r_i(u, v) \tag{1}$$

Since $0 \leq i \leq q$, the range of $j$ is also $0 \leq j \leq q$. Since $u = l(u'_j)$, the lemma is proved. *q.e.d.*

**Theorem 2:** There exists an acyclic sequential circuit whose test generation complexity represented by TEM is not $\tau$-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 $S_a$.

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 Fig. 1.

From Lemma 1, vertex $u$ in the topology graph is mapped to $(d + 1)$ different vertices in TEM as shown in Fig. 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

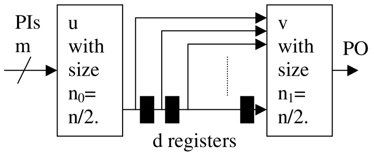

**Fig. 1** Structure of $S_a$.

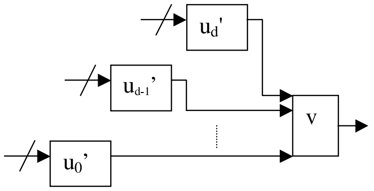

**Fig. 2** TEM of $S_a$.

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

Therefore, the test generation complexity of the acyclic sequential circuit is

$$\begin{aligned} T_A = \tau(N) &= \tau(\frac{n \times d}{2} + n) \\ &= \tau(d \times n) \\ &= \Theta(d^r \times n^r) \neq \Theta(n^r) \\ &\quad \text{for some constant } r. \end{aligned} \tag{3}$$

The equation 3 proves the theorem. *q.e.d.*

However, there might be other test generation models for acyclic sequential circuits besides TEM. "Is $T_A$ $\tau$-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. Therefore, we have the following conjecture and theorem.

**Conjecture 1:** The class of acyclic sequential circuits is not $\tau$-equivalent.

**Theorem 3:** The class of acyclic sequential circuits is $\tau^2$-bounded.

**Sketch of proof:** [5] 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.

## 4. Design for Testability

Design for testability (DFT), e.g. scan design technique, is a solution to reduce the test generation complexity. DFT modifies a given circuit to ease its testing. A given sequential circuit can be augmented into a circuit with kernel (the part of the circuit that is not modified) belonging to one of the classes of acyclic sequential circuits by using DFT. In this section, we propose a new method of design for testability called the FSR scan design technique, which is extended from the scan design technique. We then discuss the test generation complexity for scan designed circuits and FSR scan designed circuits. We also do comparisons of hardware area overhead and test application time between these two DFT methods.

### 4.1 Scan Designed Circuits

A sequential circuit is a *scan designed circuit* if some or all of the flip-flops are replaced with *scan flip-flops* so that they are chained into at least one shift register during test mode and hence they can be directly controlled and observed.

**Theorem 4:** The test generation complexity for the scan designed circuits is $\tau^k$-equivalent ($\tau^k$-bounded) if the test generation complexity for the kernels is $\tau^k$-equivalent ($\tau^k$-bounded).

**Proof:** See proof of theorem in [10].

### 4.2 FSR Scan Designed Circuits

We now propose a new scan design called feedback shift register (FSR) scan design. A sequential circuit is an *FSR scan designed circuit* (Fig. 3) if some or all of the feedback shift registers are converted into *scan shift registers* (Fig. 4) so that the scan shift registers are chained together during test mode and hence they can be directly controlled and observed. Each shift register has its own length $l$. The kernel is a combinational circuit when all the shift registers are converted into scan shift registers. Otherwise, the kernel is a sequential circuit. We call this DFT method FSR scan design technique.

**Theorem 5:** The test generation complexity for the FSR scan design circuit is $\tau^k$-equivalent ($\tau^k$-bounded) if the test generation complexity for the kernel is $\tau^k$-equivalent ($\tau^k$-bounded).

**Proof:** See proof of theorem in [10].

### 4.3 Case Study: Comparison between Scan Design and FSR Scan Design

Not all sequential circuits can be augmented into an FSR scan designed circuit except the trivial augmentation of using only scan shift registers of length 1. Note that the scan design is a special case of the FSR scan design. Therefore, FSR scan design is never worse than scan design. For a given sequential circuit where there exists at least one feedback shift register, the augmentation with FSR scan technique, where the length of scan shift registers is more than 1, may or may not be superior to scan technique in terms of hardware overhead and test application time. The following analysis shows that there exists a case where FSR scan technique results in the same hardware area overhead but lower test application than that with the scan technique.

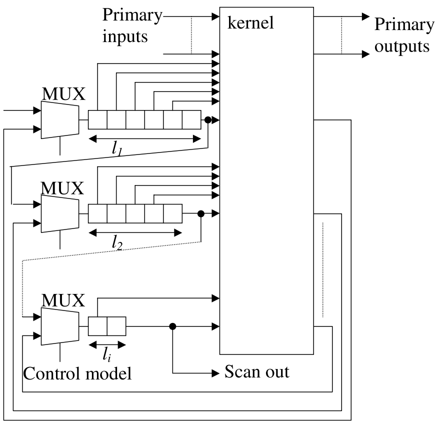

**Fig. 3** FSR scan designed circuit.

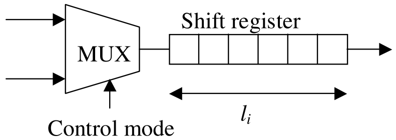

**Fig. 4** Scan shift register.

Let $SCL$ denote a scan chain length while $FSCL$ and $h_i$ denote the number of scan shift registers and length of an $FSR_i$ for $0 \leq i \leq FSCL - 1$, respectively. Let $d_1$ denote the sequential depth of the scan designed circuit kernel while $d_2$ denote the sequential depth of the FSR scan designed circuit kernel. For a test vector, the test application time for a scan designed circuit, $TAT_S$ and a FSR scan designed circuit, $TAT_F$ are given by the following equations.

$$TAT_S = (SCL + 1) * (d_1 + 1) \qquad (4)$$

$$TAT_F = ((\sum_{i=0}^{FSCL-1} h_i) + 1) * (d_2 + 1) \qquad (5)$$

FSR scan designed circuit is better than scan designed circuit in test application time if $TAT_S - TAT_F > 0$. For example, in Figs. 6 and 7, the scan designed circuit and the FSR scan designed circuit after DFT augmentation is done on the circuit $S_C$ in Fig. 5. The scan designed circuit of $S_C$ has a kernel of internally balanced sequential circuit while the FSR scan designed circuit has a kernel of combinational circuit. From equation 4 and 5, test application time for scan designed circuit is 15 clock cycles (for $d_1 = 2$, $SCL = 4$) while test application time for FSR scan designed circuit is 10 clock cycles (for $d_2 = 0$, $FSCL = 4$, $\sum_{i=0}^{FSCL-1} h_i = 9$).

The test application time of the FSR scan designed circuit is less compared to that of the scan designed circuit while the hardware area overhead is same. Therefore, FSR scan design is attractive and worth further studies.

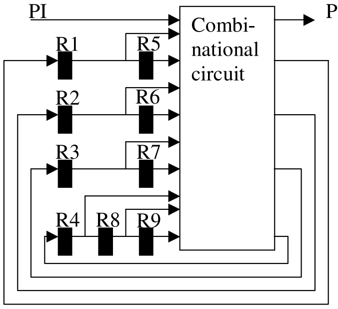

**Fig. 5** Cyclic sequential circuit, $S_C$.

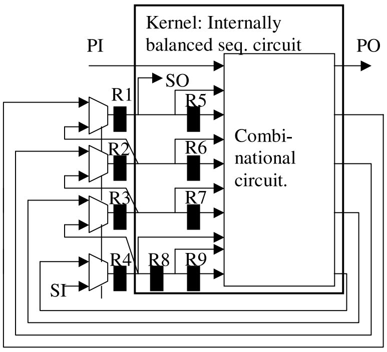

**Fig. 6** Scan designed circuit of $S_C$.

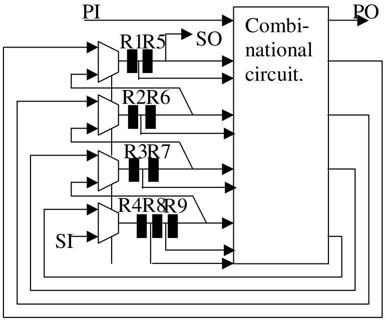

**Fig. 7** FSR scan designed circuit of $S_C$.

## 5. Classes of Easily Testable Sequential Circuits

In this paper, we consider that a class is *easily testable* if its test generation complexity is $\tau^2$-bounded. In other words, $\tau^2$-bounded classes and $\tau$-equivalent classes are easily testable. In this section, we introduce several classes of easily testable sequential circuits. We show that some cyclic sequential circuits are $\tau$-equivalent. Also, we show that synthesis for testability (SFT) can be used to augment a given sequential circuit into an easily testable circuit belonging to one of these classes. We have done case studies to compare the test application time and area overhead of these augmented circuits with their corresponding scan designed 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 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 justification and state differentiation. Logic duplication of the circuit combinational part takes place at every time frame except time frame of derivation of excitation state. 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 with $\tau$-equivalent or $\tau^2$-bounded test generation complexity, which include some cyclic sequential circuits. In such classes, it is guaranteed that any excitation state can be justified and 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 $\tau$-equivalent. Therefore, if the state justification and state differentiation can be reduced to the problem with $\tau^2$-bounded or $\tau$-equivalent or 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

$$\begin{aligned} T_S(n) &\leq T_E(n) + T_J + T_D \\ &= \tau(n) + T_J + T_D \end{aligned} \tag{6}$$

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

### 5.1 $l$-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 section, we introduce 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.

**Definition 7:** 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 $(s_i, s_{if})$, there exists a state differentiation sequence of length at most $l$, where $s_i$ is a

fault-free state and $s_{if}$ is a faulty state corresponding to a fault $f \in F$.

**Theorem 6:** $l$-length-bounded testable circuits is $\tau^2$-bounded if $l$ is $O(n)$, where $n$ is the size of the sequential circuits.

**Proof:** See the detailed proof of theorem in [10]. To generate a test sequence, firstly an excitation state is derived. Secondly, the excitation state is justified and thirdly, the fault effect is propagated to a primary output. Based on condition 1 and condition 2 of definition 7, the test generation complexity for $l$-length-bounded testable circuits is

$$\begin{aligned} T_{LBT}(n) &\leq T_E(n) + T_J(l \times n) + T_D(l \times n) \qquad (7)\\ &= \tau(n) + O(\tau^2(n)) + O(\tau^2(n))\\ &= O(\tau^2(n)), \text{which is } \tau^2\text{-bounded.} \end{aligned}$$

*q.e.d.*

Note that the fault-free state justification is done on the expanded combinational part that consists of $l$ time frames. After fault simulation, we can always obtain the state justification sequence as a prefix of the fault-free state justification sequence. Since the fault-free state justification is a subproblem of test generation of the expanded combinational part, the time complexity of the fault-free state justification is $O(\tau(l \times n))$.

### 5.2 $l$-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 of any length [14]. There are some sequential circuits where not all the states are valid states. Based on this characteristic, we define $l$-length-bounded validity-identifiable circuits.

**Definition 8:** 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 (Fig. 8) 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 ($s_i$, $s_{if}$), there exists a state differentiation sequence of length at most $l$, where $s_i$ is a fault-free valid state and $s_{if}$ is a faulty state corresponding to a fault $f \in F$.

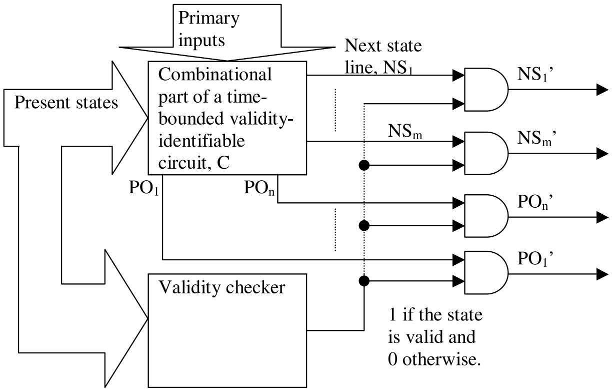
Primary inputs
Next state line, NS1
Present states
Combinational part of a time-bounded validity-identifiable circuit, C
NS1'
NSm
NSm'
PO1
POn
POn'
PO1'
Validity checker
1 if the state is valid and 0 otherwise.

**Fig. 8** Transformed combinational part C′ that consists of C embedded with a validity checker.

**Theorem 7:** The class of $l$-length-bounded validity-identifiable circuits is $\tau^2$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 in a $l$-length-bounded validity-identifiable circuit, firstly a valid excitation state is derived at time frame 0. From condition 1, the excitation state is always guaranteed to be valid by embedding a validity checker in the combinational part of the sequential circuit as shown in Fig. 8. This transformed combinational part is denoted by $C'$ in the following text. Note that the validity checker is not a DFT method. A fault is testable in $C$ with a valid state if and only if the fault is testable in $C'$. Secondly, the state justification is performed and lastly the state differentiation is performed. The test generation complexity for $l$-length-bounded testable circuits is

$$\begin{aligned} T_{LBVI}(n) &\leq T_E(2n) + T_J(l \times n) + T_D(l \times n) \qquad (8)\\ &= \tau(n) + O(\tau^2(n)) + O(\tau^2(n))\\ &= O(\tau^2(n)), \text{which is } \tau^2\text{-bounded.} \end{aligned}$$

*q.e.d.*

### 5.3 $t$-Time-Bounded Testable Circuits

We also consider the classification of sequential circuits 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 9:** 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 ($s_i$, $s_{if}$), there exists a state differentiation sequence which can be obtained in time $O(t)$, where $s_i$ is a fault-free state and $s_{if}$ is a faulty state corresponding to a fault $f \in F$.

**Theorem 8:** The class of $t$-time-bounded testable circuits is $\tau$-equivalent ($\tau^2$-bounded) if $t$ is $\tau(n)$ ($\tau^2(n)$), where $n$ is the size of the sequential circuits.

**Proof:** See the detailed proof of theorem in [10]. To generate a test sequence, firstly an excitation state is derived. Secondly, the excitation state is justified and thirdly, the fault effect is propagated to a primary output. From definition 9, the test generation complexity for $t$-time-bounded testable circuits, $T_{TBT}(n)$ is

$$\begin{aligned} T_{TBT}(n) &\leq T_E(n) + T_J + T_D \\ &= \tau(n) + O(t) + O(t) \\ &= \Theta(\tau(n)) \text{ if } t = \tau(n) \text{ or} \\ &\quad O(\tau^2(n)) \text{ if } t = \tau^2(n). \end{aligned} \tag{9}$$

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

*q.e.d.*

### 5.4 *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 time-bounded testable circuits, not all the states of a $t$-time-bounded validity-identifiable circuit are valid states.

**Definition 10:** 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 (Fig. 8) 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 $(s_i, s_{if})$, there exists a state differentiation sequence which can be obtained in time $O(t)$, where $s_i$ is a fault-free valid state and $s_{if}$ is a faulty state corresponding to a fault $f \in F$.

**Theorem 9:** The class of $t$-time-bounded validity-identifiable circuits is $\tau$-equivalent ($\tau^2$-bounded) if $t$ is $\tau(n)$ ($\tau^2(n)$), where $n$ is the size of the sequential circuits.

**Proof:** See the detailed proof of theorem in [10]. To generate a test sequence for a given fault in a $t$-time-bounded validity-identifiable circuit, firstly a valid excitation state is derived at time frame 0. From condition 1, the excitation state is always guaranteed to be valid by embedding a validity checker in the combinational part of the sequential circuit as shown in Fig. 8 such that a fault is testable in $C$ with a valid state if and only if the fault is testable in the transformed combinational part $C'$. Secondly, the state justification is performed and lastly the state differentiation is performed. From definition 10, the test generation complexity for the $t$-time-bounded validity-identifiable circuits, $T_{TBVI}(n)$ is

$$\begin{aligned} T_{TBVI}(n) &\leq T_E(2n) + T_J + T_D \\ &= \tau(n) + O(t) + O(t) \\ &= \Theta(\tau(n)) \text{ if } t = \tau(n) \text{ or} \\ &\quad O(\tau^2(n)) \text{ if } t = \tau^2(n) \end{aligned} \tag{10}$$

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

*q.e.d.*

### 5.5 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, we introduce in this section two special subclasses, namely counter-cycle finite state machine realizations and state-shiftable finite state machine realizations.

#### 5.5.1 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 $\epsilon$ that strongly connects all valid states accordingly in a LFSR counter with a feedback polynomial.
3. Column $\epsilon$ (counter-cycle operation) is realized by AND-OR logic (resp. NOT-OR-AND logic shown in Fig. 10 (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 $\epsilon$;
   - 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.

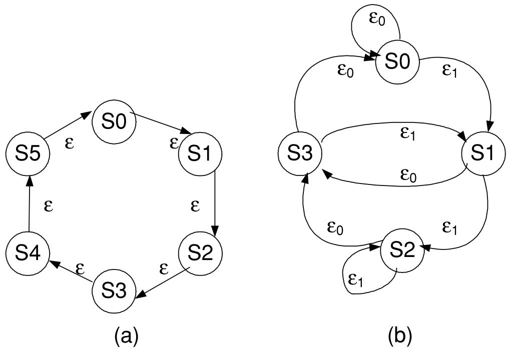

**Fig. 9** (a) State diagrams of counter-cycle FSM and (b) state-shiftable FSM.

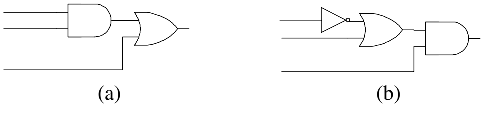

**Fig. 10** (a) AND-OR logic and (b) NOT-OR-AND logic.

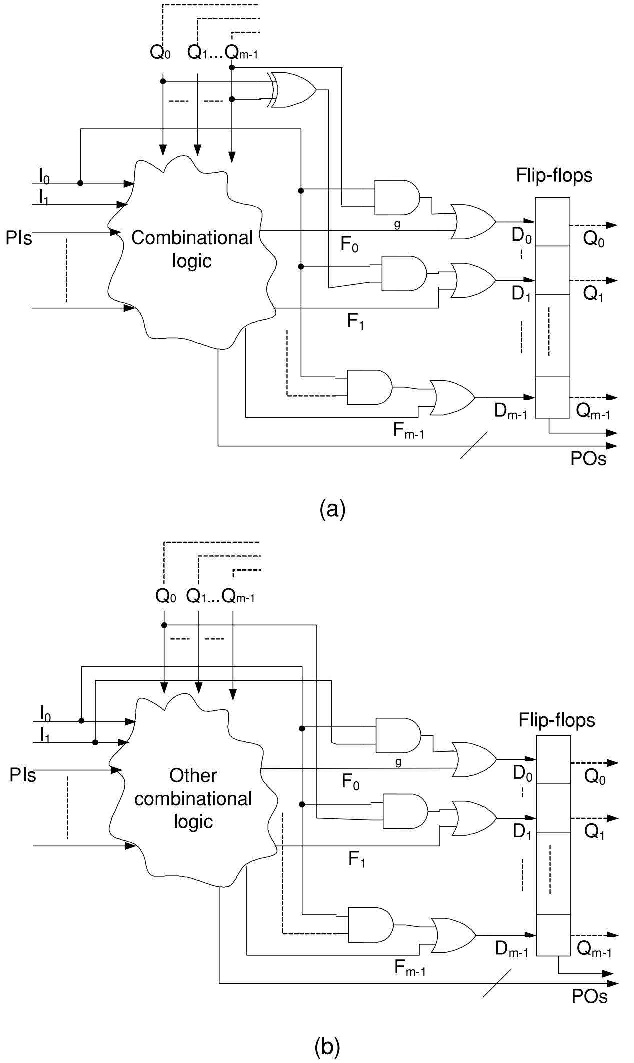

**Fig. 11** (a) Realizations of counter-cycle FSM and (b) state-shiftable FSM with feedback polynomial 0 . . . 11.

Figure 9 (a) shows an example of a counter cycle with 6 states while Fig. 11 (a) illustrates the realization of counter-cycle FSM.

**Theorem 10:** The class of counter-cycle FSM realizations is $\tau$-equivalent.

**Sketch of 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 $\epsilon$ 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 $\epsilon$ 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 $O(n^2)$, where $n$ is the size of the circuit. The detailed proof can be found in [13]. The class of counter-cycle FSM realizations is a subclass of $t$-time-bounded validity-identifiable circuits, where $t = O(n^2)$.

### 5.5.2 State-Shiftable Finite State Machine Realizations

We also introduce another class of easily testable sequential circuits which is state-shiftable FSM. A state-shiftable finite state machine [12] 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 11:** The class of state-shiftable FSM realizations is $\tau$-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 $\epsilon_0$ and $\epsilon_1$, respectively;
2. When input combination corresponds to column $\epsilon_0$ (resp. $\epsilon_1$), 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 $\epsilon_0$ and $\epsilon_1$ (shifting operation) are realized by an AND-OR logic (Fig. 10 (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 $\epsilon_0$ or $\epsilon_1$.
   - 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 Fig. 11 (b). As an example, the transitions happening in columns $\epsilon_0$ and $\epsilon_1$ of the SSFSM of degree 2 are shown in Fig. 9 (b).

**Proof:** The proof is similar to the proof of Theorem 10 by considering the length of justification and differentiation is at most $\log_2 m(= O(n))$ instead of $p(= O(n))$ and values $\epsilon_0$ and $\epsilon_1$ instead of constant value $\epsilon$.

## 6. Synthesis for Testability and Easily Testable Sequential Circuits

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.

### 6.1 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 $\epsilon$ 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 $\epsilon$ 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 $\epsilon_0$ and $\epsilon_1$ 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 $\epsilon_0$ and $\epsilon_1$ as in condition 3 of Theorem 11 and add a primary output to the flip-flop which is the most significant.

### 6.2 Case Study: Comparison between SFTs and DFTs

We performed case studies on five MCNC [15] 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, we compared state-shiftable FSM realizations and counter-cycle FSM realizations with full scan designed circuits. Table 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. We identified that one of the columns of tma corresponded to column $\epsilon_1$ of a state-shiftable FSM. Therefore, we only added one column 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 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, we can see that 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 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 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 |

## 7. Conclusion

$\tau^k$ 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 $\tau$-equivalent while the class of acyclic sequential circuits has been shown to be $\tau^2$-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. We presented 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 realization 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.

## Acknowledgment

This work was supported in part by 21st Century COE (Center of Excellence) Program "Ubiquitous Networked Media Computing" and in part by Japan Society for Promotion of Science (JSPS) under Grants-in-Aid for Scientific Research B(2) (No. 15300018). The authors would like to thank the members of Fujiwara Laboratory who have shared their valuable comments.

### References

[1] P. Goel, "Test generation costs analysis and projections," Proc. 17th DAC, pp.77–84, June 1980.

[2] M.R. Prasad, P. Chong, and K. Keutzer, "Why is ATPG easy?" Proc. 36th DAC, pp.22–28, June 1999.

[3] R. Gupta and M.A. Breuer, "The BALLAST methodology for structured partial scan design," IEEE Trans. Comput., vol.39, no.4, pp.538–544, April 1990.

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

[5] R. Gupta and M.A. Breuer, "Testability properties of acyclic structures and applications to partial scan design," Proc. IEEE VLSI Test Symp., pp.49–54, 1992.

[6] H. Fujiwara, "A new class of sequential circuits with combinational test generation complexity," IEEE Trans. Comput., vol.49, no.9, pp.895–905, Sept. 2000.

[7] R. Gupta and M.A. Breuer, "Partial scan design of register transfer level circuits," JETTA, vol.7, pp.25–46, 1995.

[8] M. Inoue, C. Jinno, and H. Fujiwara, "An extended class of sequential circuits with combinational test generation complexity," Proc. 20th Int. Conf. on Comp. Design, pp.200–205, Sept. 2002.

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

[10] C.Y. Ooi and H. Fujiwara, "Classification of sequential circuits based on combinational test generation complexity," NAIST Information Science Technical Report, NAIST-IS-TR2004001, Jan. 2004.

[11] T. Inoue, T. Hosokawa, T. Mihara, and H. Fujiwara, "An optimal time expansion model based on combinational ATPG for RT level circuits," IEEE the 7th ATS, pp.190–197, Dec. 1998.

[12] H. Fujiwara, Logic Testing and Design for Testability, The MIT Press, 1985.

[13] C.Y. Ooi and H. Fujiwara, "Some $\tau$-equivalent classes of sequential circuits," NAIST Information Science Technical Report, NAIST-IS-TR2004002, June 2004.

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

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

[16] E. Gizdarski and H. Fujiwara, "SPIRIT: A highly robust combinational test generation algorithm," IEEE Trans. Comput-Aided Des Integra. Circuits Syst., vol.21, no.12, pp.1446–1458, Dec. 2002.

**Chia Yee Ooi** received the B.E, and M.E. degrees in electrical engineering from Universiti Teknologi Malaysia, Johor, Malaysia, in 2001 and 2003, respectively. Presently, she is a Ph.D student at Fujiwara Laboratory, Nara Institute of Science and Technology. Her research interests are VLSI design and testing. She is also a student member of IEEE.

**Thomas Clouqueur** received his Engineering degree in Electrical Engineering from the Ecole Superieure d'Electricite, France in 1999, MS and PhD in Electrical and Computer Engineering from the University of Wisconsin, Madison, USA, in 1999 and 2003. He is currently a COE postdoctoral fellow at the Nara Institute of Science and Technology, Nara, Japan. His research interests include VLSI design and testing, and fault-tolerant computing.

**Hideo Fujiwara** received the B.E., M.E., and Ph.D. degrees in electronic engineering from Osaka University, Osaka, Japan, in 1969, 1971, and 1974, respectively. He was with Osaka University from 1974 to 1985 and Meiji University from 1985 to 1993, and joined Nara Institute of Science and Technology in 1993. Presently he is a Professor at the Graduate School of Information Science, Nara Institute of Science and Technology, Nara, Japan. His research interests are logic design, digital systems design and test, VLSI CAD and fault tolerant computing, including high-level/logic synthesis for testability, test synthesis, design for testability, built-in self-test, test pattern generation, parallel processing, and computational complexity. He is the author of Logic Testing and Design for Testability (MIT Press, 1985). Dr. Fujiwara is a fellow of the IEEE, a Golden Core member of the IEEE Computer Society, and a fellow of the Information Processing Society of Japan.