1997

Synthesis for Testability by Two-Clock Control

Shashank K. Mehta
*Pune University, India, skmehta@cse.iitk.ac.in*

Sharad C. Seth
*University of Nebraska - Lincoln, seth@cse.unl.edu*

Kent L. Einspahr
*Concordia College, eins@seward.ccsn.edu*

Follow this and additional works at: [http://digitalcommons.unl.edu/cseconfwork](http://digitalcommons.unl.edu/cseconfwork)

Part of the [Computer Sciences Commons](http://digitalcommons.unl.edu/compsci)


[http://digitalcommons.unl.edu/cseconfwork/43](http://digitalcommons.unl.edu/cseconfwork/43)

This Article is brought to you for free and open access by the Computer Science and Engineering, Department of at DigitalCommons@University of Nebraska - Lincoln. It has been accepted for inclusion in CSE Conference and Workshop Papers by an authorized administrator of DigitalCommons@University of Nebraska - Lincoln.
Synthesis for Testability by Two-Clock Control

Shashank K. Mehta
Pune University, India
skm@cs.unipune.ernet.in

Sharad C. Seth
Dept. of CSE
Univ. of Nebraska-Lincoln
Lincoln, NE 68588-0115
seth@cse.unl.edu

Kent L. Einspahr
Dept. of Comp. Science
Concordia College
Seward, NE 68434-1599
eins@seward.ccsn.edu

Abstract

In previous studies clock control has been inserted after design to improve the testability of a sequential circuit. In this paper we propose a two-clock control scheme that is included as a part of the logic synthesis of a finite state machine (fsm). The scheme has low area overhead and competes well with scan methods in its ability to initialize and observe circuit states. The states of the machine are assigned a pair of binary values using a novel split coding system. The purpose of the encoding is to ease navigation between any pair of states using a combination of normal and test-mode transitions. We require a Hamiltonian cycle to exist in the state transition graph. Our investigation of the fsm benchmark shows that either such a cycle already exists or can be created with the insertion of a small number of transition edges. We also present synthesis results to show that the area penalty is small.

1 Introduction

In this paper we consider improving the testability of a synchronous finite state machine (fsm) by clock control. In practice, fsm's appear in stand-alone controller circuits or as parts of data path-controller circuits. In the latter case, the data path usually has good controllability and observability and is easier to test than the controller. For example, the architecture level test generator of Lee and Patel [7] could generate tests for the data-path faults in less than one-third as much time as tests for the controller faults. Further, their tests for the data-path faults had significantly higher fault coverage. This motivates us to consider synthesis-based techniques to improve the testability of fsm circuits.

The basic idea of clock control is to divide FFs in the circuit so that the clock input of FFs in each group could be enabled independently in a test mode. The normal mode is achieved by enabling all groups simultaneously. Figure 1 illustrates an implementation for a two-clock control of the FFs in the circuit. In the \( \phi_1 \) (\( \phi_2 \)) test mode, only the clock to the first (second) group of FFs is enabled, therefore only the selected group of FFs can change their state. This scheme can be generalized to more than two clocks by splitting the system clock in the test mode into more than two lines.

If the original circuit is viewed as a fsm the test modes provide extra state transitions absent in the original machine, potentially simplifying navigation between states. This is illustrated in Figure 2 for the state transition graph of a modulo-8 counter where the left-most bit is controlled by \( \phi_1 \) and the other two bits are controlled by \( \phi_2 \). New transitions added by clock control are shown by appropriately labeled dotted edges and occur anytime a normal transition causes a change in both groups of bits, e.g., from state 011 to 100. Significantly, the new transitions originate by the restriction of the normal (functional) transitions to enabled clock groups, hence they do not involve extra logic.

As is evident from Figure 1, the area overhead in implementing clock control is quite small with the primary contribution coming from test-mode pin(s) and the steering logic shown in the figure. Unlike scan, there is no requirement for global signal routing. Fur-
thermore, because the extra logic does not appear in the functional logic, there is no time penalty.

The implementation in Figure 1 has negligible delay penalty but may not be readily acceptable to designers because it requires modification to the clock distribution logic. A logically equivalent implementation that avoids this problem but may add some area and delay costs is possible by adding a 2-input mux at the input of each FF. Under a clock-enable control the mux can either steer the current value of the FF or a new value from the data input. The clock-enable line is then used to group the FFs for two-clock control. With modern synthesis tools, the area and time penalties due to the mux can be minimized by considering it as part of the functional logic before optimization.

The original approach to clock control [1] suggested static grouping of the FFs for n-clock control and outlined a multi-dimensional extension of the time-frame-based algorithm for test generation. The approach has since been extended in several different ways by us and others. Cycles and long paths in the state-transition graph can cause excessive search in a time-frame based test generator [3], hence we suggested a FF grouping scheme to reduce the number of cycles and the path lengths in each partition [4]. More recently, the clock-control idea has been extended to include dynamic grouping of FFs [2], delay testing [6], and control of difficult-to-reach states in test generation [5, 10].

In all of the previous works, clock control is added as a post-design step to a given circuit to improve its controllability. (This is also true of scan methods; unlike scan, clock control invariably requires analyzing the topology or the function of the circuit.) Some methods also required the use of multiple clocks. The purpose of the present work is to incorporate testability in an earlier step in the design of a fsm to suggest a state assignment which results in a design that is easy to control and observe. After state assignment, standard synthesis tools can be used to optimize the circuit implementation. Remarkably, we show that with just two clocks it is possible to achieve state controllability and observability comparable to that by scan.

Due to space limitations, all proofs are omitted from this paper but are presented in [9].

## 2 Split Coding System

Let \([n]\) denote the set \(\{0, \ldots, n-1\}\). The coding \(N : [n] \rightarrow [m] \times [2^k]\) is inductively defined as follows:

1. \(N(0) = (\alpha_0, \beta_0) = (0, 0)\)
2. If \(N(j) = (\alpha_j, \beta_j)\), then \(N(j + 1) = (\alpha_j + 1 \text{ mod } m, \beta_j + 2^{\alpha_j} \text{ mod } 2^k)\)

Since \(N(j + m \cdot 2^k) = N(j)\) for all \(j\), the index set of \(N\) is defined as \(\{0, 1, \ldots, m \cdot 2^k - 1\}\).

**Notation:** Arithmetic operations (plus, minus) on the index set and on the first and second components will be \(\text{modulo } m \cdot 2^k, \text{modulo } m, \text{ and modulo } 2^k\), respectively, except when mentioned otherwise.

**Notation:** An arbitrary code word will be denoted by \((a, b)\) or \((\alpha_i, \beta_i)\).

**Definition:** The difference between \(N(i)\) and \(N(j)\) will be denoted as \(\Delta(N(j), N(i))\), and given by \(j - i\).

**Example 1:** The 12 distinct code words of the split code for \(m = 3\) and \(k = 2\) is shown in Table 2. Normally, successive code words differ in both the first and second values. The exception occurs whenever \(N(j - 1) = (a, b)\) and \(2^a \geq 2^k\). Then, the next code only changes in the first value.

## 3 Basic Properties

We state without proof several properties of the coding system introduced above.

**P1** The code mapping \(N : [m \cdot 2^k] \rightarrow [m] \times [2^k]\) is a bijection, i.e. one-to-one and onto.

**P2** If \(N(i) = (\alpha_i, \beta_i)\) and \(N(j) = (\alpha_j, \beta_j) = (\alpha_i, \beta_j)\), then \(j = i + m \cdot (\beta_i - \beta_j)\). That is the difference \(\Delta(N(j), N(i)) = j - i = m \cdot (\beta_i - \beta_j)\).

**P3** \(N(i) = (\alpha_i, \beta_i)\) and \(N(j) = (\alpha_j, \beta_j) = (\alpha_i + 1, \beta_i)\), then \(\Delta(N(j), N(i)) = j - i = m \cdot 2^{\alpha_i} + 1\).

In particular, when \(\alpha_i \geq k\), the difference is 1.
Definition: A sequence of code words $N(i_0), N(i_1), \ldots, N(i_p)$ is monotonic if 
$$\sum_{j=0}^{p-1} \Delta(N(i_{j+1}), N(i_j)) \leq m \cdot 2^k,$$
where the summation is not modulo $m \cdot 2^k$.

Example 2: In the example code shown in Table 2, the sequence $N(5), N(9), N(0), N(2)$ is monotonic but the sequence $N(5), N(0), N(9)$ is not monotonic.

Of particular interest to us are the monotonic sequences in which the successive code words can be reached by a single move as defined below.

Definition: Let $N(j) = (\alpha_j, \beta_j)$ for all $j$ in $[m \cdot 2^k]$. Then the three single moves from an arbitrary state $N(j)$ are:

1. $x$-move: $(\alpha_j, \beta_j) \rightarrow (\alpha_{j+1}, \beta_j)$,
2. $y$-move: $(\alpha_j, \beta_j) \rightarrow (\alpha_j, \beta_{j+1})$, and
3. $xy$-move: $(\alpha_j, \beta_j) \rightarrow (\alpha_{j+1}, \beta_{j+1})$.

Lemma 1 Consider a sequence $S$ of $m + 1$ codewords built from $m$ single moves with the following restriction: each move is either an $x$-move or an $xy$-move. Then $S$ is monotonic.

Corollary There exists a monotonic sequence built from $m$ single moves between two codewords that differ only in the second component.

Theorem 1 For any two code words $N(i)$ and $N(j)$, there exists a monotonic sequence from $N(i)$ to $N(j)$ requiring at most $2m - 1$ single moves.

4 Application to Two-Clock Testability

In this section we show how the split coding system can be employed effectively to solve the state control and observation problems for finite state machines.

4.1 Encoding for Enhanced Control

We adopt a two level encoding scheme for circuit states. First, encode the states of the fsm by split codes, $(a, b)$, and use two clocks $\phi_1$ and $\phi_2$ to control the FFs of $a$ and $b$, respectively. Next, assign optimal binary codes to the two parts using available schemes.

We assume the state-transition graph of the circuit has a Hamiltonian Cycle (HC). Let $S_0, S_1, \ldots, S_{p-1}$ denote successive states of the HC. Select split codes such that $p \leq m \cdot 2^k$. Encode successive states with successive split-codes, i.e., $S_i$ by $N(c + i)$ for $0 \leq i \leq p-1$, where $c$ is any constant. Hereafter, without loss of generality, we will only consider the mapping where $c = 0$.

Navigating Between States: For navigating between the states of the fsm we can exploit the freedom of setting both or either clock to ON. These can be summarized, Figure 3, as follows:

(a) When both clocks are ON, the circuit makes the normal transitions in the HC:
$$N(0) \rightarrow N(1) \rightarrow N(2) \rightarrow \ldots \rightarrow N(p-1) \rightarrow N(0).$$

(b) If only $\phi_1$ is ON the transitions are as follows:

- For $0 \leq j < p - 1$, $N(j) = (\alpha_j, \beta_j) \rightarrow (\alpha_{j+1}, \beta_j)$ and $N(p-1) = (\alpha_{p-1}, \beta_{p-1}) \rightarrow (\alpha_0, \beta_{p-1})$.

(c) If only $\phi_2$ is ON the transitions are as follows:

- For $0 \leq j < p - 1$, $N(j) = (\alpha_j, \beta_j) \rightarrow (\alpha_j, \beta_{j+1})$ and $N(p-1) = (\alpha_{p-1}, \beta_{p-1}) \rightarrow (\alpha_{p-1}, \beta_0)$.

Example 3: Consider a modulo-50 counter for which a transition from state 0 to 49 requires 49 clock cycles in the normal mode. Now, assume the states are encoded by a split code with $m = k = 4$. Suppose we want to go from state $N(0) = (0, 0)$ to state $N(49) = (1, 5) = (01 0101)$. This can be accomplished in 5 clock cycles as follows:
$$\langle 0, 0 \rangle \rightarrow (1, 1) \rightarrow \phi_1 (2, 1) \rightarrow (3, 5) \rightarrow \phi_1 (0, 5) \rightarrow \phi_1 (1, 5)$$
where, the arrows without a subscript denote normal transitions. The transition sequence, being monotonic, does not step out of the machine (functional) states.

The example illustrates the ease with which two-clock control allows navigation between states. In cases in which the number of states $p$ is smaller than the number of code words $(m \cdot 2^k)$, Theorem 1 cannot be applied directly and we must ensure that the machine does not reach an unassigned code-word state in the test mode. The following theorem states a bound on the length of a navigation sequence in which only functional states are reached.

Theorem 2 (Controllability Theorem): Let $N(r)$ and $N(s)$ be codes of any two distinct states in a transition graph with a Hamiltonian cycle. Using two-clock control, without leaving the states of the cycle, it is possible to go from $N(r)$ to $N(s)$ in no more than $2m - 1$ steps if $r > s$ and in no more than $4m - 1$ steps if $r < s$.

Example 4: For the modulo-50 counter, $m = 4$, hence the bounds of the theorem are 7 and 15, respectively.

Selection of Split-Code Parameters: Since the bounds for the length of the transition chain are pro-
Table 2. Split-Code Parameters vs # of States

<table>
<thead>
<tr>
<th># states (p)</th>
<th>n</th>
<th>n-k</th>
<th>k</th>
<th>m</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>40</td>
<td>6</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>1000</td>
<td>10</td>
<td>3</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

portional to $m$, the split-code with minimum $m$ must be used for encoding the states. If the total number of states in HC is $p$ then $n = \lceil \log p \rceil$ bits will be required. Parameters $k$ and $m$ must satisfy the conditions $k \leq m \leq 2^{n-k}$ and $p \leq m \cdot 2^k$. The optimum parameters can be calculated by finding $t$ such that $t - 1 + 2^{t-1} < n \leq t + 2^t$. Then $k = n - t$ and $m = \lfloor p/2^k \rceil$. Split code parameters are given in Table 2 for different numbers of states.

4.2 State Observation

For observing the state of the machine, we add two extra Mealy-type binary outputs. Consider first the simpler problem of initial state identification in which the first component of the code word is known to be zero, i.e., the state to be observed is encoded with a code word $N(i) = (0, \beta)$. Further, assume $\beta[k-1] \ldots \beta[1] \beta[0]$ represents the binary state assignment for $\beta$. We add a binary output $b$ to the circuit and define its value as follows:

For the state $(\alpha, \beta)$, under the normal transition (in the Hamiltonian cycle) from this state, the output $b = \alpha$-th bit of $\beta$.

Now, by observing output $b$ for $k$ successive normal transitions, $b[0], b[1], \ldots, b[k-1]$, where $b[j]$ is the first 1-bit, it can be verified that $\beta = b[k-1] b[k-2] \ldots b[j+1] b[j] b[j-1] \ldots b[0]$, provided that the transition does not involve $S_{p-1} \rightarrow S_0$.

Example 5: In the modulo-50 counter encoding with $m = k = 4$, consider the output from four normal transitions starting with $N(24) = \langle 00, 1010 \rangle$:

$\langle 00, 1010 \rangle \rightarrow \langle 01, 1011 \rangle \rightarrow \langle 10, 1101 \rangle \rightarrow \langle 11, 0001 \rangle \rightarrow \langle 00, 1001 \rangle$

The underlined bit of $\beta$ in each case represents the $\alpha$-th bit that is produced as the $b$ output (shown above the right arrow in each case). It can be seen that the first two output bits match the rightmost bits of the initial $\beta$ and the next two bits of the output are the complement of the corresponding bits of the initial $\beta$.

Now, in the general case, when the start state is $(\alpha, \beta)$ for any $\alpha$ we can deduce $\beta$ as follows: Suppose in the subsequence $b[0], \ldots, b[k-\alpha - 1]$ the first 1-bit occurs at $b[i]$ and in the subsequence $b[k-\alpha], \ldots, b[k-1]$ the first 1-bit occurs at $b[j]$. Then $\beta = b[k-\alpha - 1] \ldots b[i+1] b[i] \ldots b[0] b[k-1] \ldots b[j+1] b[j] \ldots b[k-\alpha]$, provided that the transition does not involve $S_{p-1} \rightarrow S_0$.

We use an additional output bit $a$ to determine $\alpha$.

For the state $(\alpha, \beta)$, under the normal transition (in the Hamiltonian cycle) from this state, the output $a = 0$ if $\alpha = 0$, otherwise, $a = 1$.

With the help of a two bit output, $a$ and $b$, from $k$ successive normal transitions $a_0 b_0, a_1 b_1, \ldots, a_{k-1} b_{k-1}$, the start state $(\alpha, \beta)$ can be determined by the method described above and observing the fact that $\alpha = m - i$ where $a_i = 0$. This leads to the following result:

Theorem 3 (Observability Theorem) The initial state of the machine can be identified by applying a sequence of length $2m$ in the normal mode and observing the outputs $a$ and $b$.

5 Results

We have analyzed the MCNC benchmark circuits to determine which of the state graphs contain Hamiltonian cycles. We have also determined a sufficient number of edges that must be inserted into the state graph to enable a HC for those circuits that do not initially admit such a cycle.

We obtained our results via a program that implemented an enumerative algorithm for finding Hamiltonian circuits in a directed graph [8]. For those state graphs that did not contain Hamiltonian cycles, we modified the program to provide feedback allowing us to incrementally insert appropriate edges until a HC existed. The existence of a HC in the modified graph was subsequently verified.

As illustrated in Table 3, nearly one-third (16 of 53) of the circuits in the MCNC benchmark set contain Hamiltonian cycles. Another 15 of the circuits will contain a HC if only one (well-selected) edge is inserted into the circuit. A summary of the number of edges necessary for a circuit to have at least one HC is given in Table 3. Note that the results given for those circuits requiring insertion of more than one edge are sufficient but may not be minimal.

A comparison of the circuit areas required for four circuits synthesized using our clock control model, using no design-for-testability (DFT) method, and synthesized for full-scan are shown in Table 4. The synthesis results were obtained using SIS [11] and have been normalized with respect to the no-DFT case.
use the cycle index

cover all the states with

nian cycle of normal transitions. One possibility

rable to those required in scan designs.

initialize or observe a state in our scheme are compa-

show that such a cycle either already exists (for

of the circuits) or can be created by the addition of

improves the testability of the synthesized fsm circuit

The synthesis results for the clock-control method

were performed using state assignments obtained from

the split-coding model and using the ordering of states

that provided a HC through the states. Many possible

encodings are possible due to the possible existence of

more than one HC, to using different starting states

in the HC, or to using different contiguous groups of

code words. Future investigation will explore the many

possible encodings for logic optimization. In Table 4, one

circuit, s510, contained an initial HC and required

no additional edges while only one edge was added to

the other three circuits tested (lion9, s208, and s420).

In all cases the area necessary for the clock control

circuits was less than that of the full scan circuit.

6 Conclusion

We have presented a state encoding technique that

improves the testability of the synthesized fsm circuit

through clock control. The current technique requires

that the states of the machine be connected in a single

cycle of transitions. Our results on benchmark circuits

show that such a cycle either already exists (for 30% of

the circuits) or can be created by the addition of a

small number of new transitions. The results also

show that the area penalty over non-DFT circuits is

small and compare favorably with scan realizations.

Furthermore, the number of clock cycles required to

initialize or observe a state in our scheme are compara-

tible to those required in scan designs.

Our future work will evaluate other ways to extend

the method to machines that do not have a Hamilton-

nian cycle of normal transitions. One possibility is to

cover all the states with as few cycles as possible and

use the cycle index as an additional parameter in the

test mode. Another possibility is to cover all the states

with a single cycle that is not necessarily Hamiltonian.

We also plan to develop a test generation algorithm

that can exploit the enhanced testability due to clock

control.

Acknowledgment

Mr. Sangbon Lee's help in producing the SIS results

reported in Section 4 is gratefully acknowledged. The

authors also wish to thank Dr. Vishwani Agrawal for

insightful comments about this work.

References


design for testability and test generation with two clocks.

Proc. 4th Int'l. Symp. on VLSI Design, pages 112–117,


ity combining scan and clock line control and method


method for circuits with feedback. IEEE Trans. Com-


partitioning for testability. Proc. 3rd IEEE Great

Lakes Symp. on VLSI Design, pages 42–46, March

1993.


proving circuit testability by clock control. Proc. 6th

IEEE Great Lakes Symp. on VLSI Design, pages 288–

293, March 1996.


cost DFT methodology for delay testing. Proc. Design


test generator for data path faults and control faults.


hamiltonian circuits in a directed graph. ACM Trans


A new synthesis-for-testability scheme using two-

clock control. Available from the authors or at

http://cse.unl.edu/~seth/pubs/sym_test_bck.html.


ing testability by clock transformation (getting rid of


Apr/May 1996.


R. Margai, A. S. Saldaña, H. Savoj, P. R. Stephan,

R. K. Brayton, and A. Sangiovanni-Vincentelli. SIS:

A system for sequential circuit synthesis. Electron-

ics Research Laboratory Report, Univ. of California,


Table 3. Summary of edge-insertion results.

<table>
<thead>
<tr>
<th># Edges Inserted</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>&gt;13</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td># Circuits</td>
<td>16</td>
<td>15</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>53</td>
</tr>
</tbody>
</table>

Table 4. Comparison of synthesized circuits.

<table>
<thead>
<tr>
<th>Circuit</th>
<th># States</th>
<th>Normalized Area</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>No-DPT</td>
</tr>
<tr>
<td>lion9</td>
<td>9</td>
<td>1.00</td>
</tr>
<tr>
<td>s208</td>
<td>18</td>
<td>1.00</td>
</tr>
<tr>
<td>s420</td>
<td>18</td>
<td>1.00</td>
</tr>
<tr>
<td>s510</td>
<td>47</td>
<td>1.00</td>
</tr>
</tbody>
</table>