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

Proceedings, Tenth International Conference on VLSI Design, 1997.
doi: 10.1109/ICVD.1997.568090

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 ϕ_1 (ϕ_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.

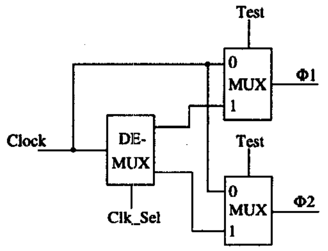

Figure 1. Clock control implementation.

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 leftmost bit is controlled by ϕ_1 and the other two bits are controlled by ϕ_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-

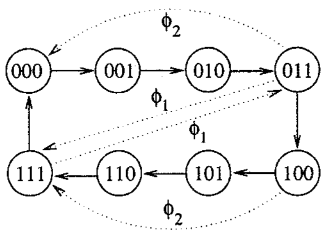

Figure 2. Binary counter state transitions.

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 methods, 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].

Table 1. Split Coding Example

Index	Code Word	Index	Code Word
0	$\langle 0,0\rangle$	6	$\langle 0,2\rangle$
1	$\langle 1,1\rangle$	7	$\langle 1,3\rangle$
2	$\langle 2,3\rangle$	8	$\langle 2,1\rangle$
3	$\langle 0,3\rangle$	9	$\langle 0,1\rangle$
4	$\langle 1,0\rangle$	10	$\langle 1,2\rangle$
5	$\langle 2,2\rangle$	11	$\langle 2,0\rangle$

2 Split Coding System

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

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

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 *modulo* $m \cdot 2^k$, *modulo* m, and *modulo* 2^k, respectively, except when mentioned otherwise.
Notation: An arbitrary code word will be denoted by $\langle a, b \rangle$ or $\langle a_i, b_i \rangle$.
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 $N(j-1) = \langle a, b \rangle$ 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 $N : [m \cdot 2^k] \to [m] \times 2^k]$ is a bijection, i.e. one-to-one and onto.

P2 If $N(i) = \langle \alpha_i, \beta_i \rangle$ and $N(j) = \langle \alpha_j, \beta_j \rangle = \langle \alpha_i, \beta_j \rangle$, 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 $N(i) = \langle \alpha_i, \beta_i \rangle$ and $N(j) = \langle \alpha_j, \beta_j \rangle = \langle \alpha_i + 1, \beta_i \rangle$, 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 $N(j) = \langle \alpha_j, \beta_j \rangle$ for all j in $[m \cdot 2^k]$. Then the three *single moves* from an arbitrary state $N(j)$ are:

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

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, $\langle a, b \rangle$, and use two clocks ϕ_1 and ϕ_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:

$$\begin{tikzcd}
\langle \alpha_{j-1}, \beta_{j-1} \rangle \arrow[r, "\phi_1"] \arrow[d, "\phi_2"'] \arrow[dr, "\text{Normal}"] & \langle \alpha_j, \beta_{j-1} \rangle \\
\langle \alpha_{j-1}, \beta_j \rangle & \langle \alpha_j, \beta_j \rangle
\end{tikzcd}$$

Figure 3. Normal and test mode transitions.

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

(b) If only ϕ_1 is ON the transitions are as follows:
for $0 \leq j < p-1$, $N(j) = \langle \alpha_j, \beta_j \rangle \rightarrow_{\phi_1} \langle \alpha_{j+1}, \beta_j \rangle$
and $N(p-1) = \langle \alpha_{p-1}, \beta_{p-1} \rangle \rightarrow_{\phi_1} \langle \alpha_0, \beta_{p-1} \rangle$.

(c) If only ϕ_2 is ON the transitions are as follows:
for $0 \leq j < p-1$, $N(j) = \langle \alpha_j, \beta_j \rangle \rightarrow_{\phi_2} \langle \alpha_j, \beta_{j+1} \rangle$
and $N(p-1) = \langle \alpha_{p-1}, \beta_{p-1} \rangle \rightarrow_{\phi_2} \langle \alpha_{p-1}, \beta_0 \rangle$.

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 $N(0) = \langle 0, 0 \rangle = \langle 00\ 0000 \rangle$ to state $N(49) = \langle 1, 5 \rangle = \langle 01\ 0101 \rangle$. This can be accomplished in 5 clock cycles as follows:
$\langle 0,0 \rangle \rightarrow \langle 1,1 \rangle \rightarrow_{\phi_1} \langle 2,1 \rangle \rightarrow \langle 3,5 \rangle \rightarrow_{\phi_1} \langle 0,5 \rangle \rightarrow_{\phi_1} \langle 1,5 \rangle$
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 is pro-

Table 2. Split-Code Parameters vs # of States

# states (p)	n	n − k	k	m
5	3	1	2	2
10	4	2	2	3
40	6	2	4	4
1000	10	3	7	8

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 $k \le m \le 2^{n-k}$ and $p \le m \cdot 2^k$. The optimum parameters can be calculated by finding t such that $t - 1 + 2^{t-1} < n \le t + 2^t$. Then $k = n - t$ and $m = \lceil p/2^k \rceil$. 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 $N(i) = \langle 0, \beta \rangle$. Further, assume $\beta[k-1] \dots \beta[1]\beta[0]$ represents the binary state assignment for β. We add a binary output b to the circuit and define its value as follows:

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

Now, by observing output b for k successive normal transitions, $b[0], b[1], \dots, b[k-1]$, where $b[j]$ is the first 1-bit, it can be verified that $\beta = \bar{b}[k-1]\bar{b}[k-2] \dots \bar{b}[j+1]b[j]b[j-1] \dots b[0]$, 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 $N(24) = \langle 00, 1010 \rangle$:

$$\begin{aligned} \langle 0, \beta \rangle \quad = \quad & \langle 00, 101\underline{0} \rangle \xrightarrow{0} \langle 01, 10\underline{1}1 \rangle \xrightarrow{1} \langle 10, 1\underline{1}01 \rangle \xrightarrow{1} \\ & \langle 11, \underline{0}001 \rangle \xrightarrow{0} \langle 00, 1001 \rangle \end{aligned}$$

The underlined bit of β in each case represents the α-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 β and the next two bits of the output are the complement of the corresponding bits of the initial β.

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

We use an additional output bit a to determine α,

> For the state $\langle \alpha, \beta \rangle$, 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 $a_0b_0, a_1b_1, \dots, a_{k-1}b_{k-1}$, the start state $\langle \alpha, \beta \rangle$ 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.

Table 3. Summary of edge-insertion results.

# Edges Inserted	0	1	2	3	4	5	6	>13	Total
# Circuits	16	15	5	6	6	1	1	3	53

Table 4. Comparison of synthesized circuits.

Circuit	# States	Normalized Area No-DFT	Scan	Clock
lion9	9	1.00	1.52	1.22
s208	18	1.00	1.22	1.14
s420	18	1.00	1.22	1.11
s510	47	1.00	1.09	1.01

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 p 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 comparable to those required in scan designs.

Our future work will evaluate other ways to extend the method to machines that do not have a Hamiltonian 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 possiblity 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

[1] V. D. Agrawal, S. C. Seth, and J. S. Deogun. Design for testability and test generation with two clocks. *Proc. 4th Int'l. Symp. on VLSI Design*, pages 112–117, January 1991.

[2] S. Baeg and W. A. Rogers. Hybrid design for testability combining scan and clock line control and method for test generation. *Proc. Int'l. Test Conf.*, pages 340–349, 1994.

[3] K. T. Cheng and V. D. Agrawal. A partial scan method for circuits with feedback. *IEEE Trans. Computers*, C-39:544–548, April 1990.

[4] K. L. Einspahr, S. C. Seth, and V. D. Agrawal. Clock partitioning for testability. *Proc. 3rd IEEE Great Lakes Symp. on VLSI Design*, pages 42–46, March 1993.

[5] K. L. Einspahr, S. C. Seth, and V. D. Agrawal. Improving circuit testability by clock control. *Proc. 6th IEEE Great Lakes Symp. on VLSI Design*, pages 288–293, March 1996.

[6] W.-C. Fang and S. K. Gupta. Clock grouping: A low cost DFT methodology for delay testing. *Proc. Design Automation Conf.*, pages 94–99, 1994.

[7] J. Lee and J. H. Patel. ARTEST: An architecture level test generator for data path faults and control faults. *Proc. Int'l. Test Conf.*, pages 729–738, October 1991.

[8] S. Martello. An enumerative algorithm for finding hamiltonian circuits in a directed graph. *ACM Trans Mathematical Software*, 9:131–138, March 1983.

[9] S. K. Mehta, S. C. Seth, and K. L. Einspahr. A new synthesis-for-testability scheme using two-clock control. Available from the authors or at *http://cse.unl.edu/~seth/pubs/syn_test_2clk.html.*

[10] K. B. Rajan, D. E. Long, and M. Abramovici. Increasing testability by clock transformation (getting rid of those darn states). *Proc. 14th IEEE VLSI Test Symp.*, Apr/May 1996.

[11] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Margai, A. S. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, and A. Sangiovanni-Vincentelli. SIS: A system for sequential circuit synthesis. *Electronics Research Laboratory Report, Univ. of California, Berkeley, CA, No. UCB/ERL M92/41*, May 1992.