Combinational Test Generation for Various Classes of Acyclic Sequential Circuits[*]

Yong Chang Kim[†]
University of Wisconsin-Madison
Madison, WI 53706 USA
kimy@ece.wisc.edu

Vishwani D. Agrawal
CAS Res. Lab, Agere Systems
Murray Hill, NJ 07974 USA
va@agere.com

Kewal K. Saluja
University of Wisconsin-Madison
Madison, WI 53706 USA
saluja@engr.wisc.edu

Abstract

It is known that a class of acyclic sequential circuits called balanced circuits *can be tested by combinational ATPG. The first contribution of this paper is a modified and efficient combinational single fault ATPG method for any general (not necessarily balanced) acyclic circuit. Without inserting real hardware, we create a "balanced" ATPG model of the circuit in which all reconverging paths have the same sequential depth. Some primary inputs are duplicated and each combinational ATPG vector for this model circuit is transformed into a test sequence. Although no time-frame expansion is used, a small set of faults still map onto multiple faults. Those are identified and dealt with again by the single fault combinational ATPG. The results show nearly an order of magnitude or greater saving in the ATPG CPU time over sequential ATPG. The second contribution consists of new partial-scan algorithms to obtain three subclasses of acyclic circuits, namely, internally balanced, balanced, and strongly balanced, which have been described in the literature. Results on ISCAS '89 circuits show that such structures require extra scan overhead, sometimes almost approaching that of full-scan, and their advantages in ATPG are marginal considering the present contribution.*

1 Introduction

For combinational circuits, algorithms and programs exist that provide acceptable fault coverages with 100% fault efficiency. The same performance has not been possible for sequential ATPG, and that is the main reason for the widespread acceptance of the full-scan design [6]. However, concerns about the full-scan overheads of area, delay and test application time have motivated designers and researchers to explore partial-scan techniques [1].

A partial-scan technique, in which scan flip-flops break feedback paths, was proposed by Cheng and Agrawal [4], and Kunzmann and Wunderlich [13]. The resulting acyclic sequential circuit is guaranteed to be initializable and has a well-defined sequential depth that bounds the complexity of sequential ATPG [3]. In spite of an analysis by Miczo [14] showing that the ATPG complexity of acyclic circuits is similar to combinational circuits, and an ATPG method proposed by Kunzmann and Wunderlich [13], combinational ATPG programs have not been applied to acyclic circuits. The basic reasons appears to be as follows:

- One may duplicate the combinational logic as many times as the sequential depth of the acyclic circuit. When the sequential depth is large, as is the case with many real circuits, the test generator must process a very large combinational circuit.
- The time-frame array model requires detection of multiple faults, a condition not handled by existing combinational ATPG programs.

There are several proposals for combinational ATPG for special subclasses of acyclic circuits. We review these subclasses in Section 2. In general, to bring an acyclic circuit to one of these subclasses one requires additional scan flip-flops. We examine the necessary modifications in Section 4 and show that their cost is non-trivial and can be quite high.

Our recent work has been toward finding an efficient method of deriving tests for a general acyclic sequential circuit using any conventional single fault combinational ATPG program. Combining the ideas of Miczo [14] and Kunzmann and Wunderlich [13] with the notion of balanced circuits introduced by Gupta *et al.* [8], we developed a combinational model for ATPG. We were able to detect all testable faults and identify all untestable faults [12]. However, the final objective of deriving the smallest model, especially for circuits with multiple outputs, was not accomplished in the previous work. In Section 3, we present a new and optimum procedure to address this issue.

1.1 Contributions of this Paper

In this paper, we assume that the given sequential circuit is fully synchronous and acyclic. If the circuit is cyclic, it can be made acyclic by scanning a minimum set of flip-flops, known as a *minimum feedback vertex set* (MFVS) [1].

[*]This work was supported in part by the National Science Foundation grant MIP9714034.

[†]Summer intern at Agere Systems, Murray Hill, NJ 07974.

- *An improved test generation algorithm* (Section 3). We derive an algorithm to obtain the smallest combinational ATPG model for generating a test for any fault in a general multi-output acyclic sequential circuit. For the partial-scan ISCAS '89 benchmark circuits, ATPG models generated by our algorithms are 1 to 6 times the original circuit, with average model size being 2.66 times the original combinational logic, where as our previous models were an average size of 2.92 times the original. The revised test generation approach reduces the ATPG time by an order of magnitude and produce vector lengths which are comparable to those obtained by sequential ATPG or other methods.
- *Study of combinationally-testable subclasses* (Section 4). Three subclasses of acyclic circuits, namely, *internally balanced* [7, 9], *balanced* [8], and *strongly balanced* [2] circuits have been defined in the literature (see Section 2) either for combinational ATPG or for reduced test length. In order to assess the scan cost saving provided by our method, we study these subclasses and the full-scan method in Section 4. Since the published literature does not give implementable algorithms, we have provided simple and efficient, though sub-optimum at times, partial-scan algorithms for the three subclasses. We convert the scan-FF selection problem to a minimum cover problem. The ISCAS '89 benchmark results in Section 4 show that the cost of extra hardware required by the subclasses can be significantly large, often approaching that of full-scan.

2 Previous Work

A clocked synchronous circuit is called *acyclic* or *cycle-free* if the flip-flops contain no feedback. The flip-flop structure is often described by the *s-graph* whose vertices are the flip-flops and arcs are combinational signal paths. An acyclic circuit is known to be *initializable* and the test sequence length of any detectable single fault has an upper bound [3]. This upper bound is $d_{max} + 1$, where d_{max} is the *sequential depth* defined as the number of vertices on the longest path in the s-graph. Figure 1 shows a multi-output acyclic circuit that we will use in the subsequent discussion. Its sequential depth is $d_{max} = 2$.

Miczo's 1986 book [14] gives an analysis of acyclic circuits and shows that the problem is similar to the combinational ATPG. Min and Rogers [15] obtain a combinational ATPG model in which all flip-flops are shorted, i.e., replaced with wires or buffers. For a stuck-at fault in this model if a test vector is found by a combinational ATPG program, then they show that the vector repeated d_{max} times will detect the fault in the original circuit. This procedure, however, leaves many faults undetected and one needs a sequential ATPG to achieve 100% fault efficiency.

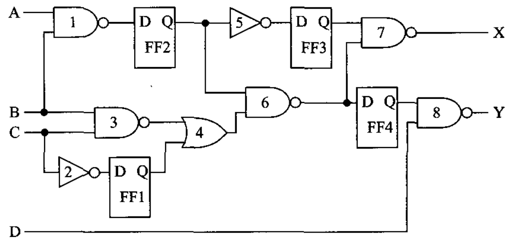


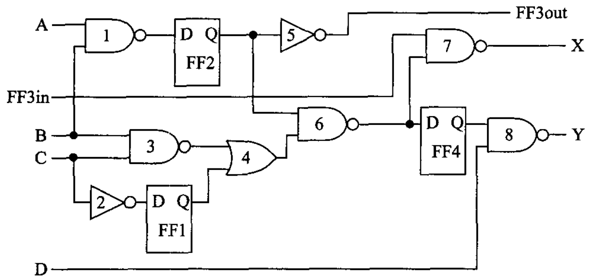
Figure 1. A multi-output acyclic circuit.

Miczo, as well as Kunzmann and Wunderlich [13], point out that an exact combinational ATPG model need not duplicate the entire combinational logic d_{max} times. They, however, did not give any procedure for multi-output circuits. Another problem that remained was that of detecting multiple faults produced even when the logic is selectively duplicated. A solution is provided by the concept of "balanced" circuit introduced by Gupta *et al.* [8]. In a *balanced* (*B*) circuit, all signal paths between any two nodes (inputs, outputs, gates and flip-flops) have the same number of flip-flops. They show that replacing flip-flops with buffers or wires gives a combinational model that can produce tests for all detectable faults. Balanced circuits are a subclass of acyclic circuits. Thus, any acyclic circuit can be converted into a balanced circuit by removing some flip-flops via partial-scan. In Section 4, we will examine the cost of such a modification.

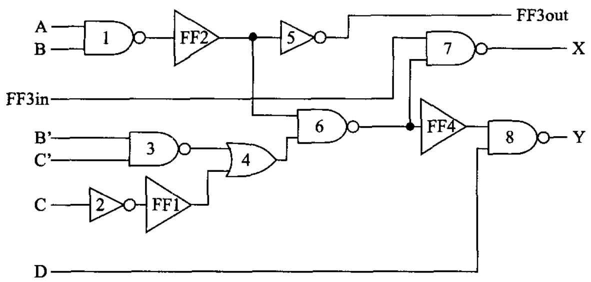
Fujiwara *et al.* [7, 9, 10, 18] introduced another class of acyclic circuits called *internally balanced* (*IB*). In an internally balanced circuit, all node-pairs except those involving a primary input are balanced. This requires lower partial-scan overhead than a balanced circuit. A combinational model is generated by replacing flip-flops with wires or buffers and the primary inputs (PIs) with unbalances are split as additional PIs. Tests are generated for this model using combinational ATPG. Each combinational vector is converted into a test sequence of length $d_{max} + 1$. The bits from a split PI are appropriately placed in the sequence for application to the corresponding original PI. The bits of unsplit PIs are simply duplicated in each vector.

The circuit of Figure 1 has fanouts at PIs B and C and at the output of $FF2$ that reconverge with different sequential depths. The circuit of Figure 2 (a) is obtained by scanning *FF3*. This circuit is internally balanced since only the PI fanouts are unbalanced. The ATPG model is generated by splitting up unbalanced PIs and replacing flip-flops with buffers as shown in Figure 2 (b). A balanced circuit, in which all reconverging fanouts are balanced, is obtained by scanning two additional flip-flops, *FF1* and *FF2*, as shown in Figure 3.

To derive more compact test sequences and to avoid full-scan, Balakrishnan and Chakradhar [2] proposed a *strongly balanced* (*SB*) circuit, which has a more restrictive structure than the balanced circuit. A strongly balanced circuit is balanced and, in addition, all paths between a node and all reach-



(a) Internally balanced circuit.



(b) Internally balanced model.

Figure 2. An internally balanced partial-scan design for the circuit of Figure 1.

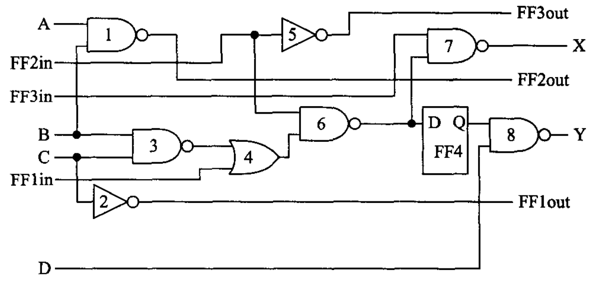


Figure 3. A balanced partial-scan design for the circuit of Figure 1.

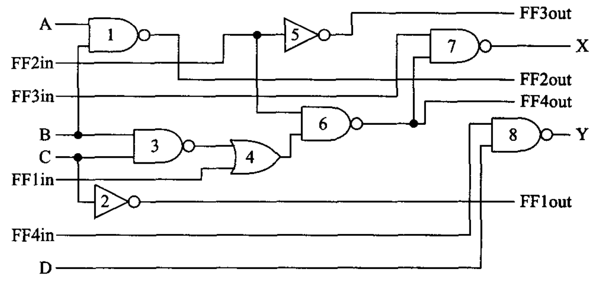


Figure 4. A strongly balanced partial-scan design for the circuit of Figure 1.

able PIs in its fanin cone have the same sequential depth. The additional constraint allows the combinational vectors to be pipelined through the sequential circuit without repeating any vector, except the last one. The test sequence is generated by concatenating combinational vectors and repeating only the last vector $d_{max} + 1$ times. The strongly balanced structure uses a combinational ATPG to derive tests that are significantly more compact than the internally balanced or the balanced circuit, but more flip-flops than the other two subclasses may be scanned to make the circuit strongly balanced. As shown in Figure 4, we scan all four flip-flops, *FF1*, *FF2*, *FF3* and *FF4*, to make the circuit of Figure 1 strongly balanced.

In the full-scan method, all flip-flops are scanned to make the circuit combinational. Starting from a general sequential circuit, we can obtain five classes of circuits by progressively scanning more flip-flops. Each subclass has its own combinational ATPG method. Figure 5 shows the set relationships: Combinational (*Cmb.*) $\subseteq$ Strongly balanced (*SB*) $\subseteq$ Balanced (*B*) $\subseteq$ Internally balanced (*IB*) $\subseteq$ Acyclic sequential circuits (*Acy.*) $\subseteq$ Sequential circuits.

3 The New Combinational ATPG Method

In a recent paper [12], we proposed a method of creating a combinational ATPG model for an acyclic sequential circuit. In our model, all *unbalancedfanouts*, i.e., fanouts reconverging with different sequential depths, are moved toward primary inputs using a retiming-like transformation. Such fanouts can supply different values to branches in different time frames. Unbalanced primary input fanouts are split as additional primary inputs and all flip-flops are shorted. A combinational test vector for a fault in this model is generated using a combinational ATPG and converted into a vector sequence that detects the corresponding fault in the original circuit. Using a fault mapping algorithm, we classified the undetected faults in this model as either untestable or multiply-testable. The latter, typically less than 5% of all faults, are modeled as special single faults in the combinational model. Those procedures produced longer tests than a conventional sequential ATPG and the ATPG model size was also large.

In this section, we propose new methods and algorithms that reduce test sequence length and generate a smaller combinational model for any acyclic sequential circuit.

3.1 Balanced Model (BM) Generation

Consider the circuit of Figure 6. It is acyclic but not balanced, and note that the circuit is different from the one shown in Figure 1. PI B has unbalanced fanouts that reconverge at gate 6 via different sequential depths. Similarly, C and *FF2* have unbalanced reconvergent fanouts to gate 4 and gate 7, respectively. In order to allow combinational ATPG to assign multiple values to a signal like B, we generate a *balanced model* (BM) using Algorithm 1 which provides sepa-

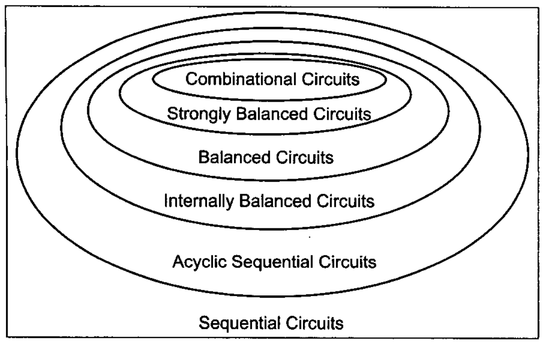

Combinational Circuits
Strongly Balanced Circuits
Balanced Circuits
Internally Balanced Circuits
Acyclic Sequential Circuits
Sequential Circuits

Figure 5. Subclasses of sequential circuits.

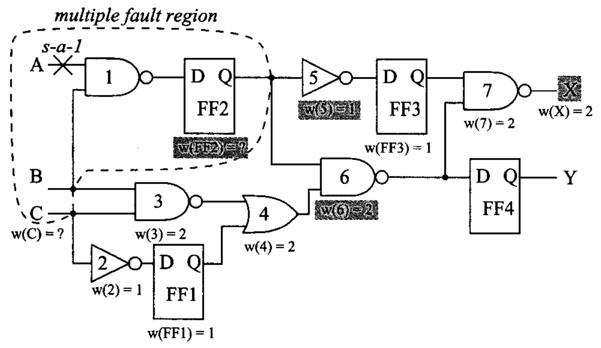


Figure 6. Analyzing an unbalanced circuit.

rate paths for each unbalanced reconvergent fanout.

Algorithm 1 Generating Balanced Combinational Model

- **Assign weights to POs:** *We identify all PIs, gates, flip-flops (FFs) and POs as nodes. In a single pass, assign weights to all PO nodes, where a PO weight is the maximum sequential depth of the PO from all PIs. The weight of a node g is written as $w(g)$. In the next step, weights of non-PO nodes are determined and the balanced model of the circuit is obtained.*

- **Balanced model generation:** *Node weights are determined separately with respect to each PO. A node is* balanced *if weights of fanout nodes are the same with respect to a PO. For each PO, we recursively determine weights of all reachable nodes until every node in the circuit is balanced. Weight assigned to a node is based on the weight of its fanouts reachable to the PO being considered. Two cases can occur as a non-PO node g_i is repeatedly processed:*

Case 1. *Node g_i either has a single fanout node g_j with assigned weight $w(g_j)$, or has multiple fanout nodes among which all nodes with assigned weight have the same value $w(g_j)$. Then, $w(g_i) = w(g_j)$, if g_i is a combinational gate or PI, or $w(g_i) = w(g_j) - 1$, if g_i is a FF.*

Case 2. Duplicate and Split (DAS): *Node g_i has fanouts to multiple nodes g_{jq}, $1 \leq q \leq n$, with n weights. Suppose there are m, $m \leq n$, distinctly different weights among these n nodes.*

Step 1. Grouping fanout weights: *Divide n fanouts to m groups in ascending fanout weight order, $\{w_1, w_2, ..., w_m\}$, where w_1 is smallest among all m fanout weights.*

Step 2. Duplication: *If node g_i has never been duplicated, then g_i is duplicated as m nodes, g_{i2} through g_{im}, each of the same type (i.e., PI, gate, or FF) as g_i. If g_i is a PI, then duplication creates $m - 1$ new PIs. Otherwise, inputs to the duplicated nodes are supplied by adding $m - 1$ new fanouts to each fanin node of g_i. Each duplicated node g_{ik} is assigned with a permanent tag number, called a "difference in depth" (DID), which is defined as $DID(g_{ik}) = DID(g_i) + w_k - w_1$ for $k = \{1, 2, ..., m\}$. DID of a node that has never been duplicated is* 0.

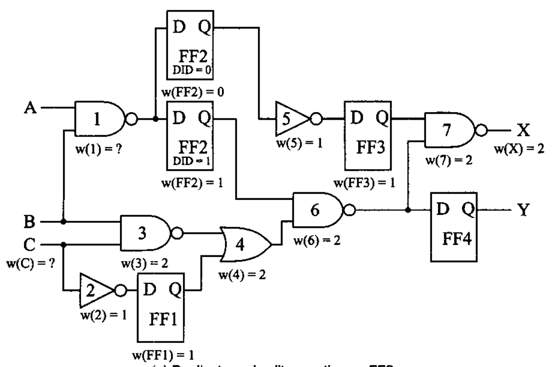
If node g_i has been duplicated while balancing previous POs, make a copy of node g_i if there is no copy of g_i with $DID = DID(g_i) + (w_k - w_1)$, for $k = \{2, 3, ..., m\}$. If node g_i is duplicated l times, where $l \leq m$ and g_i is not a PI node, add $l - 1$ new fanouts to each fanin node of g_i. Each duplicated node is tagged with appropriate DID.

Step 3. Splitting of fanout: *Split and move the fanouts of $m - 1$ groups $\{w_2, w_3, ..., w_m\}$ from g_i to g_{ik}, where $DID(g_{ik}) = DID(g_i) + w_k - w_1$ and $k = \{2, 3, ..., m\}$.*

- **Elimination of FFs:** *Once all POs are processed, the model is balanced and can be converted to a combinational model for test generation by replacing FFs with buffers.*

Algorithm 1 assigns a weight to every node reachable recursively. The recursion over POs leaves each PI with one unique weight with respect to a PO and one unique DID with respect to *all* POs. Case 1 determines the weight of a node with balanced fanouts and Case 2 handles nodes with unbalanced fanouts. For unbalanced node, the *duplicate and split* (DAS) procedure moves unbalanced fanouts one level backward (toward PIs). Successive application of DAS eventually moves all unbalances to PIs, whose splitting creates a perfectly balanced structure.

Let us generate the BM for the circuit in Figure 6. As highlighted in the figure, there is a conflict in weight assignment of *FF2* while balancing the circuit with respect to PO X. Weights on two fanout nodes of *FF2* are $w(5) = 1$ and $w(6) = 2$. Using Algorithm 1, we first group them into two


(a) Duplicate and split operation on FF2.

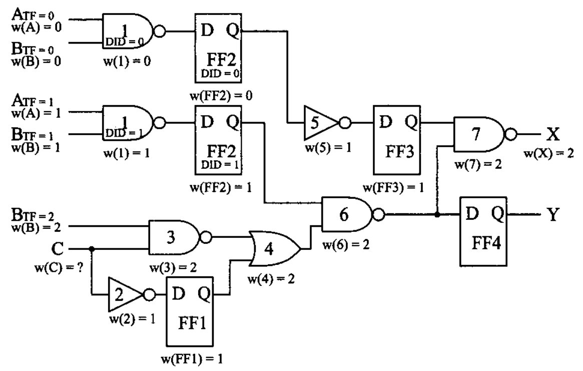

(b) Duplicate and split operation on Gate 1, PI A and B.

Figure 7. Duplicate and split (DAS) operation on FF2 and its fanin nodes.

groups, $\{1, 2\}$. Then, we duplicate the node *FF2* and split the fanout as illustrated in Figure 7 (a). Since no copy of FF2 has been made yet, $DID(FF2) = 0$ and we assign $DID = 0$ to original node and $DID = 1$ to copied node. Using Case 1 of Algorithm 1, the weight of node $FF2_{DID=0}$ is $w(5) - 1 = 0$ and weight of $FF2_{DID=1}$ is $w(6) - 1 = 1$. We recursively assign the weights to the fanin nodes of FF2 and its copy. Figure 7 (b) shows the result of applying DAS on node 1, PIs A and B.

The use of DAS provides separate signal paths in the combinational model so that the copies of a node may assume different values in different time frames. The recursive DAS transformation may create a circuit with new PIs having specific weights. When all nodes are balanced with respect to both POs, X and Y, we replace all FFs with buffers to generate a BM as shown in Figure 8.

3.2 Test Generation

Most faults in an acyclic sequential circuit map onto single stuck-at faults in the model. However, some faults map onto *multiple faults* if DAS transformation has created copies of the targeted faulty location. All copied lines and the original line are a *multiple fault*, i.e., simultaneously occurring

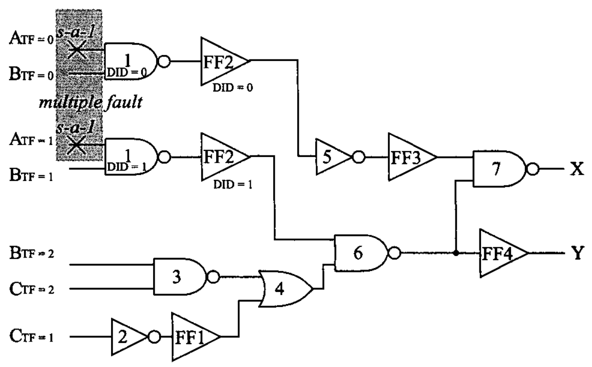


Figure 8. A balanced model for the acyclic circuit of Figures 1 and 6.

faults in the model, and are *modeled* as a single stuck-at fault. Otherwise a fault has a single fault mapping from the sequential circuit to the model. For the circuit shown in Figure 6, any faults in the *multiple fault region*, an area inside of dotted line, are multiple faults. For example, A stuck-at 1 fault is mapped as faults onto $A_{TF=0}$ and $A_{TF=1}$ stuck-at 1 faults in the model as shown in Figure 8. If there is a test for a *mapped* single fault in the BM, then the fault will be detected in the sequential circuit by the sequentialized test. Otherwise, the fault is undetectable in sequential circuit. Similarly, if there is a test for a *modeled* single fault [11] of the multiple fault in the BM, then the fault is detected by the sequentialized test in the sequential circuit. Otherwise, the fault is untestable.

Combinational tests generated with BM must be sequentialized to apply to the sequential circuit. Let d_{max} denote the maximum sequential depth of the original sequential circuit. Weights assigned to PIs in the BM are called *time frames* (TFs). TFs determine the time sequence of signal values at the PI. For example, there are three copies of the PI B of the circuit with DID = 0, 1, and 2 as shown in Figure 8, and they represent B in time frame 0, 1 and 2, respectively. Using the TF on each PI, we can determine how to convert a combinational test vector of BM to a sequential test. Each combinational vector produces a sequence of vectors of length *up to* $d_{max} + 1$. We transform the combinational test vector generated for BM to a vector sequence for the original circuit as follows:

Algorithm 2 Test-Transformation *Suppose that the BM has $k + 1$ copies of a PI node x with TFs of $\{t_0, t_1, ..., t_k\}$, $k \leq d_{max}$, sorted in ascending order, and the values on these $k + 1$ copies in a combinational test derived for BM are $\{v_0, v_1, ..., v_k\}$. The value of the PI x in the t_ith vector of the sequence is v_i. If some value t_j, $0 \leq j \leq k$, does not occur among the TFs of x, then the value of the previous time frame is assigned to the PI x in the t_jth vector. Leading vectors with all Xs (don't cares) are deleted, but the total length of the vectors must be $d_{max} + 1$ to guarantee the detection of the fault.*

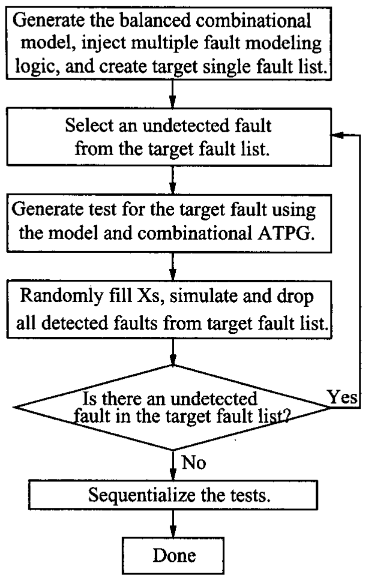

Generate the balanced combinational model, inject multiple fault modeling logic, and create target single fault list.
Select an undetected fault from the target fault list.
Generate test for the target fault using the model and combinational ATPG.
Randomly fill Xs, simulate and drop all detected faults from target fault list.
Is there an undetected fault in the target fault list?
Yes
No
Sequentialize the tests.
Done

Figure 9. Test Generation Method.

In our combinational model, each PI corresponds to an input of the sequential circuit for one or more time frames. Thus, some values of PIs need to be repeated when we sequentialize the combinational test as described in Algorithm 2.

Figure 9 shows a flow chart of our test generation method. First, we generate a fault list for the sequential acyclic circuit. Each fault in sequential circuit maps either a single or a multiple fault. For a multiple fault, we inject a special logic to model it as a single fault, and we generate a corresponding combinational target single fault list for BM [11].

We select an undetected combinational fault from the target fault list, and we generate the test using the BM. The *Xs* in the generated vector are randomly filled with *1/0*, and simulated on BM to drop all detected faults from the target fault list. The *select*, *test generation*, and *random-fill simulate drop-fault* steps are repeated until all faults are detected or classified as untestable in the BM.

Finally, the combinational tests are sequentialized using Algorithm 2.

3.3 Application of BM for test generation

We have implemented the algorithm presented in this paper in C. In our program, a *weighted* directed acyclic graph (DAG) is generated for the circuit, than Algorithm 1 is applied to DAG. The vertices of the DAG are PIs, gates and POs. An arc between a vertex-pair represents a signal flow path and the integer weight of the arc equals the number of FFs on the path.

Table 1 shows the results of experiments on the applicable ISCAS '89 benchmark circuits, i.e., circuits with at least

Table 1. Test generation for acyclic circuits.

Circuit	Combinational ATPG			Sequential ATPG		
Name	FE %	ATPG vectors	TGT s*	FE %	ATPG vectors	TGT s*
s382	100.00	86	0.03	100.00	81	0.18
s400	100.00	89	0.04	100.00	83	0.13
s444	100.00	78	0.05	100.00	77	0.15
s641	100.00	123	0.08	100.00	112	0.34
s713	100.00	126	0.34	100.00	118	1.02
s953	100.00	190	0.15	100.00	182	0.49
s1196	100.00	380	0.54	100.00	304	1.33
s1238	100.00	392	1.11	100.00	327	2.83
s1423	100.00	182	0.53	100.00	182	2.17
s5378	99.71	1230	23.30	99.71	1117	1268.00
s9234	**99.95**	1680	85.68	99.94	1233	425.63
s13207	99.97	2963	54.99	99.97	2442	1008.04
s15850	99.97	3923	140.77	99.97	2507	853.49
s35932	100.00	6542	79.44	100.00	2377	569.07
s38417	99.54	7232	98.17	99.54	5360	860.87
s38584	**99.96**	9721	239.65	99.95	5763	7293.11

* Sun Ultra 10 Workstation

one FF remaining after making them acyclic via partial-scan. For the test generation, we used the *TetraMAX* ATPG system [17] on a Sun Ultra workstation. Table 1 shows the test generation results and gives a comparison between the ATPG using the BM and the conventional sequential ATPG. Fault efficiency (FE), number of sequential ATPG vectors (*ATPG vectors*) and total test generation time (TGT) for both combinational and sequential ATPG are included in the table, under columns *FE, ATPG vectors* and *TGT*, respectively. It is evident that use of ATPG with BM is significantly faster than the sequential ATPG. We observe as much as 53 times reduction in test generation time.

In *all* cases, both new and previous combinational method yielded equal or better fault coverages (FC) and fault efficiencies (FE) then sequential method. The combinational approach gave a higher fault coverage and efficiency (shown in bold) for s9234, as two aborted faults were detected. Higher fault coverage (one more fault detected) was obtained for s38584 in less time than the sequential ATPG. The generated ATPG model is on an average 2.66 times the combinational logic of acyclic sequential circuit. The size of smallest ATPG model was the same as the original combinational logic of acyclic sequential circuit and the largest model was 6.78 times the original. Our previous algorithm [12] generated on an average of 2.92 times the original combinational logic for the partial-scan ISCAS '89 benchmark circuits, where the smallest model was same as the original combinational logic but the largest model was 8.78 times the original. Thus, our new model offers a smaller model than our previous model and it also can be generated in a fraction of TGT. Fault coverages and efficiencies of our new algorithm are equal to the results of our previous algorithm [12]. Thus, new algorithm produces an efficient combinational ATPG model for any general acyclic circuit, requiring no hardware modification to an acyclic circuit. Because of splitting of signals, certain faults in the original circuit map onto multiple faults in BM, but we insert a small logic which allows us to target them as a single fault. Our test generation system obtains a 100% fault efficiency, but the results shown in Table 1 are obtained using default time limits set by TetraMAX. We have also verified our results using Gentest [5] and HITEC [16]. Even though Gentest and HITEC are sequential ATPGs, not optimized for combinational test generation, they have also obtained an average of 3.3 times speed-up (a minimum of 1.1 and a maximum of 5.6 speed-up) using our model on ISCAS '89 benchmark circuits. The test generation times for Gentest and HITEC were within 10% of each other for equivalent coverages.

Our method correctly treats various types of faults, namely, faults detectable by repeating a pattern, faults only detectable by non-repeated patterns, faults only testable as multiple faults in the combinational model, and sequentially undetectable faults.

4 DFT Methods for Balanced structures

For special subclasses of acyclic structure, namely, internally balanced, balanced, strongly balanced and combinational structures, a combinational ATPG model is generated by replacing all FFs with wires or buffers. For full-scan, FF-selection is trivial since *all* FFs in the circuit are scanned. However, for the internally balanced, balanced and strongly balanced structures, scan FF selection is non-trivial. In this section, we present the scan-FF algorithms for various classes. We use a slightly modified s-graph [3], called an *S-Graph*, where each PI, FF and PO is represented as a node, and there is an arc between a pair of nodes if two nodes are connected via a wire or a combinational path. A pair of nodes is said to be *balanced* if all existing paths between two nodes have a same number of FF nodes. If the fanouts of a node reconverge to another node via different sequential depths, we call such fanouts as *unbalanced* fanouts.

Our FF-selection algorithms have three steps, namely, pre-processing, balancing and post-processing steps. During the pre-processing step, we extract an S-Graph of the circuit. During the balancing step, we identify and remove *essential FFs* or *essentials* from S-Graph to reduce problem size. A FF node is *essential* if the successor of this node is same as the successor of its predecessor node. Once we eliminate essential FFs from the S-Graph, we create a set of *target FF lists*, where at least one FF from each list must be scanned to make the S-Graph balanced. Using a minimum cover relation, we select scan FFs to make the S-Graph balanced. Algorithm 3 is a balancing algorithm. For the internal balancing and strong balancing, post processing steps are needed after balancing the S-Graph with the Algorithm 3.

Algorithm 3 Balancing an S-Graph

- **Identify Essentials:** *Given a S-Graph, search all non-PO nodes for essential FFs and delete them from the S-Graph.*

- **Create Target FF Lists:** *For all non-PO nodes with two or more fanouts, check for unbalanced fanouts. For each pair of unbalanced fanouts (reconverge to a node), store names of FFs in both paths to the reconvergent node as a FF list and append to the target FF lists.*

- **Find a Minimum Cover:** *Find a minimum cover of scan FFs for the target FF lists so that at least one FF from each list is scanned.*

4.1 Internally Balanced Structure

If the fanout branches of a PI do not reconverge to any PO via equal sequential depth, they are called *Separable PIs* [7, 9]. These separable PIs can be split and made additional PIs to represent the PIs at different time frames. If the circuit is balanced after separable PIs are split, circuit is said to be internally balanced. Since the internally balanced structure represents all separable PIs as independent PI nodes in S-Graph, an equal or less number of scan FFs are needed than the balanced structure. However, unlike in theory, we learned that separable PIs are very limited in general acyclic circuits. To overcome this problem, we introduce a broader definition to identify all potentially separable PIs, called *combinationally separable PIs*, to split all potentially separable PIs, then selectively either merge some PIs or keep them separated to minimize the scan FFs. To overcome this problem, we introduce a broader definition to identify all potentially separable PIs, then selectively either merge some PIs or keep them separated to minimize the scan FFs. We call such potentially separable PIs as *combinationally separable PIs*.

If a subset of fanouts of a PI converges to a FF or PO node via only combinational paths, we create a new PI called a *combinationally separable PI* and group such subset of fanouts as fanouts of the new PI. Notice that the separable PIs are subset of combinationally separable PIs. If we create a new PI for each combinationally separable PI and move the fanout(s) to the newly created PI, the corresponding S-Graph is called a *S-Graph with combinationally separable PIs*. For internally balanced structure, we balance the S-Graph with combinationally separable PIs, then selectively merge combinationally separable PIs to make them a separable PI or scan additional FFs to keep them separated.

Algorithm 4 Internally Balanced Circuit Generation

- **Pre-processing:** *Construct a modified S-Graph of the circuit with combinationally separable PIs.*

- **Balancing:** *Balance the S-Graph with combinationally separable PIs using Algorithm 3.*

- **Post-processing:** *If a pair of combinationally separable PIs can reach a PO via equal depth paths, we have to either 1) merge them or 2) scan additional FFs to make them separable PIs. Perform 1) or 2) so that the scan FFs are minimized. For 1), additional FFs may need to be scanned if the merged node have unbalanced fanouts. For 2), one of the FFs in equal depth paths to PO must be scanned so no paths with an equal depth to PO exist.*

Once the circuit is scanned to make it internally balanced, a combinational model is generated by creating an additional PI for each separable PI and replacing FFs with the buffers as shown in Figure 2. Model size for internally balanced structure is low since only additional logics are additional PIs. Combinational tests are sequentialized similar to our method in Section 3.2.

4.2 Balanced Structure

Algorithm 5 Balanced Circuit Generation

Pre-processing: *Construct a S-Graph of the circuit.*

Balancing: *Balance the S-Graph using Algorithm 3.*

Post-processing: *No post-processing is required.*

Unlike the internally balanced structure, no post-processing is needed for the balanced structure. Model is generated by replacing FFs with buffers and each combinational test is repeated $d_{max} + 1$ times.

4.3 Strongly Balanced Structure

Algorithm 6 Balanced Circuit Generation

Pre-processing: *Construct a S-Graph of the circuit.*

Balancing: *Balance the S-Graph using Algorithm 3.*

Post-processing: *For each PO, recursively check and scan first FFs in fanin paths to PIs that are longer than the smallest distance to the reachable PI.*

Since the strongly balanced structure requires that the depth of all fanins are same, we process balanced S-Graph to make all fanin paths to be same depth for each PO. Model is generated same way as balanced structure, by replacing all FFs with buffers. Unlike internally balanced or balanced structures, combinational tests need not be sequentialized, except for the last vectors which is repeated $d_{max} + 1$ times.

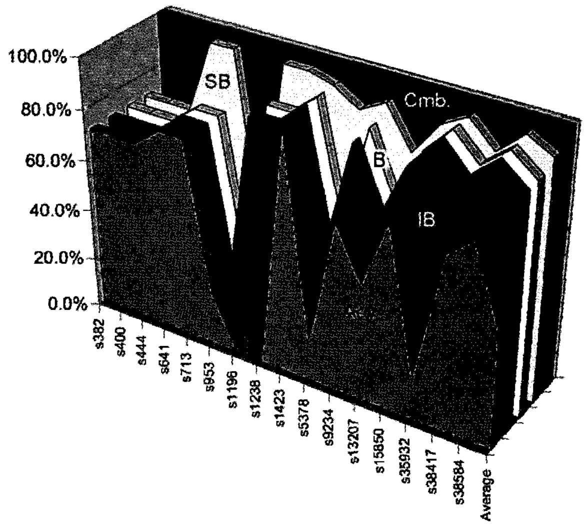


Figure 10. Percentage of partial-scan flip-flops in acyclic subclasses for ISCAS '89 circuits.

5 Results

In this section, we study and evaluate various subclasses of acyclic sequential circuits. Acyclic circuits were generated by scanning MFVS FFs and combinational circuits were generated by the full-scan DFT method. All subclasses structures were derived from the acyclic circuits by scanning additional FFs. A different structure may be obtained via different scan FF selection algorithm, i.e., selecting FFs directly from the original circuit. However, for the purpose of comparison, we use the acyclic circuits as starting point for generating all other subclasses of acyclic circuits. We use our algorithms presented in Section 4 to select scan FFs to generate IB, B and SB circuits. Model generation times for BM, IB, B and SB are negligible since they are only a small fraction of the test generation times.

Figure 10 shows the percent of FFs scanned for the each class of ISCAS '89 circuits . The vertical axis shows the percent of scan FFs and the horizontal axis shows the names of the ISCAS '89 circuits, where the last entry called *Average* shows the *weighted* average of scan FFs for five different classes. Each layer shows the percent of FFs scanned for the acyclic (*Acy.*, IB, B, SB and combinational structures (*Cmb.*, respectively. For s5378 [11], out of 179 FFs, only 30 FFs were scanned to make it acyclic, but IB, B, SB and combinational structures required 91, 96, 163 and 179 scan FFs, respectively. As described in Section 2, more scan FFs are required to make acyclic structure to IB, B, SB and combinational structures. The total number FFs in all circuits is 6729 and 3618 scan FFs (just over 50% of FFs) are required to make them acyclic. Internally balanced and balanced structures require about the same number of scan FFs, 5809 and 5866 (86.3% and 87.2%), respectively. Finally,

Table 2. Test generation time on Sun Ultra 10 for acyclic subclasses of partial-scan ISCAS '89 circuits.

Circuit	Acy.	IB	B	SB	Cmb.
s382	0.03	0.01	0.00	0.01	0.01
s400	0.04	0.02	0.02	0.02	0.02
s444	0.05	0.02	0.02	0.04	0.04
s641	0.08	0.04	0.04	0.04	0.04
s713	0.34	0.21	0.21	0.13	0.13
s953	0.15	0.09	0.09	0.07	0.07
s1196	0.54	0.13	0.13	0.20	0.17
s1238	1.11	0.40	0.40	0.36	0.36
s1423	0.53	0.19	0.19	0.23	0.25
s5378	23.30	0.62	0.61	0.37	0.24
s9234	85.68	66.72	64.65	64.63	64.63
s13207	54.99	21.85	19.99	26.53	26.53
s15850	140.77	115.62	113.75	113.16	112.32
s35932	79.44	70.06	70.04	70.79	70.80
s38417	62.75	24.21	24.21	24.81	24.81
s38584	239.65	30.14	30.17	28.86	27.97
Average	43.1	20.6	20.3	20.6	20.5

the strongly balanced structure requires 6336 FFs (94.2%), which are more FFs then other *balanced* subclasses required but less than the combinational (full-scan) structure. In general, acyclic circuits require significantly less scan FFs than the other classes.

For each class, we generated tests using *TetraMAX* ATPG on the Sun Ultra Sparc workstation. We used the test generation method described in Section 3.3. Table 2 shows the test generation time of ISCAS '89 circuits for each class. We used a time limit per fault chosen by ATPG for acyclic circuits for generating tests for each IB, B, SB and Cmb. circuit. An average test generation time is longer for the circuit with more FFs. Acyclic circuits required 165 seconds of test generation time in average, where as full-scan (combinational) took only 115 seconds. We observed that the test generation time for the acyclic circuit with a larger combinational model [12] requires more test generation time than the purely combinational version of the circuit. The test generation times for IB, B and SB circuits are very close that of the combinational circuits, since the model sizes of IB, B and SB structures are nearly same as original combinational logic.

Figure 11 show the fault coverages of ISCAS '89 circuits for different classes. Notice that the vertical scale in Figure 11 starts from 90%, not 0%. Acyclic circuits using our balanced model obtained nearly same FCs as other methods. Only for some circuits with significantly less scan FFs, such as s5378 and s13207, acyclic circuits structures obtained slightly lower FCs than the other structures, but our method identified all untestable faults correctly in both cases. In all cases, our balanced model obtained 100% fault efficiencies and detected *all* detectable faults and identified *all* untestable faults.

In general, subclasses of acyclic circuits, such as IB, B, SB and combinational, have equal or better FC and FE, and shorter test generation time than the acyclic circuits using a

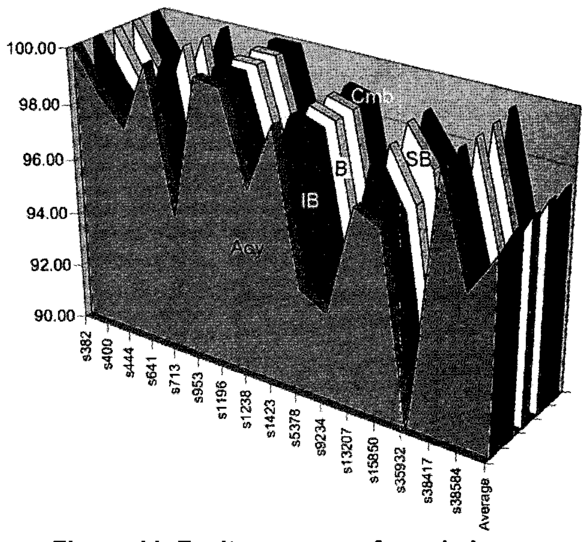


Figure 11. Fault coverages for subclasses.

conventional sequential ATPG.

Finally, Table 3 shows ATPG test vector lengths (*VL*) and clock cycles (CC) required to apply the test. CC is computed using the equation in [3], which includes a length of shift register test. $CC = (n_{ATPG} + 3)ScanFF + 4$, where n_{ATPG} is number of ATPG vectors and $ScanFF$ is number of scan FFs. Each vector length of acyclic circuits is within a magnitude of other subclasses. In general, a test vector length is reduced if more FFs are scanned, but the test application clock cycles will increase since it is a function of scan FFs. For instance, for s5378, 1230, 912, 912, 580 and 580 test vectors are required for acyclic, IB, B, SB and combinational structures, respectively. However, number of test clock cycles increased significantly as we scan more FFs. For s5378, 37K, 83K, 87K, 95K and 104K test clock cycles are required for acyclic, IB, B, SB and combinational structures, respectively.

6 Conclusion

We proposed a new test generation method using a combinational model for a general acyclic sequential circuit. Our model requires only a combinational ATPG to achieve equal or better fault coverage and fault efficiency than using a conventional sequential ATPG. Our test generation method is based on transforming the unbalanced acyclic sequential circuit to a combinational model by moving all unbalanced fanouts to PIs and then adding new PIs. We added PIs allow combinational model to create non-repeated vectors to detect a fault in the original acyclic circuit. Because a combinational model is used to generate tests, the test generation time spent on detectable as well as untestable faults is significantly lower and obtains 100% efficiency. We also developed and presented algorithms for scan FF selection to convert an acyclic circuit to its various subclasses, namely, internally balanced,

Table 3. ATPG test vectors (VL) and test clock cycles (CC) for acyclic subclasses.

Circuit	Acyclic		IB		B		SB		Combinational	
Name	VL	CC	VL	CC	VL	CC	VL	CC	VL	CC
s382	86	1,339	86	1,339	86	1,339	86	1,339	60	1,327
s400	89	1,384	89	1,384	89	1,384	89	1,384	62	1,369
s444	78	1,219	78	1,219	78	1,219	78	1,219	70	1,537
s641	123	1,894	123	1,894	123	1,894	101	1,980	100	1,961
s713	126	1,939	126	1,939	126	1,939	111	2,170	110	2,151
s953	190	1,162	190	1,162	190	1,162	136	1,255	134	3,977
s1196	380	380	312	5,044	312	5,044	201	3,676	200	3,658
s1238	398	398	268	4,340	268	4,340	268	4,882	202	3,694
s1423	182	13,139	182	13,324	182	13,324	174	12,748	172	12,954
s5378	1,230	36,994	912	83,269	912	87,844	580	95,033	580	104,361
s9234	1,680	255,820	1,138	236,191	1,138	238,473	727	160,604	766	175,336
s13207	2,963	919,464	2,328	1,039,630	2,328	1,051,285	1,238	672,626	1,355	908,506
s15850	3,923	1,719,592	1,785	945,856	1,785	954,796	1,192	672,789	1,192	713,419
s35932	6,542	2,002,774	2,319	4,012,420	2,319	4,012,420	2,320	4,014,148	2,319	4,012,420
s38417	7,232	8,067,029	4,863	7,143,292	4,863	7,167,622	3,329	4,918,036	3,384	5,541,136
s38584	9,721	10,842,264	7,722	11,054,479	7,722	11,054,479	3,645	5,278,660	3,627	5,270,764

balanced and strongly balanced circuits.

We compared the hardware and test generation complexity of an acyclic circuit with that of its subclasses, including a combinational (full-scan) circuit, using ISCAS '89 benchmark circuits. Our study concludes that the acyclic circuit in conjunction with our balance model and test generation method are substantially superior in hardware overhead and are comparable in test generation time, test application time and fault efficiency to its various subclasses.

Acknowledgment — We thank A. Balakrishnan for supplying the MFVS for several circuits and Adam Cron, V. L. Narasimha and C. V. Krishna for help with TetraMAX.

References

[1] V. D. Agrawal, editor, *Special Issue on Partial Scan Methods*, volume 7 of *J. Electronic Testing: Theory and Applic.* Boston: Kluwer Academic Publishers, Aug.-Oct. 1995. no. 1/2.

[2] A. Balakrishnan and S. T. Chakradhar, "Sequential Circuits with Combinational Test Generation Complexity," in *Proc. 9th International Conf. on VLSI Design*, Jan. 1996, pp. 111–117.

[3] M. L. Bushnell and V. D. Agrawal, *Essentials of Electronic Testing for Digital, Memory & Mixed-Signal VLSI Circuits.* Boston: Kluwer Academic Publishers, 2000.

[4] K.-T. Cheng and V. D. Agrawal, "A Partial Scan Method for Sequential Circuits with Feedback," *IEEE Trans. Computers*, vol. 39, no. 4, pp. 544–548, Apr. 1990.

[5] W. T. Cheng and T. J. Chakraborty, "GENTEST: An Automatic Test Generation System for Sequential Circuits," *Computer*, vol. 22, no. 4, pp. 43–49, Apr. 1989.

[6] E. B. Eichelberger, E. Lindbloom, J. A. Waicukauski, and T. W. Williams, *Structured Logic Testing.* Englewood Cliffs, New Jersey: Prentice-Hall, 1991.

[7] H. Fujiwara, "A New Class of Sequential Circuits with Combinational Test Generation Complexity," *IEEE Trans. on Computers*, vol. 49, no. 9, pp. 895–905, Sep. 2000.

[8] R. Gupta, R. Gupta, and M. A. Breuer, "The BALLAST Methodology for Structured Partial Scan Design," *IEEE Trans. Computers*, vol. 39, no. 4, pp. 538–548, Apr. 1990.

[9] M. Inoue, E. Gizdarski, and H. Fujiwara, "A Class of Sequential Circuits with Combinational Test Generation Complexity under Single-Fault Assumption," in *IEEE the 9th Asian Test Symposium*, Dec. 2000, pp. 210–215.

[10] M. Inoue, E. Gizdarski, and H. Fujiwara, "Theorems for Separable Primary Input Faults in Internally Balanced Structures," *Information Sceince Technical Report*, vol. ISSN 0919-9527, no. NAIST-IS-TR2000004, pp. 1–5, Mar. 2000.

[11] Y. C. Kim, *Combinational Test Generation Method for Partial-Scan Circuits.* PhD thesis, Department of Electrical and Computer Engineering, University of Wisconsin-Madison, 2001. In preparation.

[12] Y. C. Kim, V. D. Agrawal, and K. K. Saluja, "Combinational Test Generation for Acyclic Sequential Circuits using a Balanced ATPG Model," in *Proceedings of the 14th Int. Conf. on VLSI Design*, Jan. 2001, pp. 143–148.

[13] A. Kunzmann and H. J. Wunderlich, "An Analytical Approach to the Partial Scan Problem," *J. Electronic Testing: Theory and Applic.*, vol. 1, no. 2, pp. 163–174, Apr. 1990.

[14] A. Miczo, *Digital Logic Testing and Simulation.* New York: Harper & Row, 1990.

[15] H. B. Min and W. A. Rogers, "A Test Methodology for Finite State Machines using Partial Scan Design," *J. Electronic Testing: Theory and Applic.*, vol. 3, no. 2, pp. 127–138, May 1992.

[16] T. M. Niermann and J. H. Patel, "HITEC¡: A Test Generation Package for Sequential Circuits," in *Proc. of European Design Automation Conf.*, Feb. 1991, pp. 214–218.

[17] Synopsys, Inc., 700 East Middlefield Rd., Mountain View, CA 94043, *TetraMax ATPG User Guide*, v2000.11 edition, November 2000. Document Order Number: 37043-000 TBD.

[18] T. Takasaki, T. Inoue, and H. Fujiwara, "Paritial Scan Design Methods Based on Internally Balanced Structure," in *Proc. Asia and South Pacific Design Automation Conf.*, Feb. 1998, pp. 211–216.