Efficient test-point selection for scan-based BIST

Article *in* IEEE Transactions on Very Large Scale Integration (VLSI) Systems · January 1999

DOI: 10.1109/92.736140 · Source: IEEE Xplore

4 authors, including:

K.-T. Tim Cheng
University of California, Santa Barbara
477 PUBLICATIONS **12,986** CITATIONS

SEE PROFILE

Chih-Jen Lin
Samsung Electronic
15 PUBLICATIONS **480** CITATIONS

SEE PROFILE

Sudipta Bhawmik
Qualcomm
35 PUBLICATIONS **669** CITATIONS

SEE PROFILE

Efficient Test-Point Selection for Scan-Based BIST

Huan-Chih Tsai, *Student Member, IEEE,* Kwang-Ting (Tim) Cheng, *Member, IEEE,*
Chih-Jen (Mike) Lin, *Member, IEEE,* and Sudipta Bhawmik, *Member, IEEE*

***Abstract*—We propose a test point selection algorithm for scan-based built-in self-test (BIST). Under a pseudorandom BIST scheme, the objectives are 1) achieving a high random pattern fault coverage, 2) reducing the computational complexity, and 3) minimizing the performance as well as the area overheads due to the insertion of test points. The proposed algorithm uses a hybrid approach to accurately estimate the profit of the global random testability of a test point candidate. The timing information is fully integrated into the algorithm to access the performance impact of a test point. In addition, a symbolic procedure is proposed to compute testability measures more efficiently for circuits with feedbacks so that the test point selection algorithm can be applied to partial-scan circuits. The experimental results show the proposed algorithm achieves higher fault coverages than previous approaches with a significant reduction of computational complexity. By taking timing information into consideration, the performance degradation can be minimized with possibly more test points.**

***Index Terms*—Built-in self-test (BIST), test point.**

I. Introduction

RANDOM pattern resistant faults post an obstacle to the success of applying pseudorandom testing to Built-in self-test (BIST). Various works have been proposed to overcome this problem. For example, using weighted random patterns or embedding deterministic test patterns into the pattern generator has been shown to be helpful for detecting random pattern resistant faults [1]–[5]. Alternatively, the circuit under test (CUT) can be modified to be more random testable. This can be done either during the synthesis phase by augmenting the high-level descriptions with test statements or through postsynthesis techniques such as test point insertion (TPI) [6]–[15].

TPI has been studied thoroughly. Several works analyze the structure of CUT to select the test points. In [8], [9], and [13], fault simulation is performed to identify the reconvergent fan-out points and gates which block the activation and propagation of faults. These are classified as good test point candidates. An evaluation step is then applied to select the test point.

Since fault simulation is costly in terms of computational complexity, it may not be practical for large designs. An alternative is to use testability measures. For instance, in [11], COP testability measures [16] are used to select test points. In [17], a metric, U, which is based on COP testability measures has been proposed. U reflects the average expected test length for detecting a fault in a given fault set. Using U, Seiss *et al.* introduced cost reduction factor (CRF) which is an estimate of the reduction of U after inserting a test point candidate. The test point selection is done by: 1) finding a set of potentially good test point candidates using CRF's and 2) computing the actual reduction of U for each selected candidate to identify the test point. We refer to this approach as the CRF-based algorithm. Recently, a divide-and-conquer approach is proposed to partition the entire TPI process into multiple phases [14]. A set of test points are inserted in each phase. Control points are enabled in different phases with a fixed value and shared a common driving source to reduce the area overhead.

An ideal BIST scheme must achieve not only a very high fault coverage with a minimum area overhead but also low or zero performance degradation. Nonetheless, adding test points may cause performance degradation when they are inserted on the timing-critical paths. In this paper, we propose a test point selection algorithm which avoids adding test points on the timing-critical paths, hence the performance degradation is minimized. The selection of test points is based on a novel cost function computed using an efficient hybrid approach. We further propose a symbolic procedure to efficiently compute testability and cost function for a sequential circuit which does not have any global cycle (called *near acyclic circuit*, defined in Section II). With this novel technique, the test point selection algorithm is applicable to partial scan designs.

The rest of this paper is organized as follows. In Section II, an overview of the scan-based BIST architecture, basic testability analysis techniques and the limitations of the CRF-based approach are discussed. The concept of an efficient hybrid approach is described in Section III. An evaluation of the cost function is also presented in Section III. A timing-driven test point selection approach is discussed in Section IV. An extension of the testability and the cost function calculation for partial-scan circuits is given in Section V. Section VI shows the test point selection algorithm and presents the experimental results. In Section VII, a heuristic is proposed to further improve the performance.

II. Backgrounds and Definitions

Consider a directed graph $G = (V, E)$ of a synchronous sequential circuit, where a vertex v_i represents a flip-flop i and a directed edge from v_i to v_j implies that there exists a combinational path from flip-flop i to flip-flop j.

Definition 1: A near acyclic circuit (NAC) is a synchronous sequential circuit whose corresponding directed graph G *does not* contain any cycle with length *greater than one*.

Manuscript received August 14, 1997; revised December 29, 1997.

H.-C. Tsai and K.-T. Cheng are with Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106 USA.

C.-J. Lin was with Bell Laboratories, Lucent Technologies, Princeton, NJ 08542 USA. He is now with Intel Corporation, Hillsboro, OR 97124 USA.

S. Bhawmik is with Bell Laboratories, Lucent Technologies, Princeton, NJ 08542 USA.

Publisher Item Identifier S 1063-8210(98)07558-1.

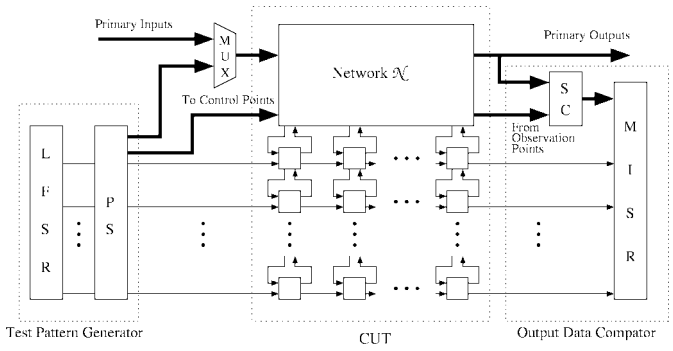

Fig. 1. A scan-based BIST structure.

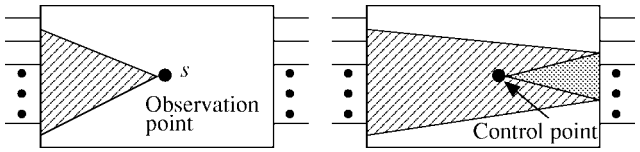

Fig. 2. The effect of a test point: (a) region influenced by an observation point and (b) regions influenced by a control point.

Fig. 1 illustrates the structure of the scan-based BIST [18]. The BIST capability can be incorporated into the circuit by the procedures noted in [18]. For full-scan BIST, all flip-flops are replaced; for partial-scan BIST, flip-flops which will convert the circuit into an NAC after scanning them are replaced with scan flip-flops. We identify these flip-flops using the cycle-breaking algorithm [19]. Test points (control or observation points) can be added to improve the overall random testability.

An observation point is inserted at a node to improve the observabilities of the node and all nodes in its fan-in cone. The effect of inserting an observation point is illustrated by the hatched region in Fig. 2(a). Actual implementation of an observation point is done by simply connecting the node to the output data compactor.

A control point is inserted at a node to improve the controllability as well as the observability of the circuit. Changing the controllability of a node also changes the controllabilities of nodes in its fan-out cone, as indicated in the shaded region in Fig. 2(b). In addition, the observabilities of nodes in the hatched area in Fig. 2(b), which includes the shaded area, are altered. The actual implementation of a control point is a two-input AND gate (0-control point) or a two-input OR gate (1-control point).

A. Random Testability Analysis and Related Works

COP is a well-known procedure to estimate the 1-controllability C_s and observability O_s of every signal s in a combinational network. Both C_s and O_s can be computed efficiently by sweeping the circuit once. C_s and O_s by themselves are not sufficient to guide the selection of test points because they only estimate a local, rather than a global, testability impact when a test point is inserted. A good metric U is defined in [17] as

$$U = \frac{1}{|F|}\left(\sum_{i \in F} \frac{1}{Pd_i}\right) \tag{1}$$

where F is the fault set, $|F|$ is the cardinality of F, and Pd_i is the detection probability of fault i. Under the stuck-at fault model, Pd_i can be estimated by one of following two equations:

$$Pd_{s/0} = C_s \cdot O_s, \qquad \text{for stuck-at-0 fault at } s \tag{2}$$

$$Pd_{s/1} = (1 - C_s) \cdot O_s, \qquad \text{for stuck-at-1 fault at } s. \tag{3}$$

U can be interpreted as the average expected test length for detecting one fault in F using pseudorandom testing because $1/Pd_i$ can be viewed as the expected test length for detecting fault i. Therefore, a smaller U represents a shorter test length which implies that the circuit is more random testable.

Definition 2: The actual cost reduction of a test point s (ACR_s) is the *difference* of the values of U before and after inserting s into the circuit.

ACR reflects the change of the average expected test length. The objective of the test point selection can be restated as finding the test point with the largest ACR and satisfying some timing constraints. One trivial way to find such a test point is to explicitly compute ACR for every possible test point candidate in the circuit but its computational complexity is too high to be practical for large designs. Notice that ACR's computed by this approach are not perfect indicators for selecting test points because 1) COP provides only an estimate of true controllability or observability because of the assumption of signal independence and 2) Pd_i is also an estimate of true detection probability because it assumes the controllability and observability are independent. However, the experimental results show that ACR's computed using this method are reasonably accurate for *large designs* (not necessarily true for small circuits though), therefore, we adopt them as reference points for verifying the quality of our new approach.

To reduce the complexity, Seiss *et al*. proposed a CRF-based algorithm using *cost reduction factor*, an estimated testability improvement [10]. The algorithm is an iterative process. At each iteration, a small set of potentially good candidates is identified using CRF's, then the ACR for every selected candidate is evaluated, finally the candidate with the largest ACR is chosen to be the test point. The algorithm stops if the number of test point inserted reaches a user-specified limit. Using two gradient values G_{C_s} and G_{O_s} of U, CRF's of *all* nodes in the circuit can be computed in linear time with respective to the number of nodes in the circuit. G_{C_s} and G_{O_s} are called controllability gradient and observability gradient of U at signal s, respectively, and they were defined in [17] as

$$G_{C_s} = \frac{\partial U}{\partial C_s} \tag{4}$$

$$G_{O_s} = \frac{\partial U}{\partial O_s}. \tag{5}$$

G_{C_s} and G_{O_s} of *all* nodes can also be computed in linear time using the algorithm given in [17].

Although CRF's can be computed very efficiently, they could significantly deviate from ACR's, especially when the circuit reaches a high fault coverage due to the following assumptions.

A1) The observability changes due to the insertion of a control point is completely ignored. That is, the observability changes of the nodes in the hatched and shaded regions in Fig. 2(b) are neglected.

A2) Assume the CUT has certain types of structures (such as a chain of AND/OR gates) and in (1), consider only hard-to-detect faults affected by a control point.

Ignoring observability changes caused by a control point obviously introduces errors to CRF's. Assuming certain structures in the CUT is generally not true either and could contribute errors, too. Moreover, the accumulative effects of all these assumptions could magnify the inaccuracy of the CRF's.

As a result, the test point candidate with the largest CRF is usually not the one with the largest ACR. Explicitly computing ACR's for a set of candidates with large CRF's is required to find the test point. Because the calculation of ACR's dominates the CPU usage of the CRF-based approach, if the size of selected candidate set is large, it may suffer from a high computational complexity. On the other hand, if the size of the selected candidate set is small, some good candidates may actually be excluded. The overall performance of the CRF-based algorithm is very sensitive to the size of the candidate set. This limit the effectiveness of the CRF-based algorithm.

In the following, we propose a better approximation of the ACR which is computed using an efficient hybrid approach. With this new estimate, test points can be selected without explicitly calculating ACR's, thus, the complexity is reduced.

III. A Hybrid Approach to Test Point Selection

A. Concept

Given a fault set F, the ACR for a test point candidate s can be expressed as

$$\begin{aligned}\mathrm{ACR}_s &= \Delta U^s = U^s - U^{\mathrm{org}} \\ &= \frac{1}{|F|}\left[\sum_{i\in F}\left(\frac{1}{Pd_i^s} - \frac{1}{Pd_i^{\mathrm{org}}}\right)\right] \\ &= \frac{1}{|F|}\left[\sum_{i\in F_1}\underbrace{\left(\frac{1}{Pd_i^s} - \frac{1}{Pd_i^{\mathrm{org}}}\right)}_{\text{the difference is large}}\right. \\ &\qquad \left. + \sum_{i\in F_2}\underbrace{\left(\frac{1}{Pd_i{}^s} - \frac{1}{Pd_i^{\mathrm{org}}}\right)}_{\text{the difference is small}}\right].\end{aligned}$$

U^{org} and U^s are the values of U before and after inserting s, respectively. Similarly, Pd_i^{org} and Pd_i^s are the detection probabilities of fault i before and after inserting s, respectively. We divide F into two subsets F_1 and F_2. For every fault i in F_1, the difference between $1/Pd_i^s$ and $1/Pd_i^{\mathrm{org}}$ is large and for faults in F_2, the differences are small. We will discuss the threshold for determining what is large and what is small later. Because the differences are small for faults in F_2, with a proper use of the gradients of U, we can derive a very good approximation of their contribution to the cost reduction. On the other hand, for faults in F_1, using gradients to estimate the cost reduction may have significant errors because the differences are large and the assumption of gradients is violated. Therefore, explicitly recomputing their new detection probabilities is required. Typically, the size of F_1 is much smaller than that of F_2. Thus, computing new detection probabilities for faults in F_1 will not require much computational effort. As a result, this new hybrid strategy can achieve both high accuracy and low complexity.

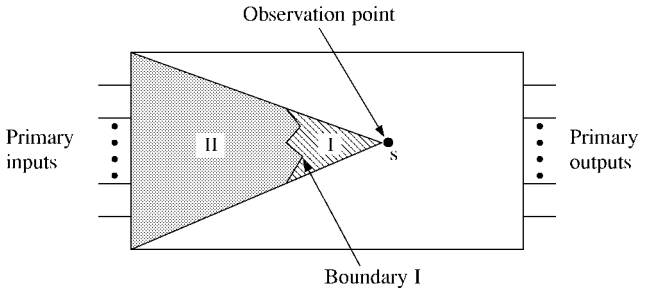

Fig. 3. Computing HCR for an observation point candidate.

An event-driven procedure is used to identify sets F_1 and F_2. This procedure is used to propagate the changes of controllability/observability caused by a test point candidate. The propagation stops if the changes drop below a threshold. Whether an event occurs for further propagation is determined by the *ratio* of $G_{T_i}\cdot\Delta T_i$ to U^{org} (i.e., $(G_{T_i}\cdot\Delta T_i)/U^{\mathrm{org}}$). Here, T_i is either C_i or O_i and ΔT_i is either ΔC_i or ΔO_i for signal i due to the insertion of a test point candidate. This ratio reflects how the value of U is influenced by the ΔT_i change at signal i. Typically, the change of the testability decreases dramatically as propagating for a few logic levels, therefore, the number of signals processed to identify set F_1 is very small and in turns the complexity is low. Notice that F_2 is not found explicitly, its effect is represented by a set of internal nodes as explained later.

B. Cost Function Computation

In this section, we introduce a new cost function called the *hybrid cost reduction* (HCR) which is an estimate of the corresponding ACR. The procedures to compute the HCR for an observation point and a control point are presented as follows.

1) Observation Point: The event-driven propagation of observability changes starts from the observation point candidate s toward the primary inputs. The effect is shown in Fig. 3. For each event associated with a node i, the new observabilty of i is updated and $(G_{O_i}\cdot\Delta O_i)/U^{\mathrm{org}}$ is compared with a given threshold to determine if the observation change at i creates new events. The propagation continues until the event list becomes empty. At the end, nodes denoted as Boundary I in Fig. 3 are identified. They have small observability changes and consequently stop the propagation.

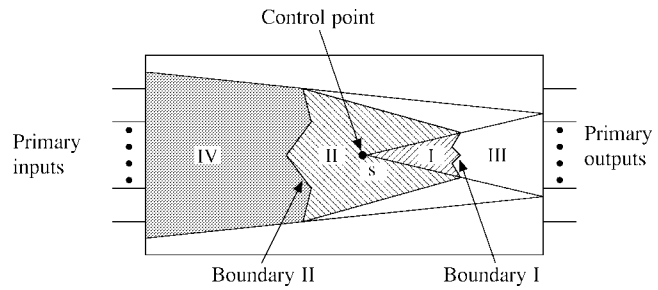

Fig. 4. Computing HCR for a control point candidate.

Because the new observabilities of nodes in region I have been computed, the cost reduction from faults in this region can be found by directly calculating the sum of $1/Pd_i^s - 1/Pd_i^{\text{org}}$ for each fault i. Because nodes on Boundary I have small changes on the observabilities, treating each node j on Boundary I as a pseudo-observation point with a ΔO_j change on its observability, the cost reduction due to the insertion of this pseudo-observation point can be approximated very well by $G_{O_j} \cdot \Delta O_j$. A set of the pseudo-observation points on Boundary I affect the detection probabilities of faults in their fan-in cones, i.e., region II in Fig. 3, therefore, the cost reduction from faults in region II can be estimated by applying the principle of superposition, $\sum_j (G_{O_j} \cdot \Delta O_j)$, for every j on Boundary I, even though their new detection probabilities are not calculated

$$\text{HCR}_s^{\text{OBS}} = \sum_{i \in \text{region I}} \left(\frac{1}{Pd_i^s} - \frac{1}{Pd_i^{\text{org}}} \right) + \sum_{j \in \text{Boundary I}} (G_{O_j} \cdot \Delta O_j) + \left(\frac{1}{Pd_{r/1}} + \frac{1}{Pd_{r/0}} \right). \quad (6)$$

In summary, the HCR for an observation point consists of three parts as shown in (6).

a) $\sum_i (1/Pd_i^s - 1/Pd_i^{\text{org}})$ for every fault i inside region I.
b) For faults in region II, $\sum_j (G_{O_j} \cdot \Delta O_j)$ is used to estimate their impact, where $j \in$ Boundary I.
c) If s is not a fan-out stem in the original circuit, the effect of the two new fan-out branch faults ($1/Pd_{r/1}$ and $1/Pd_{r/0}$) is added, where r is the branch connecting s to the output data compactor.

2) Control Point: Computing HCR for a control point candidate is slightly more complicated. The propagation of new COP values has to proceed in both directions (forward toward the primary outputs and then backward toward the primary inputs). The forward propagation begins with the control point candidate s. During the propagation,$(G_{C_i} \cdot \Delta C_i)/U^{\text{org}}$ is compared to a given threshold to determine if the further propagation is required. Once the forward propagation ends, we obtain the nodes which stop the propagation of controllability changes denoted as Boundary I in Fig. 4. Then the backward propagation starts from them.

During the backward propagation, if $(G_{O_i} \cdot \Delta O_i)/U^{\text{org}}$ is small, then the propagation stops. Eventually another set of nodes indicated as Boundary II in Fig. 4 are identified. They ends the propagation of the observability changes.

The new detection probabilities of the faults in regions I and II are available, their contribution to the cost reduction can be calculated by explicitly finding $1/Pd_i^s - 1/Pd_i^{\text{org}}$ for each fault i. Similar to how nodes on Boundary I in Fig. 3 are treated, we can view every node k on Boundary I as a pseudo-control point with a ΔC_k change on its controllability and then apply the principle of superposition, $\sum_k (G_{C_k} \cdot \Delta C_k)$, to estimate the cost reduction from faults in region III. By viewing every node j on Boundary II as a pseudo-observation point with a ΔO_j change on the observability, $\sum_j (G_{O_j} \cdot \Delta O_j)$ is used to approximate the effect from faults in region IV.

$$\text{HCR}_s^{\text{OR}} = \sum_{i \in \text{regions I and II}} \left(\frac{1}{Pd_i^s} - \frac{1}{Pd_i^{\text{org}}} \right) + \sum_{k \in \text{Boundary I}} (G_{C_k} \cdot \Delta C_k) + \sum_{j \in \text{Boundary II}} (G_{O_j} \cdot \Delta O_j) + \left(\frac{1}{Pd_{s/0}} + \frac{1}{Pd_{t/0}} \right) \quad (7)$$

$$\text{HCR}_s^{\text{AND}} = \sum_{i \in \text{regions I and II}} \left(\frac{1}{Pd_i^s} - \frac{1}{Pd_i^{\text{org}}} \right) + \sum_{k \in \text{Boundary I}} (G_{C_k} \cdot \Delta C_k) + \sum_{j \in \text{Boundary II}} (G_{O_j} \cdot \Delta O_j) + \left(\frac{1}{Pd_{s/1}} + \frac{1}{Pd_{t/1}} \right). \quad (8)$$

Equations (7) and (8) summarize the computation of HCR's for a 1-control point (OR gate) and a 0-control point (AND gate), respectively. Each equation has four components.

a) $\sum_i (1/Pd_i^s - 1/Pd_i^{\text{org}})$ is the cost reduction from faults in region I and II.
b) For faults in region III, $\sum_k (G_{C_k} \cdot \Delta C_k)$ is used to estimate their impact, where $k \in$ Boundary I.
c) Similarly, $\sum_j (G_{O_j} \cdot \Delta O_j)$ is used to estimate the cost reduction from faults in region IV, where $j \in$ Boundary II.
d) The effect of two new input faults of the inserted gate ($1/Pd_{s/0} + 1/Pd_{t/0}$ for an OR gate, $1/Pd_{s/1} + 1/Pd_{t/1}$ for an AND gate, where t is the test input) is added.

It is worth noting that a control point affects the testabilities of the nodes in both its fan-in and fan-out cones. Inserting a set of pseudo-control points on Boundary I in Fig. 4 affects all faults in regions I–IV. That is, $\sum_{k \in \text{boundary I}} (G_{C_k} \cdot \Delta C_k)$ also contains some (small) contributions from faults in regions I, II, and IV. Because the impact from region III usually dominates $G_{C_k} \cdot \Delta C_k$, the error by this approximation is negligible. Furthermore, unlike observation points which always reduce the value of U if they are inserted simultaneously, the effects of control points can cancel one another if they are added at the

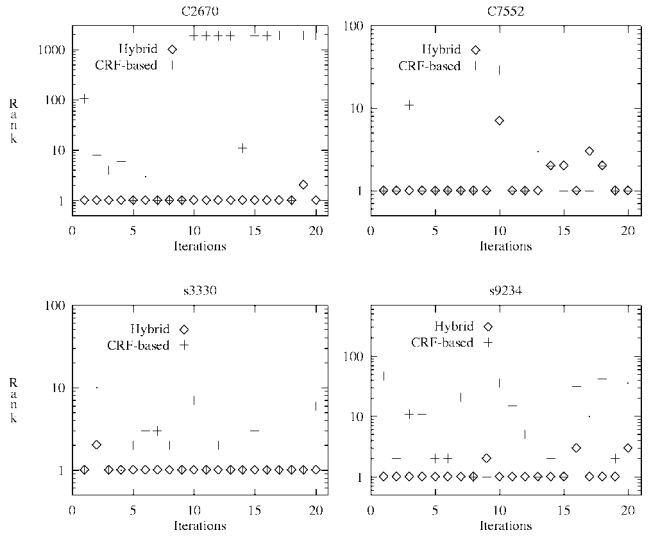

Fig. 5. Evaluation of HCR.

same time. Therefore, applying the principle of superposition to estimate the effects of a set of pseudo-control points is an approximation (while our experimental results shown later demonstrate that this approximation is very accurate).

C. Evaluation

To access the quality of the HCR's, we have conducted an experiment on four benchmark circuits—C2670, C7552, s3330, and s9234. Combinational portions of s3330 and s9234 are used in this experiment. Twenty test points are selected sequentially for each circuit. At each iteration, we explicitly compute the ACR's and HCR's for all possible test point candidates. We then check if the test point candidate with the largest HCR is indeed the one with the largest ACR. We also check the quality of the test point selected using the CRF-based algorithm in term of its *rank* according to the ACR's. The test point of rank one means it is the one with the largest ACR. At the end of each iteration, the test point candidate with the largest ACR is selected and inserted into the circuit. This modified circuit becomes the input netlist to the next iteration. The results are shown in Fig. 5. It shows that the test point candidates with the largest HCR's are almost identical to the ones with the largest ACR's for all cases. On the other hand, the test point selected using the CRF-based algorithm matches well only if it is an observation point. It is very different if a control point is selected. For C7552 and s3330, many observation points are selected using ACR's, thus, the difference between hybrid and CRF-based algorithms is not very significant. However, for C2670 and s9234, the difference becomes obvious because many control points are selected. This experiment shows: 1) the HCR is indeed a very good estimate of the ACR and 2) the assumptions made in computing CRF degrade the effectiveness of the CRF-based approach.

IV. Timing-Driven Test Point Insertion

It is very possible that signals on the timing-critical paths may have a testability problem. The signals on the timing-critical paths tend to be topologically either distant from the primary inputs or from the primary outputs or both and, thus, tend to have poor testabilities than others. However, it is not really necessary to add test points directly on the timing-critical paths. Adding a control point at the transitive fan-ins of a timing-critical net may also help to boost the testability at this net. Similarly, carefully selecting an observation point at a nontiming-critical net will also help to improve the observabilities of the signals on the timing-critical paths.

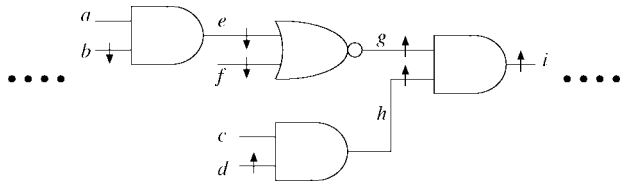

Fig. 6. Adding more test points to resolve the testability problem at a signal on the timing critical paths.

To minimize the performance impact due to the insertion of test point, timing analysis is first performed and the *slack* of each signal is recorded. The *slack* of a signal is defined as the difference of the *required arrival time* and the *actual arrival time* at the signal. All signals with slacks less than a given threshold are excluded from the test point candidate set. Once a new test point is inserted, the slacks of all signals are updated. If the slack threshold is properly chosen, performance degradation can be minimized. Because test points are not permitted in some locations due to the timing constraints, it may be necessary to add more test points to resolve the testability problems at a signal which is on the timing-critical net. Fig. 6 show an example to illustrate this situation. Suppose that controllability at signal i must be increased to improve the circuit testability as indicated by an up arrow. If signals a, c, e, g, h, and i are all on the timing-critical paths, we can still increase the controllability at i by decreasing the controllability at b and f and increasing the controllability at d that are not on the timing-critical paths.

V. Test Point Selection for Partial-Scan Circuits

In an NAC, feedback lines exist in the network so that directly computing COP testability values may require several iterations to converge that could be very time consuming. Therefore, we first propose a symbolic testability computation procedure for NAC's to avoid iterations, then the computation of HCR is extended to partial-scan circuits.

A. Testability Computation

In an NAC, flip-flops are classified as either *self-loop flip-flops* or *nonself-loop flip-flops*. Flip-flops whose corresponding vertices in G involving in cycles of length *one* are called *self-loop flip-flops* (SLFF). Others are called *nonself-loop flip-flops* (non-SLFF).

The NAC is first decomposed into blocks similar that in [18]. A block is defined as a set of logic gates and non-SLFF's feeding a SLFF or a primary output. A block with a SLFF is called a *self-loop flip-flop block* (SLFFB). It consists of a SLFF and a logic cone whose output is the input of the SLFF. An

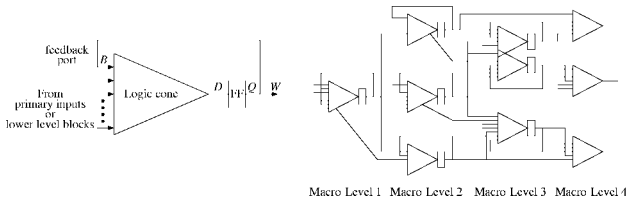

Fig. 7. A decomposed NAC: (a) a SLFFB and (b) an example of NAC after decomposition.

input of the logic cone is either a primary input, the output of the SLFF or the output of another SLFF. The logic cone includes only logic gates and non-SLFF's. Fig. 7(a) shows a SLFFB. Signal B is called a *feedback port* of this SLFFB. A *feedback port* of a SLFFB is an input of a gate inside the SLFFB and is directly connected to the output of the SLFF.

Definition 3: The derived circuit graph, G', of an NAC is a graph where a vertex v_i represents a *self-loop flip-flop* (SLFF), a *primary input* (PI) or a *primary output* (PO), and a directed edge from v_i to v_j implies that there exists a path from SLFF (or PI) i to SLFF (or PO) j.

Furthermore, SLFFB's in an NAC are levelized according to the definition of "macro-level."

Definition 4: The macro-level of a SLFF, a PI or a PO f in an NAC, denoted as $\mathcal{ML}(f)$, is the "node level" of the corresponding vertex in the derived circuit graph, G', where the node level of a vertex f is defined as

$$\mathcal{ML}(f)\begin{cases} 0, & \text{if } f \text{ is a primary input,} \\ 1+\max(\mathcal{ML}(k)), & k \in \text{immediate fan-in of } f. \end{cases}$$

An example of a decomposed and levelized NAC is shown in Fig. 7(b). There are feedback lines for each SLFFB.

1) Computing Controllability: Before starting the controllability computation, the controllabilities at the inputs of the target block (could be either the PI's or the outputs of lower macro-level SLFFB's) must have been computed. A symbolic variable, say x, is assigned to the feedback port B. COP is then used to compute the controllabilities of the internal signals. For those signals which are in the fan-out cone of the feedback port B, their controllabilities will be a polynomial of x. According to [20], any high-order exponents of x should be suppressed to eliminate the inaccuracy caused by the correlation due to the reconvergent fan-out originated from the feedback port variable x.

For example, consider the case shown in Fig. 8(a). The controllabilities at the inputs of the AND gate are $C_1 = ax+b$ and $C_2 = cx+d$, respectively. According to COP, the output controllability $C_o = C_1 \cdot C_2 = acx^2 + (ad+bc)x + bd$. According to [20] , the term of x^2 should be simplified as x to correct the inaccuracy caused by the dependency between these two input signals originating from the feedback port B. This simplification not only increases the accuracy of the controllability measures, but also simplifies the expression of the controllability function of any internal signal as a linear function of x. We also assume that a flip-flop is modeled as a straight line with a data input and a data output and the controllabilities of the data input and the data output are

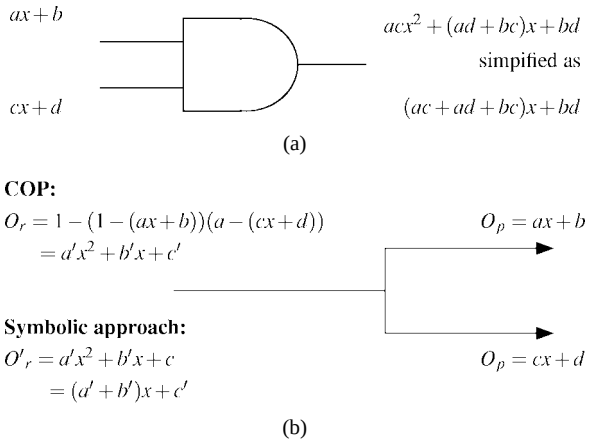

Fig. 8. Computing controllability and observability using symbolic procedures: (a) computing controllability and (b) computing observability.

identical. Therefore, if the controllability at the input, D, of the SLFF is $ax+b$ where a and b are some constants, we can easily derive the controllability at B and in turn those at internal signals in the fan-out cone of B by solving the equation $x = ax+b$.

2) Computing of Observability: To compute observabilities of signals within a SLFFB, we assign the observability of the feedback port B, O_B, a variable x [see Fig. 7(a)]. Since the observability at W should have been computed, we can express the observability at Q (or D) as a linear function of x ($O_Q = 1-(1-O_W)(1-O_B) = ax+b$), then propagate this symbolic value to the block. During the propagation of the symbolic observability value, if a fan-out stem is reached as shown in Fig. 8(b), the observability at the fan-out stem may become a quadratic function of x if both a and c are nonzero numbers. In general, the observability of the internal signal in a SLFFB will be a polynomial of x. Solving the equation of $x = f(x)$ at the feedback port of the block where $f(x)$ is a polynomial of x is a nontrivial task. To maintain the observability of every signal in the block to be a linear function of x, we suppress the second-order exponents as shown in Fig. 8(b). Note that the second-order term arises from the reconvergence of the two fan-out branches. When both coefficients a and c in Fig. 8(b) are nonzero, these two fan-out branches must converge and the value derived by COP is not exact. It is our conjecture that suppressing the high-order exponents will result in more accurate observability values. If we suppress the high-order exponents at the fan-out stem during the computation, the observability function of any signal in the block will be a linear function of x or a constant and is, in general, in the form of $ax+b$. We can prove that, for every signal, 1) both a and b are between zero and one, 2) $a+b$ is also between zero and one and, therefore, (3) $ax+b$ is between zero and one for any value of x between zero and one . Furthermore, in Fig. 8(b), because the coefficient of the quadratic term, $a' = (-ac)$, is a nonpositive number and x is a number between zero and one, O'_r must be smaller than or equal to O_r. Since COP tends to produce an observability value higher than the actual value at the fan-out stem, the suppression technique will produce a more accurate observability value.

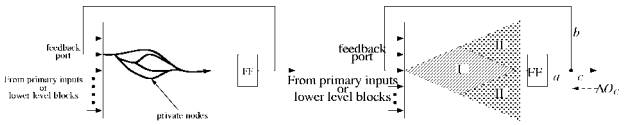

Fig. 9. Computing HCR for an NAC: (a) a SLFFB and its private nodes and (b) propagating observability change to a SLFF.

3) Gradients, G_O and G_C of U: The chain rule formulas for G_O and G_C given in are extended to apply on a symbolic value x and any higher-order exponents of x is suppressed. Then, the procedures of computing controllabilities and observabilities for a SLFFB are also applied to the calculation of G_O and G_C without iterations.

B. HCR Computation

Before the discussion of computing HCR for an NAC, we first define the *private nodes* of a SLFFB.

Definition 5: The *private nodes* of a SLFFB are those which lie on the paths between the SLFF and *feedback ports*.

Lemma 1: In an NAC, a private node of one SLFFB *cannot* be a private node of any other SLFFB.

Fig. 9(a) shows a SLFFB and its private nodes. Before computing HCR's for an NAC, the private nodes of every SLFFB are identified. To propagate the controllability changes, we use the procedure described in Section III with the following exception. That is, if the SLFF of a SLFFB is reached, the event-driven procedure is stopped and the new controllabilities of all private nodes of the SLFFB are calculated. After that, it is not necessary to schedule a private node of the current SLFFB because its controllability is stable. Thus, further propagation won't affect the private nodes of the current SLFFB and in turns no iterative process will incur.

For the propagation of the observability changes, let's consider a SLFFB in Fig. 9(b). Assume that the observability change, ΔO_c, is propagated from signal c and we would like to compute the new observability of signal a. The ideal case is to have the new observabilities of the nodes in region II ready so that we can calculate the new observabilities of nodes in region I using the symbolic procedure. Because the event propagation is done level by level, to achieve this ideal case, we must postpone some events until all events associated with signals in region II are processed. Maintaining such information during the computation of HCR introduces an excessive amount of overheads. Therefore, we use the same procedure as described in Section III except we compute the new observability of a signal only once to prevent the iterative process caused by the feedback lines. This keeps the complexity as low as that of the full-scan case, but it incurs some errors. However, the experimental results show that using this approximation, the $\mathrm{HCR}^{\mathrm{OBS}}$ is still very accurate and the error is negligible.

VI. Implementation and Experimental Results

A. Algorithm

We summarize the test point selection algorithm with the timing-driven option in Algorithm 1. A flag, $timing$, is used to control the algorithm to be either timing-driven or area-driven. If $timing$ is false, then the algorithm is set to area-driven and the objective is to minimize the number of test points. If $timing$ is true, then the algorithm is set to timing-driven and the objective to minimize the performance degradation with as few test points as possible. The algorithm selects the test point sequentially according to the HCR's.

Algorithm 1 Test Point Selection Algorithm
while ($(FC <$ desired $FC) \wedge$ (# of tp $< Max\_tp$)) **do**
  if $timing$ **then**
    compute slacks for all nodes in the circuit;
  end if
  compute the controllability C_s for every node s;
  compute the observability O_s for every node s;
  compute the gradients G_{C_s} and G_{O_s} for every node s;
  for each node s in the circuit **do**
    if $((\neg timing) \vee (Slack_s > Slack_{OBS\_th}))$ **then**
      compute HCR_{O_s};
    end if
    if $((\neg timing) \vee (Slack_s > Slack_{CNTL\_th}))$ **then**
      compute $\mathrm{HCR}_{C_s}^{OR}$;
      compute $\mathrm{HCR}_{C_s}^{AND}$;
    end if
  end for
  Select and insert the test point with the largest HCR;
end while

B. Experimental Results

The algorithm has been implemented and tried on several large ISCAS85 and ISCAS89 benchmark circuits [21], [22] that are random pattern resistant. The circuits were first optimized for performance using Berkeley SIS [23] and mapped into Lucent's 0.9-μm CMOS cell library. SIS is also used for timing analysis (slack computation) assuming that the timing information for each cell is available in the library. The testability computation is performed at the primitive gate level (AND, OR, NAND, NOR, NOT, etc.) while the timing analysis is done at the cell level. We restrict the test points to be only at the boundaries of standard cells but not inside the cells, otherwise, remapping the logic will be required and the timing information derived for guiding the test point selection will be inaccurate. An added control point is mapped into a two-input AND or OR gate. An added observation point is assumed to add an extra load which is equivalent to the load of a flip-flop at the net and, thus, will also cause some extra delay.

Table I shows the results of area- and timing-driven TPI. In this experiment, we set $Slack_{OBS\_th}$ and $Slack_{CNTL\_th}$ to 1 and 3 ns, respectively and also set the threshold for an

TABLE I
RESULTS OF FULL-SCAN CIRCUITS

Circuit	Before TPI		Area-driven TPI				Timing-driven TPI			
	FC[1](%)	Delay (ns)	CP/OP	FC (%)	Delay (ns)	CPU	CP/OP	FC (%)	Delay (ns)	CPU
C2670	83.23	40.6	4/1	98.31	40.6	78 s.	4/1	98.31	40.6	72 s.
C7552	95.12	62.7	3/7	98.23	62.7	3.4 m.	3/7	98.23	62.7	3.1m.
s3330	86.54	13.4	7/6	99.92	14.3	4.9 m.	16/9	99.95	13.4	6.2 m.
s3384	96.82	20.4	0/9	99.78	20.8	128 s.	5/4	99.78	20.4	121 s.
s4863	99.22	27.5	1/1	99.64	27.5	61 s.	1/1	99.64	27.5	50 s.
s9234	87.99	15.1	16/3	96.10	16.1	12.3 m.	20/2	96.13	15.1	10.7 m.
s15850	90.56	24.5	26/8	97.41	29.1	34.4 m.	23/25	97.14	24.5	22.5 m.
s38417	87.37	18.2	30/16	99.19	18.3	1.7 h.	36/12	99.18	18.2	1.9 h.

event propagation in computing HCR to 0.1%. The results of area-driven TPI are shown in Columns 4–6, and those of timing-driven TPI are shown in Columns 7–9. Circuits are fault simulated with 32 K random patterns and the CPU time is measured on a SUN SPARCstation 5. As expected, some circuits suffer from performance degradation with area-driven TPI. The degradation could be as much as 19% (s15850), which is obviously unacceptable. With a few more test points, timing-driven TPI achieves comparable fault coverages and minimizes the performance degradation. Even though the slacks for all signals need to be computed and the number of selected test points may be larger with timing-driven TPI, the run time is not necessarily longer than that of the area-driven TPI. This is because the number of legal test point locations with timing-driven TPI is smaller due to the timing constraints. The results indicate that there is no timing penalty *for circuits we have tested* with timing-driven TPI. As noted in Section IV, it may require more test points to achieve both a high fault coverage as well as low performance degradation using timing-driven TPI. For circuits like s15850, there are lots of signals on the timing-critical paths. Thus, more test points are needed to achieve a comparable fault coverage for timing-driven TPI. On the other hand, if circuits do not have many signals on the timing-critical paths, such as C2670, C7552, and s3384, the performance degradation can be minimized without increasing the number of test points.[1]

The results also indicate that there exists many combinations of test points that achieve comparable fault coverages (e.g., s3384). The issue is how to find a good combination. To demonstrate that the proposed algorithm can find a good combination of test points, we have conducted following experiment.

We randomly select the location and type of a test point. The location is limited to the boundaries of standard cells and every legal location has equal probability to be selected. Each test point type (1-control point, 0-control point, or observation point) also has equal probability to be chosen. For each circuit, the same number of test points is selected using both random and hybrid approaches. Ten random selections are performed for each circuit. Table II shows the comparison of hybrid and the random approaches. The fault coverage is obtained after fault simulated 32K random patterns for each case. The minimal, maximal, and average fault coverages using random selections are shown in Columns 5, 6, and 7, respectively. The experiment reveals that even though random selection has a low complexity, however, the selected test points are not effective.

TABLE II
COMPARISON OF HYBRID AND RANDOM APPROACHES

Circuit	Before TPI(%)	# of TP	Hybrid (%)	Random		
				Min.(%)	Max.(%)	Ave.(%)
C2670	83.23	5	98.31	82.81	84.16	83.31
C7552	95.12	10	98.23	95.36	95.91	95.61
s3330	86.54	13	99.92	86.04	89.20	87.17
s3384	96.82	9	99.78	96.82	98.09	97.08
s4863	99.22	2	99.64	99.22	99.26	99.23
s9234	87.99	19	96.10	87.18	89.85	88.40
s15850	90.56	34	97.41	89.91	91.53	90.75
s38417	87.37	46	99.19	87.42	89.52	88.23

TABLE III
COMPARISON OF HYBRID AND CRF-BASED ALGORITHMS

Circuit	# of TP	Hybrid		CRF-based		CPU ratio(%)
		FC (%)	CPU	FC (%)	CPU	
C2670	5	98.31	78 s.	98.31	264 s.	30
C7552	10	98.23	3.4 m.	97.92	22 m.	15
s3330	13	99.92	4.9 m.	99.39	29 m.	17
s3384	9	99.78	128 s.	99.78	659 s.	19
s4863	2	99.64	61 s.	99.61	201 s.	30
s9234	19	96.10	12.3 m.	94.79	75 m.	16
s15850	34	97.41	34.4 m.	97.08	7.7 h.	7
s38417	46	99.19	1.7 h.	98.84	92.1 h.	2
Ave.	-	98.57	-	98.22	-	17

Furthermore, we compare the hybrid and the CRF-based methods, the results are as shown in Table III. Same number of test points are selected and the location of test points are limited to the boundaries of standard cells. As shown in Table II, The hybrid approach achieves higher fault coverages but requires much less CPU time than the CRF-based approach, especially for large circuits. The reduction of the CPU time mainly comes from eliminating the computation of ACR's for selected candidates. On average, the hybrid

[1] FC = # of detected faults/Total # of faults.

TABLE IV
RESULTS OF PARTIAL-SCAN CIRCUITS

Circuit	# of FFs (total/scan)	Before TPI (%)	Area-driven TPI	
			CP/OP	FC (%)
s1423	74/22	77.6	9/1	96.93
			11/4	99.29
			17/8	99.72
s3330sir	127/8	74.2	1/9	98.47
			3/12	99.59
			9/16	99.94
s3384	183/8	92.6	0/5	98.40
			0/10	98.69
			0/15	98.72
s5378	179/30	90.7	10/0	98.62
			14/1	99.19
			16/4	99.39
s9234.1	183/8	83.9	7/3	97.26
			10/5	98.67
			14/6	99.10
			15/10	99.53
s15850.1	394/23	93.9	7/3	97.70
			16/4	98.67
			22/8	99.19
			28/12	99.32

approach achieves about six times speedup over the CRF-based approach. Both methods can find a relatively good combination of test points, but the proposed hybrid approach has a clear advantage of short CPU time.

Experiments for partial-scan circuits are conducted on several ISCAS89 benchmark circuits. Table IV shows the results. Because circuit s3330 has a large number of redundant faults, we ran a sequential redundancy removal program [24] first. The resulting circuit is named s3330sir. For each circuit, we first use the cycle-breaking program to select a set of scan flip-flops to convert the circuit into an NAC. The second column of Table IV shows the total number of flip-flops in the circuit and the number of scan flip-flops. Circuits are first optimized for area and timing using SIS and mapped into Lucent's 0.9-μm CMOS cell library. Fault simulation with 32 K random patterns is performed for each circuit assuming the initial states of flip-flops are zero. The selected test points significantly increase the fault coverage in each case. Applying proposed algorithm on partial-scan circuits is as effective as it does on full-scan circuits.

VII. HEURISTIC

The efficiency of the algorithm can further be improved using a simple heuristic. As noted in Section II, one dilemma which the CRF-based method has is reducing computational complexity and increasing size of selected candidate set for ACR calculation. The conflict between these two objectives degrades the effectiveness of the CRF-based approach. In Section III, we show that the HCR is a very good alternative of the ACR and also show how it can be computed efficiently. Therefore, in the CRF-based algorithm, instead of calculating ACR's, we compute HCR's for the selected candidates. This can significantly improve the run time efficiency of the CRF-based algorithm. Meanwhile, taking advantage of the efficient computation of the HCR's, we can select more candidates in the first phase to reduce the possibility of excluding good ones. We call this modified CRF-based approach as *enhanced hybrid algorithm.*

TABLE V
RESULTS OF ENHANCED HYBRID APPROACHES (FULL-SCAN)

Circuit	Hybrid			Enhanced hybrid			CPU ratio (%)
	CP/OP	FC (%)	CPU	CP/OP	FC (%)	CPU	
s9234	16/3	96.10	12.3 m.	16/3	96.10	5.4 m.	44
s15850	26/8	97.41	34.4 m.	26/8	97.41	17.7 m.	51
s38417	30/16	99.19	1.7 h.	30/16	99.19	0.62 h.	36

TABLE VI
RESULTS OF ENHANCED HYBRID APPROACHES (PARTIAL-SCAN)

Circuit	Hybrid			Enhanced hybrid			CPU ratio (%)
	CP/OP	FC (%)	CPU	CP/OP	FC (%)	CPU	
s1423	17/8	99.72	17.6 m.	22/3	99.81	1.9 m.	11
s3330sir	9/16	99.94	12.5 m.	13/12	99.94	2.9 m.	23
s3384	0/15	98.72	8.3 m.	0/15	98.72	1.0 m.	12
s5378	16/4	99.39	8.4 m.	16/4	99.39	1.4 m.	17
s9234.1	15/10	99.53	15.7 m.	15/10	99.53	2.1 m.	13
s15850.1	28/12	99.32	53.6 m.	32/8	99.46	3.6 m.	7

We conducted the experiments on large circuits for both full- and partial-scan cases to demonstrate the efficiency of enhanced hybrid algorithm. Tables V and VI show the comparisons of the hybrid and the enhanced hybrid approaches for full-scan and partial-scan circuits, respectively. Both the fault coverage and the CPU time are listed and the ratio of the CPU time of the enhanced hybrid method to that of the hybrid method is also listed in the last column. For full-scan circuits, the same fault coverages are achieved, but the enhanced hybrid method is about two times faster. For the partial-scan cases, the fault coverages obtained are comparable, but the run time of the enhanced hybrid algorithm is significantly less. On average, the enhanced hybrid algorithm is about seven times faster than the hybrid algorithm. As the circuit becomes larger, the difference in CPU time between these two approaches becomes more significant.

VIII. CONCLUSIONS

In this paper, we propose a test point selection algorithm for scan-based BIST. The proposed algorithm selects the test points to maximize the overall testability improvement with or without timing constraints. A hybrid approach is used to efficiently estimate the circuit random testability improvement. The algorithm is not only suitable for full-scan circuits but also extended to partial-scan circuits by introducing a novel symbolic testability computation procedure for circuits with feedbacks. The experimental results illustrate that the proposed algorithm achieves a higher fault coverage and requires a lower complexity than previous approaches. The timing-driven TPI can minimize the performance degradation with possibly more

test points inserted. TPI for partial-scan circuits has also been shown to be very effective. A heuristic is presented to further improve the runtime efficiency. The results show that the heuristic hybrid algorithm achieves 10–15 times speedup over the CRF-based algorithm with comparable fault coverages.

REFERENCES

[1] F. Brglez, C. Gloster, and G. Kedem, "Hardware-based weighted random pattern generation for boundary scan," in *Proc. Design Automation Conf.*, Aug. 1989, pp. 264–274.

[2] H.-J. Wunderlich, "Multiple distribution for biased random test patterns," *IEEE Trans. Computer-Aided Design*, vol. 9, pp. 584–593, June 1990.

[3] S. Hellebrand, J. Rajski, S. Tarnick, S. Venkataraman, and B. Courtois, "Built-in test for circuits with scan based on reseeding of multiple-polynomial linear feedback shift registers," *IEEE Trans. Comput.*, vol. 44, pp. 223–233, Feb. 1995.

[4] N. A. Touba and E. J. McCluskey, "Altering a pseudo-random bit sequence for scan-based BIST," in *Proc. Int. Test Conf.*, Oct. 1996, pp. 167–175.

[5] H.-J. Wunderlich and G. Kiefer, "Bit-flipping BIST," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 1996, pp. 337–343.

[6] C. H. Chen, T. Karnik, and D. G. Saab, "Structural and behavioral synthesis for testability techniques," *IEEE Trans. Computer-Aided Design*, vol. 13, pp. 777–785, June 1994.

[7] F. F. Hsu, E. M. Rudnick, and J. H. Patel, "Enhancing high-level control-flow for improving testability," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 1996, pp. 322–328.

[8] A. J. Briers and K. A. E. Totton, "Random pattern testability by fast fault simulation," in *Proc. Int. Test Conf.*, Sept. 1986, pp. 506–514.

[9] V. S. Iyenhar and D. Brand, "Synthesis and pseudo-random pattern testable designs," in *Proc. Int. Test Conf.*, Aug. 1989, pp. 501–508.

[10] B. Seiss, P. Trouborst, and M. Schalz, "Test point insertion for scan-based BIST," in *Proc. Eur. Test Conf.*, Apr. 1991, pp. 253–262.

[11] Y. Savaria, M. Youssef, B. Kaminska, and M. Koudil, "Automatic test point insertion for pseudo-random testing," in *Proc. Int. Symp. Circuits and Systems*, June 1991, pp. 1960–1963.

[12] K.-T. Cheng and C.-J. Lin, "Timing-driven test point insertion for full-scan and partial-scan BIST," in *Proc. Int. Test Conf.*, Oct. 1995, pp. 506–514.

[13] N. A. Touba and E. J. McCluskey, "Test point insertion based on path tracing," in *Proc. VLSI Test Symp.*, Apr. 1996, pp. 2–8.

[14] N. Tamarapalli and J. Rajski, "Constructive multi-phases test point insertion for scan-based BIST," in *Proc. Int. Test Conf.*, Oct. 1996, pp. 649–658.

[15] H.-C. Tsai, K.-T. Cheng, C.-J. Lin, and S. Bhawmik, "A hybrid algorithm for test point selection for scan-based BIST," in *Proc. Design Automation Conf.*, June 1997, pp. 478–483.

[16] F. Brglez, "On testability of combinational networks," in *Proc. Int. Symp. Circuits and Systems*, May 1984, pp. 221–225.

[17] R. Lisanke, F. Brglez, A. J. Degeus, and D. Gregory, "Testability-driven random test pattern generation," *IEEE Trans. Computer-Aided Design*, vol. CAD-6, pp. 1082–1087, Nov. 1987.

[18] C.-J. Lin, Y. Zorian, and S. Bhawmik, "Integration of partial scan and built-in self-test," *J. Electron. Test.*, vol. 7, no. 1–2, pp. 125–137, Aug. 1995.

[19] K.-T. Cheng and V. D. Agrawal, "A partial scan method for sequential circuits with feedback," *IEEE Trans. Comput.*, vol. 39, pp. 544–548, Apr. 1990.

[20] K. P. Parker and E. J. McCluskey, "Probabilistic treatment of general combinatorial networks," *IEEE Trans. Comput.*, vol. 24, pp. 668–670, June 1975.

[21] F. Brglez, D. Bryan, and K. Kozminski, "Combinatorial profiles of sequential benchmark circuits," in *Proc. Int. Symp. Circuits and Systems*, May 1989, pp. 1929–1934.

[22] F. Brglez and H. Fujiwara, "A neural netlist of 10 combinational benchmark circuits and a target translator in Fortran," in *Proc. Int. Symp. Circuits and Systems*, June 1985, pp. 663–698.

[23] E. Sentovich, K. Singh, C. Moon, H. Savoj, R. Bryant, and A. Sangiovanni-Vincentelli, "Sequential circuit design using synthesis and optimization," in *Proc. Int. Conf. Computer Design*, Oct. 1992, pp. 328–333.

[24] K.-T. Cheng, "Redundancy removal for sequential circuits without reset states," *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 13–24, Jan. 1993.

Huan-Chih Tsai (S'98) received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1992 and the M.S. degree in electrical and computer engineering from the University of California, Santa Barbara, in 1995. He is currently working toward the Ph.D. degree at the University of California, Santa Barbara, on CAD of VLSI systems.

His research interests include built-in self-test and design-for-testability.

Kwang-Ting (Tim) Cheng (S'88–M'88) received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1983 and the Ph.D. degree in electrical engineering and computer science from the University of California, Berkeley, in 1988.

From 1988 to 1993, he worked at AT&T Bell Laboratories, Murray Hill, NJ. He joined the faculty at the University of California, Santa Barbara, in 1993, where he is currently Professor of Electrical and Computer Engineering. His current research interests include VLSI testing, design synthesis, and design verification. He has published more than 120 technical papers, coauthored two books, and holds six U.S. patents in these areas. He has also been working closely with industry in the United States for projects in these areas.

Dr. Cheng received Best Paper Award at the 1994 Design Automation Conference and the Best Paper Award at the 1987 AT&T Conference on Electronic Testing. He currently serves on the Editorial Boards of IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, IEEE DESIGN AND TEST OF COMPUTERS, and *Journal of Electronic Testing: Theory and Applications*. He has been General Chair and Program Chair of IEEE International Test Synthesis Workshop. He has also served on the technical program committees for several international conferences on CAD and testing.

Chih-Jen (Mike) Lin (S'84–M'86) received the B.S. degree in electronic engineering from National Chiao-Tung University, Shin-Chu, Taiwan, in 1978. He received the M.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1982 and the Ph.D. degree in electrical and computer engineering. from the University of Iowa, Iowa City, in 1988.

In 1988, he joined the Engineering Research Center of AT&T Bell Laboratories, Murray Hill, NJ. Since 1996, he has been with Intel, Hillsboro, OR. His main interests include DFT, BIST, delay test, and burn-in test. He has four U.S. patents and four more are pending.

Sudipta Bhawmik (S'87–M'90) received the B.Tech. and M.Tech. degrees in electronics and electrical communication engineering and the Ph.D. degree in computer science and engineering from the Indian Institute of Technology, Kharagpur, India.

He is a member of Technical Staff in the IC Test Technology and Tools Group in Bell Labs, Lucent Technologies, Murray Hill, NJ. He has been responsible for the development and application of several generations of built-in self-test technology and tools currently in use within Lucent. His current research interests are in the field of high-level DFT/BIST and system-on chip test methods. He has several technical publications and a patent.