Scan BIST Targeting Transition Faults Using a Markov Source

Hangkyu Lee
School of ECE
Purdue University
West Lafayette, IN 47907
lee10@ecn.purdue.edu

Irith Pomeranz
School of ECE
Purdue University
West Lafayette, IN 47907
pomeranz@ecn.purdue.edu

Sudhakar M. Reddy
ECE Department
University of Iowa
Iowa City, IA 52242
reddy@engineering.uiowa.edu

Abstract

We propose a new scan BIST method for transition faults. The method uses a Markov source for test pattern generation. We develop this method for skewed-load testing as well as for broadside testing. In the design of a Markov source for transition faults, we first use statistics of deterministic tests for stuck-at faults. Then, we propose a new method for identifying subsets of stuck-at tests whose statistics are useful in detecting hard-to-detect transition faults. Finally, by using statistics of tests for transition faults, we detect all the remaining detectable transition faults. The method reported achieves complete coverage of detectable transition faults in benchmark circuits using a limited number of pseudo-random tests.

1. Introduction

Built-In Self Test (BIST) [1] is a cost-effective solution to testing of integrated circuits with ever-increasing density and complexity because of several advantages it has over external test. First, BIST allows at-speed testing. When we test a circuit using an external tester, the tester is usually slower than the circuit-under-test. Hence we cannot test the circuit with timing accuracy since it does not operate at the intended speed. With BIST, the circuit can be tested at its normal operation speed. Furthermore, BIST can reduce the cost of external testers. By using BIST, we can reduce the external test data volume drastically and, therefore, the cost of the external tester can be reduced.

The design of an efficient test pattern generator is a critical issue in BIST, and a large number of techniques have been proposed for this purpose [2, 3, 4, 5, 6, 7, 8]. Especially, the weighted random pattern generation technique [9, 10, 11, 12, 13] has been shown to be a very effective way of detecting random pattern resistant faults by biasing 0 and 1 probabilities at each circuit input. Recently in [14, 15], techniques using Markov sources were shown to be effective for weighted random pattern generation. In [14], a Markov source is designed for test pattern generation in non-scan sequential circuits. In [15], a weighted random pattern generator design for scan BIST based on Markov sources is proposed and has achieved complete stuck-at fault efficiency for large benchmark circuits with very little hardware overhead and relatively small numbers of tests.

The rapid increase in clock frequencies of high-speed designs requires BIST structures capable of detecting defects that affect the timing behavior of the circuit. The goal of this paper is to present a scan BIST structure targeting transition faults using Markov sources. By doing this, we extend the application of the BIST methodology based on Markov sources to fault types that require two-pattern tests.

To design a Markov source it is necessary to determine state-transition probabilities for the finite-state machine implementing the source. In order to handle transition faults effectively, we introduce a new technique for finding subsets of a deterministic test set whose statistics are effective in determining the state-transition probabilities for the Markov source, and will lead to the detection of hard-to-detect transition faults. We consider two cases initially. One is the maximal case where all the deterministic tests for yet-undetected faults are used for determining state-transition probabilities. The other is the minimal case where only one of the deterministic tests for the yet-undetected faults is used for computing state-transition probabilities. According to fault simulation results, we decide whether we should increase the test set size used for computing state-transition probabilities starting from the minimal case or decrease the test set size starting from the maximal case in order to find an effective subset. We apply this technique first using tests for stuck-at faults, which are easier to generate, and then using tests for transition faults. We focus our efforts on achieving complete coverage of detectable transition faults for each circuit considered within limited test application time and with acceptable hardware overhead. We first assume skewed-load testing and then discuss the application to broadside testing.

Other approaches to the generation of two-pattern tests as part of a BIST scheme were described in [16, 17, 18, 19, 20, 21]. In [16], the two-pattern test capabilities of LFSRs, cellular automata, and their parallel implementation are explored using transition coverage as the metric. A weighted random pattern generation technique was proposed in [17] for robust path delay fault

IEEE COMPUTER SOCIETY

testing. A multiple-input shift register (MISR) with a constant parallel input vector is used in [18] as a test pattern generator for detection of delay faults. In [19], a new synthesis procedure for an LFSR of size n is described, in which a deterministic set of n pre-computed test pairs for delay fault testing is embedded in the maximal length pseudo-random test sequence of the LFSR. A bit transition maximization technique was used in the design of a test pattern generator to enhance the fault coverage for gate delay faults and stuck-open faults in [20]. An input reduction technique which can be used to construct test pattern generators for BIST of delay faults is described in [21].

The important feature of the proposed approach based on Markov sources is that it allows complete coverage of detectable transition faults to be achieved for benchmark circuits using limited numbers of tests and it is a natural extension to the Markov source for stuck at faults, which has similar properties.

The rest of this paper is organized as follows: Section 2 gives a brief overview of the proposed BIST methodology. Section 3 describes the new procedure for generating weighted random patterns using Markov sources for transition faults. Experimental results are presented in Section 4.

2. Preliminaries

In order to detect a transition fault on a line, the second pattern of the test has to detect a stuck-at fault on the line. For example, let us consider a transition fault that delays a rising transition ($0 \rightarrow 1$) on a line l. We have to apply a two-pattern test $<p_1, p_2>$ to detect the fault. The first pattern p_1 must assign the value 0 to line l and the second pattern p_2 must assign the value 1 to line l and propagate the effect of the faulty value 0 to a primary output. Hence the second pattern p_2 must be a test for a stuck-at 0 fault on line l. In general, there is a one-to-one correspondence between transition faults and stuck-at faults, and every transition fault has a corresponding stuck-at fault whose test is needed for detecting the transition fault. Since there is one more constraint for the first pattern of a transition fault test, the number of detectable transition faults can be smaller than the number of detectable stuck-at faults.

Similar to the procedure for stuck-at faults [15] we assume that the circuit is fully scanned and that it has only one scan chain, in which all the inputs are included. We initially assume skewed-load tests for transition faults (we discuss broadside testing later). Under this approach, the second pattern of a two-pattern test is obtained by one bit shifting of the first pattern. Therefore, if the scan chain length is L, we need L+1 bit values for each two-pattern test. In our implementation there is a dependency between two consecutive two-pattern tests determined by the following test application scheme. To obtain the first pattern of a test, we shift the scan chain by L-1 positions. As a result, the first bit of the second pattern of the previous test remains as the last bit of the first pattern of the current test. To obtain the second pattern of a test, we shift the scan chain once more. As a result we only need L new bit values for each two-pattern test, in which the first pattern consists of one bit value from the previous test and L-1 new bit values, and the second pattern has one additional new bit value. This testing method is sometimes referred to as last-shift test. We generate L bit values for each two-pattern test using a Markov source based only on the second pattern of a deterministic two-pattern test. This is acceptable due to the fact that there are usually more constraints on the second pattern than on the first pattern since, while the former is a test for a stuck-at fault, the latter only has to assign a specific value to a specific line.

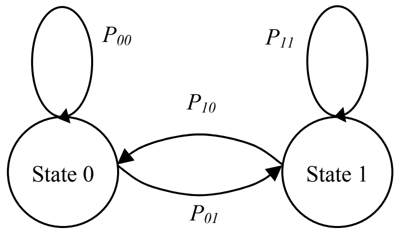

Figure 1. 2-state Markov chain

Next, we provide some details regarding the Markov source used in [15]. We concentrate on the 2-state Markov source, which we use in our experiments. A 2-state Markov chain is shown in Figure 1. The probabilities of the state-transitions leaving a state always sum to 1. Therefore, it is sufficient to specify P_{01} and P_{10} in order to specify the 2-state Markov chain completely. A 2-state Markov chain can be implemented by the 2-state finite-state machine (FSM) depicted in Figure 2. The machine is implemented using a T flip-flop, a 2:1 multiplexer, and a source of random patterns that produces patterns with 1-probability equal to P_{01} and P_{10}.

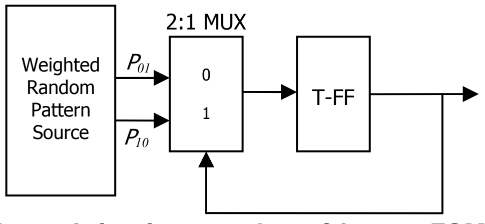

Figure 2. Implementation of 2-state FSM

The state-transition probabilities, P_{01} and P_{10}, are calculated in [15] based on deterministic test sets for stuck-at faults. The probabilities of finding the 2-state Markov chain in states 0 and 1 (denoted by P_0 and P_1, respectively) are equal to the 0 and 1 probabilities of the bit sequence generated by the 2-state FSM. They are set according to the deterministic test set so as to ensure that the 0 and 1 probabilities in the generated bit sequence are as close as possible to the 0 and 1 probabilities of bits in the test set. Specifically, the method of [15] uses the weight set $\{w_1, w_2, ..., w_m\}$ calculated from a deterministic test set. w_i corresponds to the weight, or proportion of 1s, in the ith bit position of all the tests in the given test set. The values of P_0 and P_1 are determined so as to match $w_1, w_2, ..., w_m$ as closely as possible. Once P_0 and P_1 are determined, P_{01} and P_{10} can be determined so as to ensure that the FSM is in state 0(1) with probability P_0 (P_1).

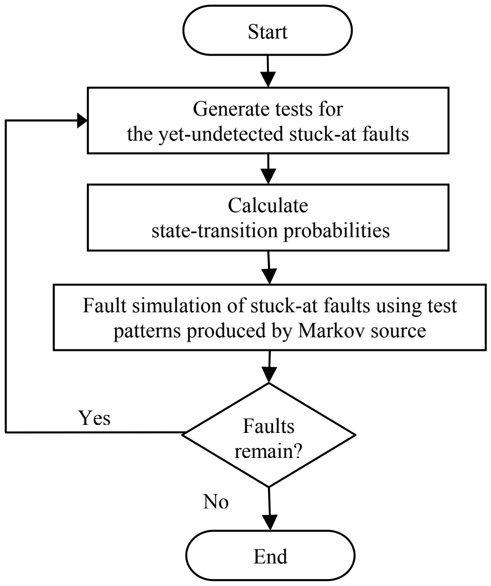
Start
Generate tests for the yet-undetected stuck-at faults
Calculate state-transition probabilities
Fault simulation of stuck-at faults using test patterns produced by Markov source
Yes
Faults remain?
No
End

Figure 3. Flow of a phase [15]

The procedure of [15] typically requires several pairs of state transition probabilities P_{01} and P_{10} in order to detect all the detectable stuck-at faults. Accordingly, the Markov source of Figure 2 is loaded with several pairs of state-transition probabilities, and it generates a fixed number of tests under every pair. The procedure for selecting the state-transition probabilities is shown in Figure 3. In every phase of the procedure, a test set is found for the yet-undetected stuck-at faults. State-transition probabilities P_{01} and P_{10} are computed based on this test set. The Markov source is simulated with P_{01} and P_{10} until N consecutive patterns do not detect any fault. If any faults remain undetected, a new phase begins.

Initially in our method for transition faults, we compute state-transition probabilities for the Markov source using the statistics of deterministic tests for the stuck-at faults corresponding to the yet-undetected transition faults. In this way we benefit from the relative simplicity of dealing with stuck-at faults, and the similarity between tests for stuck-at faults and tests for transition faults. However, test sets for stuck-at faults are not sufficient for achieving complete coverage of transition faults. Therefore, after using stuck-at test sets we switch to using tests for the yet-undetected transition faults to determine the transition probabilities of the Markov source. For both types of test sets we introduce a new procedure for selecting subsets of tests that will help us focus on the detection of yet-undetected transition faults.

The main steps of the new procedure are the following.

Step 1. We use test sets for all the stuck-at faults corresponding to the yet-undetected transition faults in order to select state-transition probabilities. At some point we reach complete fault coverage for the corresponding stuck-at faults and still have undetected transition faults.

Step 2. We again use tests for stuck-at faults corresponding to the yet-undetected transition faults in order to calculate additional state-transition probabilities. However, now we use smaller subsets of these tests. With smaller subsets the state-transition probabilities of the Markov source are better at reproducing the tests and detecting the corresponding transition faults. We introduce a procedure for selecting the subsets in Section 3.

Step 3. For the remaining transition faults, we use transition fault tests to calculate state-transition probabilities of the Markov source. We use the procedure of Step 2 to select subsets of these tests that will lead to the detection of all the remaining faults.

3. New random pattern generation procedure

In this section, we explain the new procedure in detail. The Markov source we design applies tests in one or more phases. Each phase is characterized by the state-transition probabilities of the Markov source. Different state-transition probabilities produce different test patterns, which are applied to the circuit-under-test to detect yet-undetected faults.

3.1 Using stuck-at fault tests (Step 1)

A phase of Step 1 of the procedure for transition faults consists of generating tests for stuck-at faults corresponding to the yet-undetected transition faults, calculating state-transition probabilities based on these tests, and simulating the tests produced by the Markov source with these transition probabilities. We simulate two sets of faults (with fault dropping): transition faults and stuck-at faults. We simulate tests until N consecutive tests do not detect any new transition fault. For stuck-at fault simulation we use only the second pattern of every two–pattern test.

Initially, we include all the detectable transition faults in one fault set and we include all the stuck-at faults corresponding to the detectable transition faults in the other fault set. Therefore, the initial sizes of the two fault sets are identical. At later phases, the set of yet-undetected transition faults is larger than that of yet-undetected stuck-at faults since stuck-at faults are more easily detected. Step 1 terminates when all the stuck-at faults are detected. If there are undetected transition faults at this point, we switch to a new process, described in the next subsection.

The use of tests for stuck-at faults in this step and in the next step avoids the generation of tests for transition faults until the third step when very few transition faults (if any) are left.

Our strategy for generating tests for yet-undetected faults is different from that of [15]. While ATPG is used at every phase for yet-undetected faults in [15], we perform fault simulation using a pre-generated test set that detects every detectable fault. Out of this test set, fault simulation allows us to identify the tests required to detect the yet-undetected faults.

3.2 Using subsets of stuck-at fault tests (Step 2)

In a phase of the first step, all the tests required to detect the stuck-at faults corresponding to the yet-undetected transition faults are given to the procedure for computing the state-transition probabilities of the Markov source. In the new process described in this subsection, we allow several cases for each phase. In the case referred to as the maximal case, all the tests for the yet-undetected faults are given to the procedure. In the cases referred to as the minimal cases, a single test at a time is given to the procedure. With a single test, the procedure for selecting the state-transition probabilities focuses on reproducing the test. This is effective in detecting hard-to-detect transition faults. For a test set of size n, we call the procedure once for the maximal case and n times for the minimal cases. We select the best case among the n+1 cases, which is the one that results in the detection of the largest number of transition faults. If two cases have the same number of detected faults, we select the one with the smaller number of test patterns.

If the best case is the maximal case, we decrease the test set size by 1 and continue to consider n test sets of size n - 1. In the ith test set, test i is missing, relative to the maximal case. If the best case is one of the minimal cases, we increase the test set size by 1 and continue to consider n-1 test sets of size 2 that include the best minimal test set and one additional test. We compute state-transition probabilities and test patterns for every one of the cases to identify the best one. We compare the new best case with the previous best case. If the number of detected transition faults in the new best case is larger than or equal to that in the previous best case, then we keep increasing or decreasing the test set size. If not, we select the previous best case and terminate the phase. In the worst case, we will increase or decrease the test set size n-1 times until we find the best state-transition probabilities for the current phase.

There is a point at which no new transition fault is detected for M consecutive phases, where M is a user-defined value. Then we switch to using deterministic test sets for transition faults and apply the same process using these test sets as described next.

3.3 Using transition fault tests (Step 3)

Transition faults may remain undetected after Step 2 due to the fact that the deterministic tests we use for stuck-at faults do not incorporate the information that skewed-load testing is used. Thus, the tests we use do not take into account that a test for a fault line l stuck-at a must be a single shift version of a pattern that sets l=a for the corresponding transition fault to be detected. In Step 3, we overcome this problem by using an ATPG for transition faults that generates skewed-load tests. We use the ATPG described in [22] for this purpose. We apply the same process of Step 2 except that we use transition fault tests instead of stuck-at fault tests. We do not use test sets for transition faults in earlier steps because the size of a test set for transition faults is much larger than that for stuck-at faults, and the complexity of generating it is higher.

3.4 Broadside testing

In broadside testing, the first pattern of a transition fault test is applied through the scan chain and the second pattern is applied through the functional path of the circuit. Since the second pattern of a two-pattern test is determined by the response of the combinational logic of the circuit to the first pattern, it is more difficult to obtain a particular second pattern by controlling the first pattern. To overcome this problem we use several techniques, briefly described next.

We use statistics of stuck-at tests, corresponding to the first pattern of a two-pattern test for a transition fault, in calculating weights for Step 1 and Step 2. For example, consider a rising transition fault ($0 \rightarrow 1$) on a line l. The first pattern of the test should assign the logic value 0 to line l. Since the test for the stuck-at 1 fault on line l should assign the logic value 0 to line l, we use the stuck-at 1 test for weight calculation.

In Step 2 described in Subsection 3.2, we consider several options for each phase before we select the best one. This is also done in the modified Step 2. However, during this optimization process, every time we compute a set of parameters that detects any yet-undetected faults, we define a phase based on these parameters. We use the same process in Step 3. As a result we obtain significantly more phases than in the skewed-load case. We remove unnecessary phases by reverse order simulation of the final set of phases.

In Step 3 we use a broadside test set to compute additional weight assignments. These tests allow us to achieve complete fault coverage for the circuits considered.

Table 1. Experimental result of skewed-load testing

Circuits	Stuck-at [15]		Step 1			Step 2			Step 3			Total Phases	Total Vectors
	Phases	Vectors	Phases	FE(%)	Vectors	Phases	FE(%)	Vectors	Phases	FE(%)	Vectors		
S9234	6	44.8K	5	99.47	141.9K	10	99.84	86.1K	9	100	28.5K	24	256.4K
S13207	4	24.8K	2	99.18	88.5K	9	99.89	63.7K	1	100	8.1K	12	160.3K
S15850	4	45.5K	5	99.63	156.5K	6	99.98	34.3K	3	100	1.9K	14	192.6K
S38417	6	87.1K	9	99.74	460.6K	8	99.96	117.9K	18	100	21.6K	35	600.0K
S38584	6	49.7K	7	99.91	213.8K	7	99.99	22.7K	5	100	22.2K	19	258.7K

Table 2. Experimental results of broadside testing

Circuits	Step 1			Step 2			Step 3			Total Phases	Total Vectors
	Phases	FE(%)	Vectors	Phases	FE(%)	Vectors	Phases	FE(%)	Vectors		
S9234	3	91.85	295.9K	128	99.19	1,038.0K	36	100	109.4K	94	1,239.5K
S13207	3	97.44	278.8K	50	99.96	535.5K	4	100	10.6K	36	501.9K
S15850	3	96.51	269.6K	40	99.28	327.6K	35	100	184.6K	62	558.4K
S38417	3	99.03	651.7K	274	99.98	2,345.6K	10	100	46.1K	118	1,717.7K
S38584	3	98.86	508.3K	129	99.85	874.1K	38	100	870.6K	116	2,021.5K

4. Results

We performed experiments on the large ISCAS89 benchmark circuits. We first consider skewed-load testing. We use the same setup used in [15]. Specifically, we use virtual scan chains as in [15]. The use of virtual scan chains means that the single scan chain of a design is conceptually divided into multiple subchains, and state-transition probabilities are selected for each subchain separately. This is needed because the larger the length of a scan chain, the harder it becomes to satisfy weight requirements at every bit position. We used the same virtual chain lengths used in [15]. For s9234, s13207 and s15850 the virtual chain length is 48. For s38417 and s38584 the virtual chain length is 64.

If the weight of the generated sequence and the weight required at a circuit input are considerably different, the FSM output values corresponding to the circuit input are inverted before feeding the scan chain. This is done by an inversion logic similar to the one described in [15].

As a stopping criterion for each phase, if 4096 consecutive patterns (2048 two-pattern tests) do not detect any yet-undetected fault, then the application of weighted random patterns is terminated. In [15], the stopping criterion considers 2048 consecutive patterns.

We used five quantization levels, 0.125, 0.25, 0.5, 0.75 and 0.875 in determining the state-transition probabilities. Every one of the five quantization levels is allowed in the first four phases of each step and only two quantization levels, 0.25 and 0.75 or 0.125 and 0.875, are used in the later phases of each step. In [15], every one of the five quantization levels is used only in the first phase. Since we need more phases than [15], the constraint of a single phase with all five quantization levels is too strong for our case.

We used a 2-state FSM as part of the Markov source in order to reduce hardware overhead and computational complexity. Complete fault coverage is achieved in every case using a 2-state FSM. By complete fault coverage, we mean that all the detectable transition faults are detected. This is equivalent to 100% fault efficiency. The undetectable transition faults using skewed-load tests were determined by the ATPG from [22].

Results of skewed-load testing are presented in Table 1. The first column contains the circuit name. The second and third columns list the number of phases and the number of vectors required to achieve 100% stuck-at fault efficiency in [15]. Columns 4, 5 and 6 show the number of phases used, the transition fault efficiency (FE) and the number of vectors applied when Step 1 is terminated. At the end of this step 100% of the stuck-at faults corresponding to the detectable transition faults are also detected. Columns 7, 8 and 9 and column 10, 11 and 12 are for Step 2 and Step 3, respectively. In column 13 and 14, the total number of phases and the total number of vectors applied to each circuit are presented, respectively.

For every circuit considered, we achieved complete coverage of detectable transition faults. After Step 1, the fault efficiency for each circuit is above 99%, which demonstrates that Markov sources designed using stuck-at fault tests are effective in detecting almost all the transition faults. For the remaining transition faults, we had to use Step 2 and Step 3 to detect them. There is a tendency that the more random pattern resistant faults a circuit has, the more phases of Step 3 are required to achieve complete fault coverage. For s9234 and s38417, that are well-known random pattern resistant circuits, the number of phases in Step 3 is much larger than that of the other circuits.

If we compare our new method for transition faults to the original method [15] for stuck-at faults we can see that the number of applied vectors in the new method is much smaller than ten times the number of applied vectors in the original method. This is in contrast to more than an order of magnitude increase in the number of pseudo-random tests for transition faults over those for stuck-at faults reported by an earlier study [23], even when incomplete coverage of transition faults was achieved. This shows the effectiveness of the new method when we consider the difficulty of detecting transition faults using random patterns.

We show the results of broadside testing before and after reverse order simulation in Table 2. The number of phases and the number of patterns obtained after reverse order simulation are shown in the last two columns of Table 2. As a stopping criterion for each phase, if 8192 consecutive patterns (4096 two-pattern tests) do not detect any yet-undetected fault, then the application of weighted random patterns is terminated.

As in skewed-load testing, we allowed three quantization level sets, {0.125, 0.25, 0.5, 0.75, 0.875}, {0.25, 0.75} and {0.125, 0.875}. For every step, we used each quantization level set once. In Step 1, one pass under each quantization level set produces only one phase. On the other hand, in Step 2 and Step 3, one pass under each quantization level can produce several phases. The undetectable transition faults using broadside tests were determined by the ATPG from [22].

The results show that 4.17 times as many phases and 4.31 times as many patterns are required to achieve the same fault efficiency as in skewed-load testing on the average. The fault efficiency after Step 1 is lower than under skewed-load testing. Nevertheless, Steps 2 and 3 compensate for this and eventually achieve 100% fault efficiency.

5. Conclusion

We described a new weighted random pattern generation procedure based on Markov sources for scan BIST targeting transition faults. This procedure consists of three steps. Each step uses a distinct strategy to detect yet-undetected faults. Step 1 uses statistics of test sets for yet-undetected stuck-at faults corresponding to yet-undetected transition faults, Step 2 finds optimal subsets of test sets for yet-undetected stuck-at faults corresponding to yet-undetected transition faults, and Step 3 uses test sets for transition faults instead of stuck-at faults in a similar way to Step 2.

The experimental results for skewed-load testing showed that every detectable transition fault of large benchmark circuits can be detected through the three steps by applying a reasonable number of weighted random patterns. This demonstrated that a BIST methodology based on Markov sources can be an effective solution not only for stuck-at faults but also for transition faults. We also discussed the application to broadside testing and presented experimental results.

References

[1] M. Abramovici, M. Breuer and A. Friedman, *Digital System Testing and Testable Design*, IEEE Press, Piscataway, NJ, 1990.

[2] E.B. Eichelberger, and E. Lindbloom, "Random-Pattern Coverage Enhancement and Diagnosis for LSSD Logic Self-Test", *IBM Journal of Research and Development*, Vol. 27, No. 3, May 1983 , pp. 265-272.

[3] B. Koenemann, " LFSR-Coded Test Patterns for Scan Designs", *Proc. of European Design and Test Conference*, 1991, pp. 237-242.

[4] S. Venkataraman, J. Rajski, S. Hellebrand, and S. Tarnick, "An Efficient BIST Scheme Based on Reseeding of Multiple Polynomial Linear Feedback Shift Registers", *Proc. of Int'l Conference on Computer-Aided Design*, Nov. 1993, pp. 572-577.

[5] K.T. Cheng and C.J. Lin, "Timing-Driven Test Point Insertion for Full-Scan and Partial-Scan BIST", *Proc. of Int'l Test Conference*, 1995, pp. 506-514.

[6] N. Zacharia, J. Rajski, J. Tyszer, "Decompression of Test Data Using Variable-Length Seed LFSRs", *Proc. of VLSI Test Symposium*, 1995, pp. 426-433.

[7] N. Touba and E. McCluskey, "Altering a Pseudo-random Bit Sequence for Scan-based BIST", *Proc. of Int'l Test Conference*, Oct. 1996, pp. 167-175.

[8] H. J. Wunderlich and G. Kiefer, "Bit-flipping BIST", *Proc. of Int'l Conference on Computer-Aided Design*, Nov. 1996, pp. 337-345.

[9] H.J. Wunderlich, "Multiple Distributions for Biased Random Test Patterns", *IEEE Trans. on Computer-Aided Design*, Vol. 9, No. 6, Jun. 1990, pp. 584-593.

[10] F. Muradali, V.K. Agarwal, and B. Nadeu-Dostie, "A New Procedure for Weighted Random Built-in Self Test", *Proc. of Int'l Test Conference*, 1990, pp. 660-669.

[11] I. Pomeranz, and S.M. Reddy "3-weight Pseudo-Random Test Generation Based on a Deterministic Test Set for Combinational and Sequential Circuits", *IEEE Trans. on Computer-Aided Design of Circuits and Systems*, Vol. 12, No. 7, July 1993, pp. 1050-1058.

[12] B. Reeb and H.J. Wunderlich, "Deterministic Pattern Generation for Weighted Random Pattern Testing", *Proc. of European Design and Test Conference*, 1996, pp. 30-36.

[13] H.S. Kim, J. Lee, and S. Kang, "A New Multiple Weight Set Calculation Algorithm", *Proc. of Int'l Test Conference*, 2001, pp. 878-884.

[14] L. Brehelin, O. Gascuel, G. Caraux, P. Girard, C. Landrault, **"**Hidden Markov and Independence Models with Patterns for Sequential BIST", *Proc. of VLSI Test Symposium*, 2000, pp. 359 -367.

[15] N.Z. Basturkmen, S.M. Reddy, I. Pomeranz, "Pseudo Random Patterns Using Markov Sources for Scan BIST", *Proc. of Int'l Test Conference*, 2002, pp 1013-1021.

[16] K. Furuya, E. J. McCluskey, "Two-Pattern Test Capabilities of Autonomous TPG Circuits", *Proc. of Int'l Test Conference*, 1991, pp. 704-711.

[17] W. Wang, and S.K. Gupta, "Weighted Random Robust Path Delay Testing of Synthesized Multilevel Circuits", *Proc. VLSI Test Symposium*, 1994, pp. 25-28.

[18] B. Wurth and K. Fuchs, "A BIST Approach to Delay Fault Testing with Reduced Test Length", *Proc. of European Design and Test Conference*, 1995, pp. 418-423.

[19] C. Dufaza and Y. Zorian, "On the Generation of Pseudo-Deterministic Two-Patterns Test Sequence with LFSRs", *Proc. of European Design and Test Conference*, 1997, pp. 69-76.

[20] B.F. Cockburn and A.L.-C. Kwong, "Transition Maximization Techniques for Enhancing the Two-Pattern Fault Coverage of Pseudorandom Test Pattern Generators", *Proc. VLSI Test Symposium*, 1998, pp. 26-30.

[21] C.A. Chen, S.K. Gupta, "Efficient BIST TPG Design and Test Set Compaction for Delay Testing via Input Reduction", *Proc. of International Conference on Computer Design*, 1998, pp. 32-39.

[22] Y. Shao, I. Pomeranz and S.M. Reddy, "On Generating High Quality Tests for Transition Faults", *Proc. Asian Test Symposium*, 2002, pp. 1-8.

[23] J.A. Waicukauski, E. Lindbloom, B. Rosen and V. Iyengar, "Transition Fault Simulation", *IEEE Design and Test*, Apr. 1987, pp. 32-38.