A Novel Approach to Random Pattern Testing of Sequential Circuits

Lama Nachman, Kewal K. Saluja, *Fellow*, *IEEE*,
S.J. Upadhyaya, *Member*, *IEEE*, and Robert Reuse

Abstract— Random pattern testing methods are known to result in poor fault coverage for most sequential circuits unless costly circuit modifications are made. In this paper, we propose a novel approach to improve the random pattern testability of sequential circuits. We introduce the concept of holding signals at primary inputs and scan flip-flops of a partially scanned sequential circuit for a certain length of time, instead of applying a new random vector at each clock cycle. When a random vector is held at the primary inputs of the circuit under test or at the scan flip-flops, the system clock is applied and the primary outputs of the circuit are observed. Information obtained from a testability analysis or test generator is used to determine the number of clock cycles for which each random vector is to be held constant. The method is low cost and the results of our experiment on the benchmark circuits show that it is very effective in providing fault coverage close to the maximum obtainable fault coverage using random patterns with full scan.

Index Terms—Fault coverage, hold method, partial scan, random pattern testing, sequential circuit testing.

1 Introduction

In most sequential circuits, random testing is simply not an option because of the poor fault coverage obtained, even if a very large number of random vectors is used. Weighted random pattern testing [1], [2] has proven marginally more effective; however, the excessive complexity of determining the weights, as well as the huge number of test vectors required [3], decreases its desirability. Sequential Automatic Test Pattern Generators (ATPG) like FASTEST [4], multiple observation time test strategy [5], HITEC [6], and GENTEST [7] do achieve high fault coverage for sequential circuits; however, they are computationally expensive and do not achieve satisfactory coverage for some complex circuits. Converting sequential circuits to combinational circuits using the full-scan approach [8] and, then, using combinational ATPG or random testing offers certain benefits. But, full-scan often requires a high cost in hardware and performance. To control the hardware overhead and performance degradation associated with full-scan, partial-scan techniques [9] are employed. In such a case, sequential ATPG is required for deterministic test generation and it can clearly be costly. Efforts have also been made to use partial scan based design-for-test to improve the weighted random pattern testability of sequential circuits [10]. This method has been studied for a special set of circuits and we believe that, if this method is applied to benchmark circuits, it will require a large number of flip-flops to be included in the partial scan path.

In this paper, we propose a systematic approach to increase the fault coverage obtained by random pattern testing. Our approach consists of two main components, a hold method and partial scan. The hold method is a very simple, yet effective, technique to increase the fault coverage of a sequential circuit. It consists of applying a random vector at the primary inputs of the sequential circuit and holding it constant for some number, k, of clock cycles, instead of applying a new random vector every clock cycle. The concept of keeping certain signals constant during testing to achieve better performance from a test has been studied in testing literature [11], [12] and is found to be productive. For instance, the internal state of a sequential circuit is held constant for certain number of clock cycles in FREEZE [11]. In [12], a randomly addressed memory location is initialized and held at that value for a fixed number of clock cycles to test the post-logic of embedded arrays. In our technique, initially reported in [13], as in the other similar techniques, the holding of signals can be easily integrated with the mechanism used to supply the random vectors. The number k, for which the input is held constant, is circuit dependent and can be calculated using either testability analysis methods or a test pattern generator. In a sense, k is a measure of the number of clock cycles required to propagate a value from a flip-flop to a primary output in the circuit. If TPG is chosen to compute k, then the TPG is used only for a small number of faulty lines in the circuit, and, as a result, the computation time is much less than in the case of deterministic testing.

In those cases where the fault coverage obtained using the hold method is not adequate, partial scan can be used to increase the coverage. In this case, a number of flip-flops is chosen to be added to the scan chain, using any of the known flip-flop selection methods, such as the Reverse direction Empirical Testability Difference (RETD) [14] or cycle breaking method [15]. Then, random testing can be used on the modified circuit. The partial scan approach does improve the coverage obtained by random pattern testing; however, unlike the hold method, it requires hardware modification. By simulating the full-scan version of the circuit, we obtain an upper bound on the fault coverage achievable by random pattern testing. This coverage is the best coverage that can be obtained using random pattern testing. Thus, the goal of our method is to achieve the upper bound fault coverage using partial scan in conjunction with the hold method.

This paper is organized as follows. In Section 2, we introduce the conceptual reasoning behind the working of the hold method, and explain how to calculate the value of k. In Section 3, we present the details of our algorithm. In Section 4, we report the results obtained using our approach. Section 5 compares our work with related work and Section 6 concludes the paper.

2 Hold Method

In the conventional random pattern testing approach, a new random vector is applied to the circuit at every clock cycle. To see why this method is ineffective, let us consider the following case. Assume that a random vector v is applied to the circuit, which causes some faults to be excited but not yet propagated to a primary output. The application of a system clock will capture the fault effects in some flip-flops. If we now change v (apply a new random vector), the new vector might break a path that was already sensitized by the vector v, and, as a result, the faults that were excited by the previous vector will not be propagated by applying a new random vector. On the other hand, if the vector v was held at the primary inputs for some number of clocks k, long enough to propagate the faults to primary outputs, there is a good chance that the faults would be detected. Thus, if we were to use random pattern testing in sequential circuits, the goal should be to give sufficient time to each vector so that the excited fault(s) are propagated to a primary output before applying the next random vector. Given the above reasoning, the number k should be equal to the number of clock cycles needed to propagate a value from the input of a flip-flop in the circuit to a primary output. Hence, we need to answer the following two questions:

- *L. Nachman is with Intel Corporation,*
- *K.K. Saluja is with the Department of Electrical and Computer Engineering, University of Wisconsin, Madison, WI 53706. E-mail: saluja@engr.wisc.edu.*
- *S.J. Upadhyaya and R. Reuse are with the Department of Electrical and Computer Engineering, State University of New York, Buffalo, NY 14260. E-mail: shambhu@eng.buffalo.edu.*

For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number 105948.

0018-9340/98/$10.00 © 1998 IEEE

1) How do we estimate the number of clock cycles k needed to propagate a fault from the input of a given flip-flop to any primary output in the circuit?
2) What would the value of k be, given the numbers calculated in step 1 for all flip-flops in the circuit?

To answer the first question, we could use either testability analysis methods or a test pattern generator.

- Testability analysis based techniques:
  SCOAP [16] or similar programs can be used to calculate the sequential observability values of all flip-flops in a circuit. Those values can then be normalized to get the number of clock cycles required to observe each flip-flop at a primary output.
- Test pattern generation based technique:
  A test pattern generator, like FASTEST [4], can be used to detect faults on the output of each flip-flop in the circuit (output stuck-at-0 and output stuck-at-1). The length of the test sequence generated by the test generator for each fault, is the number of clock cycles required to both excite the fault and propagate it to a primary output. To calculate the number of clock cycles required to only propagate the fault, we should initialize the state of the flip-flop to the opposite value of the fault (i.e., if the fault is output stuck-at-0, then the flip-flop should be initialized to logic 1 and vice versa). By doing so, we are exciting the fault using the initial state, and all the test generator needs to do is to propagate the fault forward. Thus, a test generator can provide the number of clock cycles required to observe a flip-flop at a primary output.

The second question deals with the problem of combining values obtained, in either of the two techniques, for all flip-flops to calculate the value k that is to be used during the testing of the circuit. One way would be to choose k as the average of all the values obtained over all flip-flops. Alternatively, one could use the sequential depth [15] of the circuit, which is defined as the largest number of flip-flops on a path between inputs and outputs. This is the simplest method; however, there are better solutions. Instead of using only one value of k for a circuit, we could use multiple values of k as detailed in the following section.

3 The Algorithm

Our algorithm starts with running a fault simulator on the sequential circuit using n randomly generated vectors. If the fault coverage obtained from conventional random testing is sufficient, then the circuit is classified as random pattern testable, and the algorithm quits at this point. If the fault coverage is not adequate, then the algorithm applies the hold method. The values of k are calculated using FASTEST. At this point, the random vectors obtained a priori are used, however, each vector is held constant for a number of clock cycles equal to k. As a result, only n/k vectors are applied in this step. This is done basically to keep the total number of clock cycles used to test the circuit constant. If the fault coverage obtained in this step is adequate, the procedure quits; otherwise, it moves to the partial scan step. In this step, a number of flip-flops (f) are added to the scan chain, and a set of random vectors is generated. Again, the number of clock cycles used to test the circuit is kept constant, and, as a result, the number of vectors generated at this step is equal to n/f, to compensate for the extra clocks required to scan each vector. The fault simulator is invoked again with the new set of random vectors, using the partial scan circuit, and the fault coverage is checked. If it is adequate, the algorithm exits; otherwise, the full-scan circuit is simulated with the same number of vectors, as in partial scan, i.e., n/f, to obtain an upper bound on the fault coverage achievable using partial-scan. If

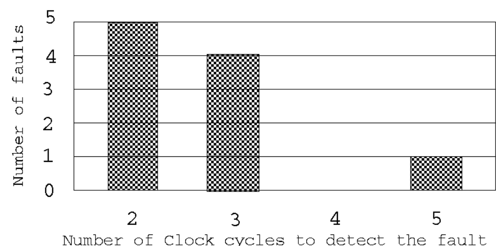

Fig. 1. The distribution of the k values for the circuit s820.

the upper bound is equal to the fault coverage obtained using partial scan, then no further improvement can be achieved by this procedure, and the algorithm quits; otherwise, FASTEST is used again to generate the new values of k for the partially scanned circuit and the hold method is used again, but this time for the partial scan circuit and using the new values of k.

Below we explain each step of our algorithm through an example. We use the ISCAS benchmark circuit s820 as the example circuit to facilitate the explanation.

Step 1: We generated n = 100,000 random vectors for s820, and ran it through a sequential fault simulator. The fault coverage obtained at this step was 49.3 percent.

Step 2: We now need to determine the value (or values) of k. The circuit s820 has five flip-flops; hence, FASTEST was used to generate tests for 10 faults (treating each flip-flop output to be stuck-at-1 and, then, stuck-at-0). We used FASTEST to obtain the number of clock cycles required to detect each fault. Fig. 1 contains the results of this step. The number of faults is shown on the y-axis and the number of clock cycles required to detect (*excite* + *propagate*) each fault is shown on the x-axis. For example, five out of a total of 10 faults require only two clock cycles to excite and propagate the faults.

Step 3: Based on this result, three values of k are chosen for use in the hold method. The value k = 2 is used for 50 percent of the time, k = 3 is used for 40 percent of the time, and k = 5 is used for 10 percent of the time. To be able to compare the fault coverage obtained from the previous step with the coverage using the hold method, the total number of clock cycles used in testing the circuit is kept at 100,000. For k = 2, we need a total of 50,000 clock cycles, and, since each random vector needs to be held at the inputs for two clock cycles (k = 2), the number of unique vectors will be 25,000. Similarly, for k = 3, 13,333 unique vectors are applied in 40,000 clock cycles, and, for k = 5, 2,000 unique vectors are applied in 10,000 clock cycles. This amounts to only a total of 40,333 different random vectors.

The fault coverage obtained by running the fault simulator with the above distribution of vectors as inputs to the circuit was 57.8 percent, which is an improvement of 8.5 percent over the conventional random pattern testing method. This is in spite of the fact that we are using fewer unique vectors (40,333) as compared to 100,000 used in step 1.

Step 4: We consider partial scan solution, provided the fault coverage obtained in step 3 is not acceptable, as is the case for s820. We chose to add two flip-flops to the scan chain. The selection of those flip-flops was obtained from the previous work done by Kim and Kime [14].

The circuit with partial scan is fault simulated using random vectors as follows. Since it is important to compare the fault coverage using some common metric, we choose the metric to be the total number of test clocks (100,000 in this case). The total number of vectors used for this step is obtained by dividing the original number of vectors by the number of flip-flops in the scan chain. Therefore, we generated 50,000 random vectors and applied these to the partially scanned circuit using our fault simulator. The fault

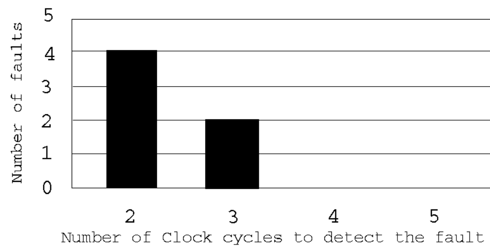

Fig. 2. The distribution of the *k* values for the partially scanned s820.

coverage obtained was 94.4 percent. This is an improvement of 36.6 percent over the coverage obtained in step 2.

Step 5: Next, we determine the bound on the random pattern testability of the combinational part of the circuit. For this, we converted the circuit to a full-scan circuit and, again, ran it through the fault simulator using the same number of vectors as in the partial scan step (50,000 vectors). The coverage obtained was 100 percent, which means that there might still be some room for improvement. Had the coverage been the same as in Step 4, we would have terminated the experiment at this point.

Step 6: This step is the same as Step 2, except for the fact that we use the partially scanned circuit, instead of the original circuit, to obtain different values of *k*. The results of this step, using FASTEST, are shown in Fig. 2. Note that the circuit s820 has only three nonscan flip-flops. Therefore, values of *k* were obtained for only six faults at the outputs of these flip-flops.

From the histogram shown in Fig. 2, we determined the values of *k* to be two and three. The total number of test clocks is assumed to be constant. Thus, the total number of clocks for a given *k*, for which random vectors are used, is computed as follows:

$$T(k) = (\textit{original number of vectors}) \times F(k),$$

where

$$F(k) = (\textit{fraction of } k)/(\# \textit{ of FFs in scan path}).$$

For our example (s820), there are two flip-flops in the scan path. The fractions of *k* are 4/6 and 2/6 for *k* = 2 and 3, respectively. The total number of unique vectors for each *k* is given by:

$$\textit{Number of unique vectors}(k) = T(k)/k.$$

Thus, assuming that the total test time is 100,000 clocks, 16,667 unique vectors are applied for 33,337 clock cycles and 5,556 unique vectors are applied for 16,667 clock cycles. Note that, in this kind of testing, we will need to apply both scan clocks, as well as system clocks. The fault coverage obtained in this step was 96.4 percent, which is an improvement of 2 percent over the coverage obtained in the previous step.

TABLE 1
CIRCUIT DETAILS

Circuit	Number of Inputs	Number of Outputs	Number of Gates	Number of Flip-flops	Number of Faults
s382	3	6	158	21	399
s420	19	2	196	16	430
s444	3	6	181	21	474
s820	18	19	289	5	850
s838	35	2	390	32	857
s953	16	23	395	29	1,079
s1423	17	5	657	74	1,515
s5378	35	49	2,779	179	4 603
s13207	62	152	7,951	638	9,589
s35932	35	320	16,065	1,728	39,094

4 EXPERIMENTAL EVALUATION

We performed experiments on the ISCAS-89 benchmark circuits listed in Table 1.

The table includes the various attributes of the circuits used in our experiments. We first generated *n* = 100,000 random vectors for each circuit. The value of 100,000 was used basically to keep the time to complete the experiment within a manageable limit and also to have a common basis for comparison. In practice, a much larger value should be used for larger circuits. We then used the fault simulator SEQSIM, developed by us, on each circuit, using 100,000 random vectors. The fault coverage obtained for each of the circuits is shown in column 2 of Table 2. As stated earlier, the fault coverage values are very poor for most circuits, except for the circuit s35932, which seems to be a random pattern testable circuit to begin with.

Following this, we calculated the *k* values for each circuit and ran the fault simulator again using the hold method. The fault coverage values, the number of unique vectors, and the different values of *k* obtained using FASTEST are shown in columns 3, 4, and 5, respectively, of Table 2. Also shown in the table (column 6) are the fault coverage values using the sequential depth metric, which is often cited as a measure of sequentiality of a circuit. This metric is obtained from published data on the ISCAS-89 benchmarks [17]. In some circuits, we see a major improvement in fault coverage (s382, s444, s1423), others show a minor improvement (s820, s5378), and two circuits show no improvement (s953, s13207). However, none of the circuits show any degradation in fault coverage as a result of applying the hold method and fewer number of unique vectors. In the case of the two circuits, s420 and s838, FASTEST was unable to initialize them, and, as a result, we weren't able to estimate the values of *k* for these two circuits. However, the results based on sequential depth show that significant increases in coverage are obtained by the hold method.

TABLE 2
CONVENTIONAL RANDOM TESTING COMPARED TO THE HOLD METHOD

Circuit	Normal Random Testing	Random Testing with the Hold Method				
		Varying Values of k			k Fixed to Seq. Depth	
	FC (%) (n = 100K)	FC (%)	Number of Unique Vectors	Values of k	FC (%)	Seq. Depth
s382	13.28	86	34,807	2,3,13,20	78.3	13
s420	5.59	-	-	See text	40.2	15
s444	12.66	80.4	34,807	2,3,13,20	78.8	12
s820	49.3	57.8	40,333	2,3,5	62.4	5
s838	4.5	-	-	See text	28.5	25
s953	8.34	8.3	50,000	2	8.3	6
s1423	57.76	86.3	14,469	5, 8, 11, 15, 20	78.0	57
s5378	69.74	74.4	25,868	2, 3, 6, 9, 14, 25	70.4	57
s13207	9.19	9.2	18,229	2, 6, 10, 13, 17	9.1	61
s35932	86	87	41,667	2, 3	83. 3	30

TABLE 3
COMPARISON OF THE HOLD METHOD WITH DETERMINISTIC TEST GENERATION SYSTEMS

Circuit	Hold Method (Table 2) FC (%)	Deterministic Test Generation Systems FC (%)	ATPG
s382	86	90.7	DUST
s444	80.4	87.8	Lee & Reddy
s820	57.8	95.8	SAT
s953	8.3	8.3	SAT
s1423	86.3	86.4	FASTEST
s5378	74.4	78.5	FASTEST
s13207	9.2	9.4	SAT
s35932	87	89.7	SAT

The reference sources for these test generators are: DUST [18], Lee and Reddy [19], SAT [20], and FASTEST [4].

How well does the hold method compare with deterministic testing? To answer this question, we include Table 3, which contains the coverage that can be obtained by deterministic test generators for these circuits. Only those circuits to which hold method could be applied using the tools available to us are included in Table 3. The fault coverages given in the table are taken from the different sources identified in the table. Note that the noninclusion of some circuits is not a limitation of the algorithm, but was primarily due to the limitation of the test tools used by us. Table 3 demonstrates that hold method is fairly effective in getting close to the coverage offered by the deterministic methods, and at a nearly negligible cost of fault simulation, as opposed to test generation.

We then moved to the partial scan step. The total number of flip-flops as well as the number of selected flip-flops in the scan chain are shown in Table 4 (columns 2 and 3). The selection of the scan flip-flops was obtained from [14]. Random vectors were generated for the partially scanned circuits and the fault simulator was used on these circuits. The fault coverage values, as well as the number of unique vectors, are shown in Table 4, columns 4 and 5, respectively. In all circuits except s35932, we see an improvement in fault coverage when using partial scan. We believe that the reason for poor performance of random vectors on s35932 is the fact that our limit on the total test time (100,000 clocks), combined with a fairly large number of FFs in the scan path (297 FFs), allows only a very small total number of random vectors (337 vectors) to be used to test the circuit. This number of random vectors is clearly very low to test a fairly large sequential circuit. However, the fact that the fault coverage doesn't deteriorate substantially, in spite of a huge decrease in the number of unique vectors (fault coverage decreases by only 5.6 percent, whereas the number of vectors reduced from 100,000 to 337), proves the effectiveness of partial scan in random pattern testing of sequential circuits.

Next, we ran the fault simulator on the full-scan circuits to obtain an upper bound on the fault coverage that can be obtained using random testing in the partial scan environment. We used the same number of vectors as in the partial scan case to arrive at a possibly obtainable upper bound for the fault coverage. This upper bound does not include failures due to timing, since the test may not be performed at normal speed [21]. Note that, although the number of test vectors is the same as those in the partial scan case, the test time for full-scan circuits will be between 103 percent to 967 percent of the test time for partial scan circuit. The results of this simulation are shown in Table 5. We see that the circuits s420, s838, and s953 achieve the same coverage as in partial scan. As a result, these circuits are eliminated from any further consideration, as the maximum achievable coverage using random vectors has already been achieved for these circuits.

After eliminating the circuits mentioned above, we calculated the values of *k* for the remaining partially scanned circuits. We

TABLE 4
THE RESULTS OF COMBINING RANDOM PATTERN TESTING WITH PARTIAL SCAN

Circuit	Num. of FFs	FFs in Scan Path	Random Testing with Partial Scan FC (%)	Num. of Unique Vectors	Comments
s382	21	7	95.7	14,286	
s420	16	15	80.0	6,667	Max FC†
s444	21	6	92.8	16,667	
s820	5	2	94.3	50,000	
s838	32	31	58.0	3.226	Max FC†
s953	29	3	99.9	33,333	Max FC†
s1423	74	41	89.0	2,439	
s5378	179	29	88.7	3,448	
s13207	638	391	58.7	256	
s35932	1,728	297	81.4	337	

† This is the maximum achievable fault coverage. See text and Table 5.

TABLE 5
FAULT COVERAGE OBTAINED USING RANDOM VECTORS ON THE FULL SCAN CIRCUITS

Circuit	Random Testing with Full Scan FC (%)	Number of Unique Vectors	Number of Test Clocks
s382	100.0	14,286	300,006
s420	80.0	6,667	106,672
s444	97.0	16,667	350,007
s820	100.0	50,000	250.000
s838	57.5	3.226	103,332
s953	99.9	33,333	966,657
s1423	95.3	2,439	180.486
s5378	97.5	3,448	617,192
s13207	74.7	256	163,328
s35932	89.6	337	582,336

Note: The number of vectors used is the same as in the partial scan results.

then simulated these circuits with partial scan, using the hold method, and obtained the results given in Table 6. All circuits, except s5378 and s13207, show an additional improvement in fault coverage over the previous step. We believe that the fault coverage of the two circuits in question did not improve because of using pessimistic values of *k* (number of clock cycles required to both excite and propagate the faults, as opposed to looking at propagation values only). For the circuit s5378, we estimated the *k* values for fault propagation alone and re-ran the experiment. Propagation values were indirectly estimated from the results of FASTEST. The fault coverage obtained was 88.2 percent, which is about the same as the coverage obtained in the previous step (88.7 percent).

It is evident from the results shown in Tables 4, 5, and 6 that our algorithm has been able to obtain close to maximum achievable fault coverage for all benchmark circuits in 100,000 test clocks. Clearly, further improvement in fault coverage can be obtained by increasing the test clocks and the number of random vectors.

5 COMPARISON TO RELATED WORK

A number of investigations have been carried out over the past decade on random testing of sequential circuits. The approaches taken in these investigations can be classified into two categories: State hold method and Weighted Random Pattern (WRP) test techniques. In this section, the related work will be briefly reviewed and contrasted to our technique.

5.1 State Hold Method

A method called FREEZE, based on holding the internal state of a sequential circuit to a constant while changing the input, was given in [11]. By temporarily holding the clock inactive, the

TABLE 6
USE OF THE HOLD METHOD ON THE PARTIALLY SCANNED CIRCUITS

Circuit	Random Testing with the Hold Method and the Partial Scan		
	FC (%)	# of Unique Vectors	Values of k
s382	97.5	6,936	2, 3, 4
s444	96.2	6,936	2, 3, 4
s820	96.4	22,223	2, 3
s1423	92.6	859	2, 3, 4, 8
s5378	85.1	1,175	2, 3, 4, 7, 11
s13207	56.6	109	2, 3, 6
s35932	84.9	135	2, 3, 7

TABLE 7
COMPARISON WITH FREEZE

	FC(%)	
Circuit	FREEZE [11]	Hold Method
s420	38.4	40.2
s820	53.6	62.4
s953	8.9	8.3
s1423	53.7	86.3
s5378	59.5	74.4
s13207	5.5	9.2
s35932	85.3	87

sequential behavior of the circuit is disabled, thereby allowing the application of a group of vectors in every state. In this way, one can detect those faults whose detection is conditioned by the circuit being in a given state. The technique allows many faults to be combinationally detected using faster combinational test generation algorithms. However, a major limitation of FREEZE is that the faults that have been activated in previous time frames and whose effects are stored in flip-flops are not detected by this method.

The fault coverages obtained by FREEZE and the hold method are compared in Table 7. Only those circuits for which the coverages with both techniques are known are given in the table. As is evident from the table, the hold method outperforms FREEZE for all circuits, except for s953, which has a very poor coverage even with the best deterministic test generator (see Table 3). An advantage of FREEZE is that the number of test vectors (generated by ATPG) needed to obtain the reported coverages in the table is far less than the number of vectors needed by the hold method (random vectors).

Bardell and Savir [12] also used a method similar to FREEZE to test a class of sequential circuits. In their method, a randomly addressed memory array location is initialized and held for a fixed number of clock cycles, while new random vectors are fed into the combinational logic. The goal of this approach is to test the output logic of products with embedded memory arrays. What makes this technique distinct from FREEZE is that random vectors are applied at the primary inputs, unlike FREEZE, which uses an ATPG. In addition, Bardell and Savir [12] also provide for the collection of output patterns in a signature register to facilitate fault diagnosis.

5.2 Weighted Random Pattern Test Techniques

Wunderlich [10] proposed a two-phase design of random testable sequential circuits in which a minimal set of flip-flops is included in the partial scan path and the modified sequential circuit is tested by weighting the random test inputs. The weights at the primary inputs and pseudoprimary inputs are time dependent. Computing these weights for general sequential circuits such that the test length is reduced significantly has exponential time complexity. Therefore, certain restrictions are placed on the circuit structure so that random pattern testability is ensured. The results on three special types of circuits show significant improvements in fault coverage. However, due to the restricted applicability of this technique, no results are available on the ISCAS-89 benchmarks.

Muradali et al. [22] observed that the low fault coverage of random testing of sequential circuits is the direct result of failure of the internal flip-flops to switch states, owing to their poor controllability. This effectively makes the set of all circuit flip-flops an internal weighted random pattern (WRP) generator. In an unmodified sequential circuit tested with random patterns, this default weight distribution can restrict the range of circuit activity and, hence, impede fault coverage. A flip-flop modification method was proposed in [22] such that the number of circuit states attainable using random patterns is increased. Using this method, the circuit is jolted to produce a larger spread of states so that the bias (the probability that a line value is one) at a flip-flop is close to 0.5. The result is an internally generated sequence of biased test vectors.

Muradali et al.'s WRP experiments used 100,000 random test vectors. The fault coverages obtained for selected ISCAS-89 benchmark circuits are in the range of 85 to 95 percent, whereas the results from the hold method are in the 30 to 80 percent range for the same circuits. However, the better coverage of WRP test approach comes at the cost of modifications to almost all the flip-flops in the benchmark circuits. On the other hand, the fault coverages by the hold method with partial scan compare well with their results. The advantage of our technique is that modification is needed on only a subset of flip-flops, unlike their technique. Further, it must be noted that the weight generation step in [22] requires either testability measures or iterative simulation of the circuit with large number of random vectors.

6 Conclusion and Discussion

In this paper, we proposed a systematic approach to improve the random testing of sequential circuits. Our approach consists of a hold method (where random vectors are held at the inputs for more than one clock cycle) combined with partial scan. The hold method is extremely simple to implement and requires no hardware modification, yet we showed that it is highly effective in improving the fault coverage. Partial scan can be used to further improve the fault coverage, and a combination of the hold method and partial scan can be used when necessary. We used our algorithm on the benchmark circuits. Results of our experiments show that the algorithm achieves a major improvement over conventional random testing. The fault coverage for most of the circuits was pushed close to the maximum obtainable fault coverage using full-scan and random testing. We believe that, for the large benchmark circuits (s5378, s13207, and s35932), using a more realistic number of random vectors will result in maximum obtainable fault coverage.

The hold method has a limitation in that it performs only marginally better than conventional random testing for circuits with high random test coverage to begin with. In such situations, holding a vector does not necessarily sensitize the excited faults any better than the next random vector. Thus, the hold method is targeted to increase the coverage of circuits which have an inherently low random test coverage.

The success of the hold method in random pattern resistant circuits can also be explained in a different way (suggested by one of the referees) than what is given earlier in this paper. If a vector randomly excites a hard-to-excite fault, keeping that hard-to-excite fault excited for multiple clock cycles is more likely to lead to an error than if it is excited for only one clock cycle. How long one should keep the fault excited depends on the structure and function of the circuit. The k metric and the sequential depth discussed earlier in the paper encapsulate such circuit dependability. The analysis of state transition graph of the circuit may give more insight into the evaluation of this metric. Some of the open and interesting problems to solve are the appropriate choice of the number of test vectors (n), selection of flip-flops that form a good set of candidates for inclusion in the scan path for hold method of test-

ing, and the use of even simpler computational methods to estimate the values of *k*. The method proposed here is an excellent candidate for use in built-in self-test (BIST) environment. But, clearly, use of multiple values of *k* can be costly; therefore, methods to choose only few values of *k* while maintaining a high fault coverage is of interest. These problems require further study and improvements in the method proposed in this paper.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous referees for their insightful comments which helped improve the quality of the paper. This work was supported in part by U.S. National Science Foundation grant MIP-9111886 and a grant from the AT&T Foundation. An earlier version of this paper appeared in the Proceedings of the Fault-Tolerant Computing Symposium, 1996.

REFERENCES

[1] K. Parker, "Adaptive Random Test Generation," *J. Design Automation and Fault Tolerant Computing*, vol. 1, pp. 62-68, 1976.
[2] F. Siavoshi, "WTPGA: A Novel Weighted Test Pattern Generation Approach or VLSI Built-In Self-Test," *Proc. Int'l Test Conf.*, pp. 256-262, 1988.
[3] J. Waicukauski and E. Lindbloom, "Fault Detection Effectiveness of Weighted Random Patterns," *Proc. Int'l Test Conf.*, pp. 245-255, 1988.
[4] T. Kelsey, K. Saluja, and S. Lee, "An Efficient Algorithm for Sequential Circuit Test Generation," *IEEE Trans. Computers*, vol. 42, no. 11, pp. 1,361-1,371, Nov. 1993.
[5] I. Pomeranz and S. Reddy, "The Multiple Observation Time Test Strategy," *IEEE Trans. Computers*, vol. 41, no. 5, pp. 627-637, May 1992.
[6] T. Niermann and J. Patel, "HITEC: A Test Generation Package for Sequential Circuits," *Proc. European Conf. Design Automation*, pp. 214-218, 1991.
[7] W. Cheng and T. Chakraborty, "Gentest—An Automatic Test Generation System for Sequential Circuits," *Computer*, pp. 43-49, Apr. 1989.
[8] M. Abramovici, M. Breuer, and D. Friedman, *Digital Systems Testing and Testable Design*. Computer Science Press, 1990.
[9] V. Agrawal, K. Cheng, D. Johnson, and T. Lin, "Designing Circuits with Partial Scan," *IEEE Design and Test of Computers*, pp. 8-15, Apr. 1988.
[10] H. Wunderlich, "The Design of Random-Testable Sequential Circuits," *Proc. Int'l Symp. Fault-Tolerant Computing*, pp. 110-117, June 1989.
[11] M. Abramovici, K. Rajan, and D. Miller, "FREEZE: A New Approach for Testing Sequential Circuits," *Proc. European Conf. Design Automation*, pp. 22-25, 1992.
[12] P. Bardell and J. Savir, "Test and Diagnosis of Associated Output Logic for Products Having Embedded Arrays," U.S. Patent No. 5442640, Aug. 1995.
[13] L. Nachman, K.K. Saluja, S. Upadhyaya, and R. Reuse, "Random Pattern Testing of Sequential Circuits Revisited," *Proc. Int'l Symp. Fault-Tolerant Computing*, pp. 44-52, June 1996.
[14] K. Kim and C. Kime, "Partial Scan Using Reverse Direction Empirical Testability," *Proc. Int'l Test Conf.*, pp. 498-506, 1993.
[15] K.-T. Cheng and V. Agrawal, "An Economical Scan Design for Sequential Logic Test Generation," *Proc. Int'l Symp. Fault-Tolerant Computing*, pp. 28-35, 1989.
[16] L.H. Goldstein and E.L. Thigpen, "SCOAP: Sandia Controllability/Observability Analysis Program," *Proc. Design Automation Conf.*, pp. 190-196, 1980.
[17] K. Cheng and V. Agrawal, "Concurrent Test Generation and Design for Testability," *Proc. Int'l Symp. Computer Automated Systems*, pp. 1,935-1,938, 1989.
[18] N. Gouders and R. Kaibel, "Advanced Techniques for Sequential Test Generation," *Proc. European Test Conf.*, pp. 293-300, 1991.
[19] D. Lee and S.M. Reddy, "A New Test Generation Method for Sequential Circuits," *Digest Int'l Conf. Computer-Aided Design*, pp. 446-449, 1991.
[20] B. So, "Time Efficient Automatic Test Pattern Generation Systems," PhD thesis, Dept. of Electrical and Computer Eng., Univ. of Wisconsin-Madison, 1994.
[21] P. Maxwell and R. Aitken, "All Fault Coverages Are Not Created Equal," *IEEE Design and Test of Computers*, vol. 10, pp. 42-51, Mar. 1993.
[22] F. Muradali, T. Nishida, and T. Shimizu, "A Structure and Technique for Pseudorandom-Based Testing of Sequential Circuits," *J. Electronic Testing: Theory and Application*, vol. 6, pp. 107-115, 1995.