

BIST/TEST-DECOMPRESSOR DESIGN USING COMBINATIONAL TEST SPECTRUM

Nitin Yogi¹ and Vishwani D. Agrawal¹

Abstract

ATPG vectors for a combinational circuit exhibit correlations among the bits of a test vector. We propose a BIST/decompressor circuit design methodology using spectral methods which utilizes the correlation information. This circuit serves dual purposes. It generates BIST vectors that are similar to the ATPG vectors with higher test coverage as compared to random and weighted random vectors. The same circuit can also function as a test data decompressor for compressed ATPG vectors applied from an external tester. The proposed design method consists of spectral analysis of ATPG vectors to determine prominent spectral components and a vector shuffling algorithm to minimize noise. A BIST/decompressor circuit is then constructed using the spectral information and the noise level. For ISCAS'85 circuit c7552 and the combinational part of ISCAS'89 circuit s15850 we compare the new methodology against ATPG, and random or weighted random BIST vectors with respect to test coverage, test data volume, test application time and area overhead. For test application time, we assume that the on-chip system clock is ten times faster than the external tester clock. For c7552, the pure BIST mode achieves test coverage of about 99.25% with zero external test data volume in the same test time as that for external application of ATPG vectors having 100% coverage. Using the decompressor mode, when compressed ATPG vectors are applied from an external tester, we achieve 100% coverage with test data compressed to around 5%. In a hybrid mode, where some compressed external ATPG vectors serve as seeds for BIST, we again achieve 100% test coverage with test data volume reduced to around 1.5%, in comparison to external ATPG test vectors. The area overhead of the proposed BIST/decompressor circuit is similar to that of random and weighted random pattern BIST.

Keywords: BIST, Test decompressor, Test pattern generator, Spectral testing

1. Introduction:

Built-In Self-Test (BIST) has been a popular approach for testing digital circuits, which employs an on-chip test pattern generator and a response analyzer to test the Circuit-Under-Test (CUT). The BIST approach exhibits several advantages, such as eliminating the need for expensive external testers, reducing testing time, reducing test data volume, providing vertical testing capability from wafer to system-level, and several others.

¹ Auburn University, Dept. of ECE, 200 Broun Hall, Auburn University, AL 36849, USA; Email: yoginit@auburn.edu, vagrawal@eng.auburn.edu.

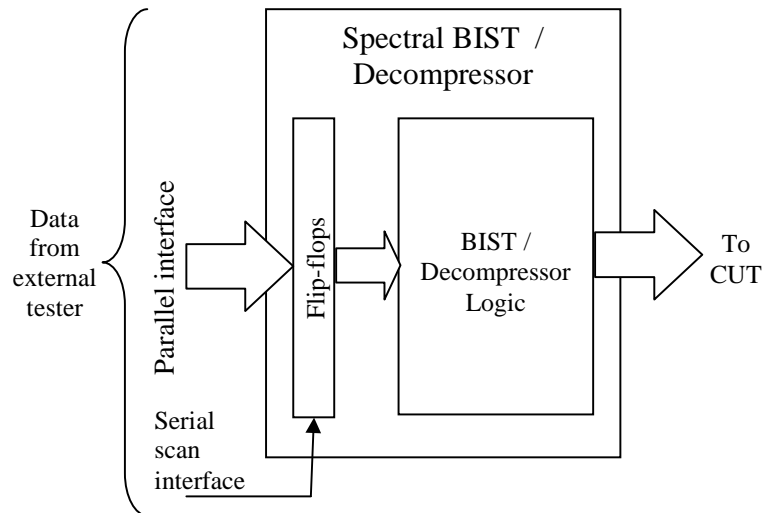


Figure 1: Block diagram of proposed spectral BIST/decompressor.

One of the main challenges for BIST has been the design of a potent test pattern generator (TPG) that can generate test vectors having satisfactory fault coverage. A simple TPG that is widely used is a pseudo-random pattern generator (PRPG) constructed as either a Linear Feedback Shift Register (LFSR) or a Cellular Automata Register (CAR) (Bushnell and Agrawal (2000), McCluskey (1985)). Although a PRPG covers most faults, its coverage is limited by random pattern resistant faults that require specific test vectors (Eichelberger and Lindbloom (1983)). Several approaches have been proposed for covering these faults. Mixed-mode testing is one way to cover these faults; in which testing is first performed using pseudo-random patterns and then deterministic test patterns obtained from an Automatic Test Pattern Generator (ATPG) are applied to detect the remaining random pattern resistant faults. The test data storage requirement is determined by the number of ATPG patterns to be applied and can be prohibitively large. One of the early suggestions was to use weighted random test vectors, where the weighting of the bits (probability of being logic '1') is modified to detect the random pattern resistant faults (Eichelberger, Lindbloom, Motica, and Waicukauski (1989)). However, multiple combinations of weight sets may be required, which limits the applicability of this approach (Wunderlich (1990)). Another approach is to insert test points for improving the controllability and observability of hard-to-detect faults (Touba and McCluskey (1996)). This approach incurs area and delay penalties that might be unacceptable for certain designs.

A popular approach is reseeding of the LFSR such that relevant states are loaded in the LFSR emulating the deterministic patterns that detect the hard-to-detect faults (Koenemann (1991)). This approach covers the random pattern resistant faults and also provides encouraging test data compression capabilities. To determine the seed for a test cube, a set of linear equations is solved based on the polynomial of the LFSR. It has been shown that a testcube of length S_{max}

can be encoded with a PRPG of size $S_{max} + 20$ with high probability (Koenemann (1991)).

All the above described approaches are based on the LFSR as a TPG. In this paper we propose spectral techniques to design a TPG/test decompressor, which generates test vectors for a combinational circuit in BIST environment. The designed hardware recreates the essential spectral properties of ATPG vectors. Figure 1 shows the block diagram of the proposed BIST/test-decompressor hardware. The hardware can be viewed as two sub-modules, a set of flip-flops and combinational logic. As will be shown in the results section, the number of flip-flops in the BIST hardware is much smaller than the number of inputs of the CUT being driven, which gives this architecture interesting test data decompression capabilities.

The proposed hardware can function in three modes; pure BIST mode, external tester mode (ETM) and hybrid BIST mode (HBM). The salient parameters by which we can compare these modes of operation are fault coverage, test data volume and test application time. In **pure BIST mode**, there is no data being supplied from the external tester and the TPG generates test vectors for the CUT. This mode leads to lower fault coverage than externally applied test vectors and has ideally zero test data volume. Its test application time is determined by the required fault coverage and the frequency of operation of the on-chip system clock. In **External Tester Mode (ETM)**, seed vectors are loaded from an external tester into the set of flip-flops (either through a serial scan interface or through a parallel interface) and are decompressed by the BIST logic and applied to the inputs of the CUT. The salient feature of this mode is high fault coverage. Test data volume depends on the number of flip-flops to be loaded and the number of seed vectors. Test application time is determined by the number of decompressed test vectors and the frequency of operation of the external tester clock. In **Hybrid BIST Mode (HBM)** the TPG generates test vectors for the CUT and is loaded with relevant seeds in the flip-flops at frequent intervals to improve its performance. This mode is marked with high fault coverage, lower test data volume and test application time as compared to the external tester mode.

2. Background

Our BIST synthesis approach is based on the premise that the spectrum of vectors that detect faults in a circuit reflects important characteristics such as spatial and temporal correlations among the bits of primary input vectors along with some amount of noise or randomness, which corresponds to uncorrelated bits in the test. We use Walsh functions (Weisstein (2009)) contained in Hadamard matrices to spectrally analyze binary bit-streams. Walsh functions are a set of orthogonal functions that consist of +1s and -1s. Any binary bit-stream of k bits can be represented as a linear combination of Walsh functions contained in the Hadamard matrix, $H(\log_2 k)$. Hence, by spectral analysis of test vector bit-streams applied to the CUT the principal contributing Walsh functions for each input of the CUT can be determined, as described in the literature (Giani, Sheng, Hsiao and Agrawal (2001), Yogi and Agrawal (2006)).

3. Proposed Spectral BIST Method

One may use any commercial or home-grown ATPG tool, such as ATALANTA (Lee and Ha (1993)) in our case, to generate test vectors for the CUT. The goal is then to regenerate similar vectors in hardware using minimum area overhead such that the original fault coverage of the ATPG vectors is achieved.

The proposed method consists of two steps. In the first step the ATPG vectors are processed through shuffling and Hadamard transform analysis to enhance the inherent spectral components and determine the remaining random (noise-like) content. Using the extracted spectral information and noise content, a BIST circuit is implemented in the second step.

3.1 Vector Processing using Spectral Analysis

In this step, the ATPG vectors are analyzed using Hadamard transform. Since the order of the vectors for combinational circuits is immaterial, vectors can be shuffled and inherent spectral properties can be amplified, thus reducing the ambiguous noise-like content. Another technique that has been found useful in extracting the spectral components unambiguously is to append the original vector set with vectors such that the aggregate weighting of the bit-stream entering each primary input is balanced to 0.5, i.e., equally probable logic '0' and logic '1'.

To analyze the ATPG vectors, bit-streams entering various inputs of the CUT are examined separately. The 0s and 1s in a bit-stream are represented as -1s and +1s, respectively. Spectral analysis is performed using Hadamard transform (Yogi and Agrawal (2006)) to obtain the components of Walsh spectrum in each bit-stream. The shuffling algorithm reorders the vectors such that certain spectral components are amplified and noise is reduced. The shuffling algorithm is described below.

Shuffling algorithm:

N_I : No of inputs

N_V : No. of vectors

$V(1:N_V, 1:N_I)$: Vector Set of dimensions $N_V \times N_I$

hd : Dimension of Hadamard matrix

H : Hadamard transform matrix of dimension $2^{hd} \times 2^{hd}$

Vector set V is appended with redundant vectors such that the weighting of the bit-streams of all inputs is 0.5

for $i=1$ to N_I

perform spectral analysis on bit-stream of input i : $S = V(:, i) \times H$;

pick the prominent spectral component $Sp(i)$ from S

rearrange vector set V such that maximum bits in the bit-streams of inputs 1 to i match with the picked prominent spectral components $Sp(1$ to $i)$ respectively.

end

Perform spectral analysis for the bit-streams of all inputs to determine the prominent spectral components and noise.

After executing the shuffling algorithm, prominent spectral components for each input are obtained along with the corresponding noise.

3.2 Spectral BIST Implementation

The goal of BIST implementation is to design hardware that will generate vectors exhibiting similar spectral properties as the original ATPG vectors. This is achieved by combining the chosen M prominent components in appropriate proportions and phases using a “spectral component synthesizer”. Randomness or noise, if required, is inserted in appropriate amount to the generated vectors using a “randomizer circuit”. Figure 2 shows the proposed spectral BIST architecture for a CUT with three inputs. This architecture, described in our earlier work (Yogi and Agrawal (2008)) for sequential circuits, is repeated here for completeness.

A Hadamard wave generator provides spectral components that are combined by component synthesizers (generally, one per PI) using random bit-streams from the weighted pseudo-random bit-stream generator. The spectral component signals are shown in solid bold lines while the weighted random signals are in dotted bold lines. Noise is inserted in the combined spectral components by a randomizer (also one per PI) supplied with an appropriate weighted random bit-stream. In this example, the first input of the CUT has three prominent components, which are combined by the component synthesizer. The second input of the CUT has only one prominent component; hence no component synthesizer is required. A randomizer adds the required amount of noise. The third input of the CUT has no prominent component and hence a random bit-stream is directly fed from the pseudo-random bit-stream generator.

3.2.1 Hadamard Wave Generator

There is much literature on Walsh function generators (Harmuth (1972)), which provide implementations giving trade-offs between speed and area. We use a Walsh function generator proposed by Harmuth (Harmuth (1972)) that uses a counter and has low hardware overhead. A generator of order N generates 2^N Walsh functions. It requires N flip-flops and $2^N - N - 1$ XOR gates. The order N of Hadamard matrix is used in spectral analysis and to implement the Hadamard

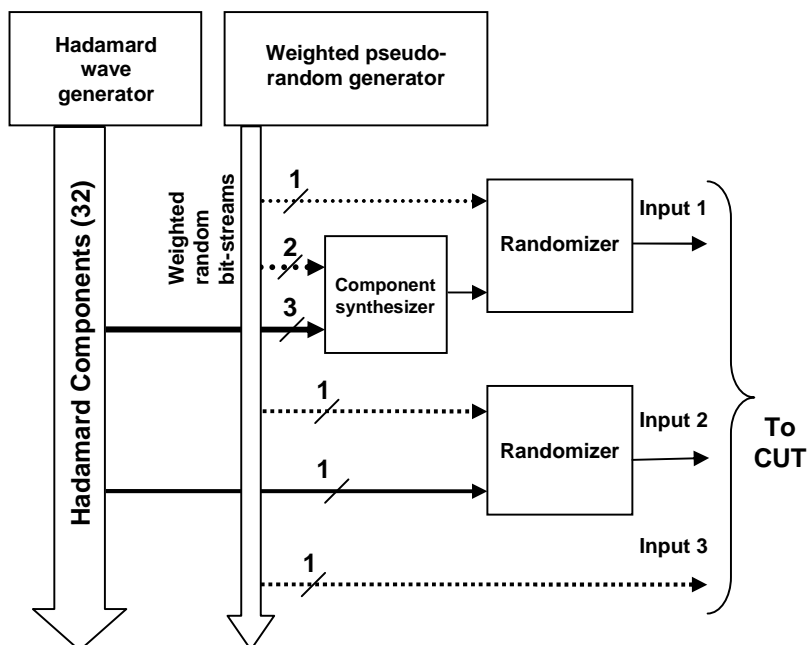


Figure 2: Proposed spectral BIST architecture.

wave generator. Higher order Hadamard matrices are constructed from lower order matrices (Weisstein (2009)). Higher order matrices are better able to characterize a given bit-stream than lower order matrices. However, the area overhead for implementing the Hadamard wave generator increases with the order of the Hadamard matrix as mentioned earlier. We choose a value of N optimally to give a good trade-off between performance and area. This value is determined from the ATPG vectors.

3.2.2 Spectral Component Synthesizer

To synthesize vectors, M prominent spectral components are combined in required proportions (equal to their relative power magnitudes) and phases (their signs), both obtained from spectral analysis of Section 3.1. This is achieved using the “spectral component synthesizer” that is a multiplexer structure whose inputs are driven by M chosen spectral components, generated by the Hadamard wave generator, and the select lines are driven by weighted random bit-streams. The weighting of the bit-stream is the proportion in which the components are to be combined. For our experiment we choose $M = 1$.

Table 1: Details of circuits used as examples.

Circuit	No. of inputs	No. of outputs	No. of gates	No. of collapsed faults
c7552	207	108	3512	7550
s15850 (combinational)	600	670	9772	11697

Table 2: Details of implemented spectral BIST TPG.

Circuit	Hadamard dimension	Cellular Automata Size	Total flip-flops used	No. of gates
c7552	6	24	30	856
s15850 (combinational)	7	28	35	2532

3.2.3 Randomizer

Along with the prominent spectral components, some randomness or noise is present in the ATPG vectors. That needs to be inserted in the regenerated vectors. The level of noise to be inserted is estimated from the relative power magnitudes of the prominent spectral components and the other noise components. The perturbation is inserted in the generated vectors using the randomizer, which performs an XOR operation on selected spectral component(s) with a weighted random bit-stream. The weight is proportional to the amount of noise needed.

3.2.4 Weighted Pseudo-Random Bit-Stream Generator

This circuit generates weighted pseudo-random bit-streams required by the component synthesizer and randomizer. It uses a Cellular Automata Register (CAR) having L -flip flops and a combination of AND-OR gates. The different weights to be generated are determined by the mixing proportions of the spectral components and by the amount of randomness to be added. We use the quantized weights given by $W = (i \times 2^{-w})$ for $i = 0$ to $2^w - 1$. We use a value of $w = 4$ and also generate two additional weights of 2^{-6} and 2^{-8} for the randomizer. The size L of the CAR needs to be sufficiently large to prevent any unwanted output bit correlations arising due to structural dependencies of the implemented BIST logic. Currently we choose a value of L such that all faults in the CUT are testable in the presence of the BIST logic. This is determined by providing the combined BIST logic and CUT circuitry to an ATPG program.

4. Results

We implemented the proposed BIST methodology on two circuits; ISCAS'85 c7552 and the combinational part of ISCAS'89 s15850 benchmark circuits. Table 1 gives the details of the two circuits. ATPG vectors were generated for stuck-at faults with don't care bits using the program ATALANTA (Lee and Ha (1993)). The test coverage of ATPG vectors was 100% since redundant faults were not considered. These ATPG vectors were processed using spectral analysis and the shuffling algorithm of Section 3 to obtain the spectral information, which was then used to construct the BIST TPG hardware. The BIST TPG hardware was synthesized using the tool Mentor Graphics Leonardo Spectrum in TSMC 0.18 μ m technology library.

Table 2 gives the details of the synthesized BIST circuits. The BIST TPG along with benchmark circuits were then fault simulated using Mentor Graphics tool FlexTest (Mentor Graphics (2004)) and test coverages were determined.

Table 3: Test coverage comparison of random, weighted random and proposed spectral BIST method after applying 64,000 vectors.

Circuit	Random vectors	Weighted Random vectors	Spectral BIST
c7552	97.41%	97.86%	99.81%
s15850 (combinational)	96.81%	97.41%	98.77%

Table 3 gives the test coverage results of the proposed method. We compare with random and weighted random pattern generators for 64,000 vectors. Weights for the random vectors were obtained from the corresponding ATPG vectors without quantization. Random and weighted random vectors were generated using a software random number generator. As shown in Table 3, the proposed spectral TPG obtains better test coverage for both circuits.

Figures 3 and 4 show test coverages for circuits c7552 and s15850 (combinational) with the number of test vectors applied for random, weighted random and the proposed Spectral BIST method. The figures also show the graph of test coverages when reseeding is employed using 33 and 134 seeds, respectively, for c7552 and s15850 circuits. As can be observed, the test coverage of the test vectors generated by our spectral BIST TPG increases more rapidly than those with random and weighted random test vectors.

Table 4 gives the number of gates of the original circuit, the area overhead of the proposed method and the area overhead of the pseudo-random pattern generator (PRPG). For the PRPG, the area overhead comprises of the size of the LFSR (number of flip-flops) which is proportional to the number of inputs of the CUT. As can be observed the area overhead of our proposed method is comparable to that of the pseudo-random pattern generator.

As mentioned earlier, our proposed BIST/test-decompressor architecture works in a mode that can load external test data into its flip-flops. The two modes that use this feature are External Tester Mode (ETM) and Hybrid BIST Mode (HBM). The flip-flops used in the Hadamard wave generator and the weighted

Table 4: Area overhead comparison of proposed spectral BIST and pseudo-random pattern generator (PRPG).

Circuit	No. of gates in circuit	Spectral BIST		PRPG	
		No. of gates	% Area overhead	No. of gates	% Area overhead
c7552	3,513	976	27.78	830	23.63
s15850 (combinational)	9,772	2,672	27.34	2,400	24.56

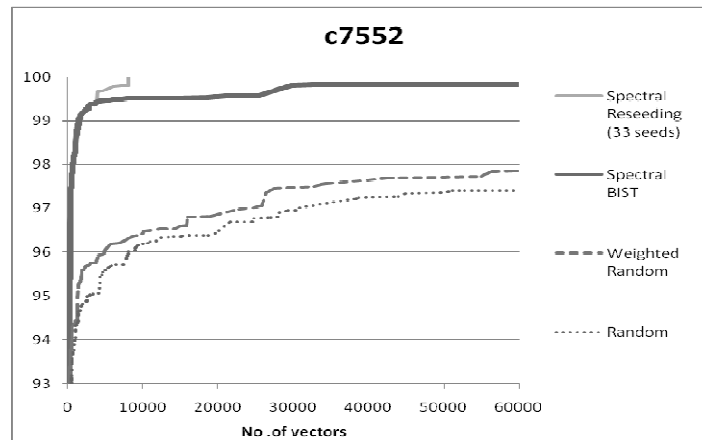


Figure 3: Test coverage comparison for c7752 of random, weighted random, spectral BIST and spectral BIST with reseeding.

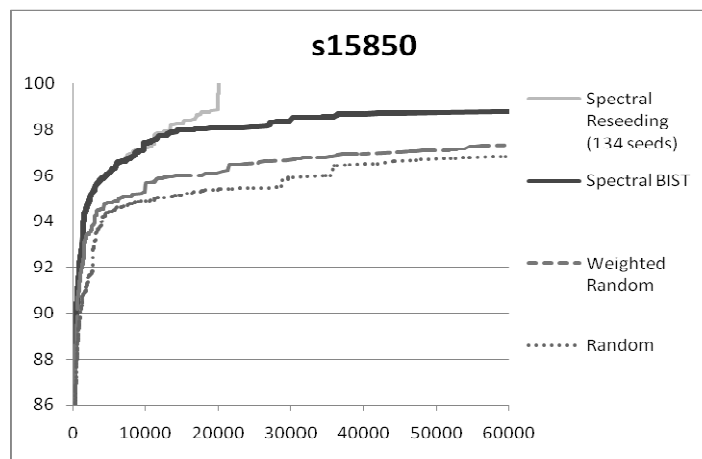


Figure 4: Test coverage comparison for s15850 of random, weighted random, spectral BIST and spectral BIST with reseeding.

pseudo-random bit-stream generator, indicated in column 4 of Table 2, are loaded with required seed from the external tester. Since only these two modes are able to provide 100% test coverage (fault coverage of the pure BIST mode saturates below 100%), we compare these modes in terms of test data volume and test application time. The seeds can either be loaded through a serial scan interface or through a parallel interface. Several approaches have been proposed for parallel reseeding including on-chip dynamic techniques (Kalligeros, Kavousianos and Nikolos (2004)).

As discussed earlier, in the External Tester Mode (ETM), the TPG is used as a decompressor for one-seed-per-vector operation. In this mode the TPG is not used to generate any new vectors, but only to decompress the seeds that are being applied to its inputs. In Hybrid BIST Mode (HBM), the TPG is operated in the normal mode to generate new vectors and is reseeded at specific intervals

to improve the fault coverage. The specific intervals for reseeding are determined by a coverage threshold and a step-size (number of vectors). Whenever the test coverage improvement over a given step-size falls below threshold, reseeding is performed. We use a test coverage improvement threshold of 0.1% for a step-size of 1000 vectors. When the number of faults detected by 1000 vectors drops to one, the TPG is run only for one clock cycle after reseeding till all the faults are detected. The next test seed to load for reseeding is obtained using an ATPG program run on the remaining undetected faults in the CUT.

Table 5 gives a comparison of test data volume and test time for ATPG and the different modes of operation of Spectral BIST for the circuit c7552. For ATPG, the test vectors are applied directly to the CUT (without any BIST logic) by the tester with a clock period equal to T_{tester} . The test vectors can either be applied through a parallel interface or scanned in into a shift register of size equal to the number of inputs of the CUT. In external tester mode (ETM), the TPG functions as a decompressor for one-seed-per-vector operation and is controlled by the tester clock (T_{tester}). Reseeding can be performed using a serial or parallel interface. In the hybrid BIST mode (HBM), TPG runs on internal on-chip system clock (T_{clk}) during normal mode of operation, and on the tester clock (T_{tester}) during reseeding. Again reseeding can be performed in this mode through a serial or parallel interface.

In Table 5, column 1 gives the number of vectors/seeds that need to be applied for 100% test coverage. Column 2 gives the no. of inputs that are being driven by the tester. For ATPG, the tester drives the inputs of the CUT directly. For Spectral BIST, the tester drives the inputs of the BIST logic. Column 3 gives the test data volume that needs to be applied. Columns 4 and 5 give the number of clock cycles of the tester and the CUT that are required to apply the test data. Column 6 gives test application time in microseconds (μs). For calculating test time, it is assumed that the period of the tester clock is $T_{\text{tester}} = 10\text{ns}$ and that of

Table 5: Comparison of test data volume and test time for ATPG and different modes of operation of Spectral BIST for c7552.

	(1)	(2)	(3)	(4)	(5)	(6)	
Mode of test application	No. of vecs./ seeds	No. of inputs	Test data volume (bits)	No. of tester cycles	No. of system clock cycles	Test time (μs) [†]	
ATPG (parallel)	247	207	51129	247	0	2	
ATPG (serial)	247	1	51129	51129	0	511	
Spectral BIST	ETM (parallel)	197	30	5910	197	0	2
	ETM (serial)	197	1	5910	5910	0	59
	HBM (parallel)	33	30	990	33	8034	8
	HBM (serial)	33	1	990	990	8034	18

[†] assuming tester clock period $T_{\text{tester}} = 10\text{ns}$ and on-chip system clock period $T_{\text{clk}} = 1\text{ns}$.

Table 6: Comparison of test data volume and test time for ATPG and different modes of operation of spectral BIST for s15850 (combinational).

		(1)	(2)	(3)	(4)	(5)	(6)
Mode of test application		No. of vecs./seeds	No. of inputs	Test data volume (bits)	No. of tester cycles	No. of system clock cycles	Test time (μ s) [†]
ATPG (parallel)		530	600	318000	530	0	5
ATPG (serial)		530	1	318000	318000	0	3180
Spectral BIST	ETM (parallel)	455	35	15925	455	0	5
	ETM (serial)	455	1	15925	15925	0	159
	HBM (parallel)	134	35	4690	134	20129	21
	HBM (serial)	134	1	4690	4690	20129	67

[†] assuming tester cycle period $T_{\text{tester}} = 10\text{ns}$ and on-chip system clock period $T_{\text{clk}} = 1\text{ns}$.

the system clock is $T_{\text{clk}} = 1\text{ns}$. We chose a much slower tester clock for several reasons. The scan circuitry is generally not optimized for high speed operation. Also, using a fast tester clock may increase test power to undesirable levels. Since the test vectors generated are non-functional, there is a probability of failing functionally good chips when running the tester clock at higher frequencies by activating long non-functional paths.

The results for parallel interface and serial scan interface are given in Table 5. We observe that the test data volume for spectral BIST is an order of magnitude lower than direct application of ATPG vectors. The test application time shows a marked reduction in case of serial interface reseeding for the two BIST modes as compared to the ATPG mode. The test application time does not show any improvements for the parallel interface reseeding and is found to actually increase slightly in the HBM parallel mode, although the test data volume is reduced even further. We believe using an improved algorithm in the HBM mode for picking the optimum intervals for reseeding and determining an optimum mix of tester clock and system clock cycles will benefit the results. Table 6 gives a comparison of test data volume and test application time for the circuit s15850 (combinational), which shows a very similar trend.

6. Conclusion

Our BIST methodology implements TPGs for combinational circuits using spectral techniques. Spectral properties of ATPG vectors are obtained by spectral analysis and executing a shuffling algorithm. These properties are used to design a TPG in hardware which generates vectors having similar spectral properties as ATPG vectors. The proposed method was employed to design two ISCAS benchmark circuits which yielded encouraging results. The spectral TPG obtained better fault coverages than the conventional random and weighted

random TPGs for similar number of applied test vectors. We also exhibit test data compression capabilities of the proposed BIST architecture. This architecture provides a maximum test data compression exceeding 90% and a proportional test time reduction for serial interface reseeding.

References

- [1] M. L. Bushnell and V. D. Agrawal (2000), *Essentials of Electronic Testing for Digital, Memory & Mixed-Signal VLSI Circuits*, Springer.
- [2] E. B. Eichelberger and E. Lindbloom (1983), "Random-Pattern Coverage Enhancement and Diagnosis for LSSD Logic Self-Test," *IBM J. Res. Develop.*, vol. 27, no. 3, pp. 265–272.
- [3] E. B. Eichelberger, E. Lindbloom, F. Motica, and J. Waicukauski (1989), "Weighted Random Pattern Testing Apparatus and Method," U.S. Patent 4,801,870, January.
- [4] A. Giani, S. Sheng, M. S. Hsiao, and V. D. Agrawal (2001), "Novel Spectral Methods for Built-In Self-Test in a System-on-a-Chip Environment," *Proc. 19th IEEE VLSI Test Symp.*, pp. 163–168.
- [5] H. F. Harmuth (1972), *Transmission of Information by Orthogonal Functions*, Springer-Verlag.
- [6] B. Koenemann (1991), "LFSR-Coded Test Patterns for Scan Design," *Proc. European Test Conf.*, pp. 237–242.
- [7] H. K. Lee and D. S. Ha (1993), "ATALANTA: An Efficient ATPG for Combinational Circuits," Dept. of Elect. Eng., Virginia Polytechnic Inst. and State Univ., Blacksburg, VA, Tech. Rep. 93-12.
- [8] E. J. McCluskey (1985), "Built-In Self-Test Techniques," *IEEE Design & Test of Computers*, vol. 2, no. 2, pp. 21–28, Apr. 1985.
- [8] Mentor Graphics (2004), *FastScan and FlexTest Reference Manual*, 2004.
- [10] N. A. Toubia and E. J. McCluskey (1996), "Test Point Insertion Based on Path Tracing," *Proc. VLSI Test Symp.*, 1996, pp. 2–8.
- [11] E. W. Weisstein (2009). From MathWorld—A Wolfram Web Resource. <http://mathworld.wolfram.com/search/?query=walsh+functions&x=10&y=13>
- [12] H. J. Wunderlich (1990), "Multiple Distributions for Biased Random Test Patterns," *IEEE Trans. on CAD*, vol. 9, no. 6, pp. 584–593, June.
- [13] N. Yogi and V. D. Agrawal (2008), "Sequential Circuit BIST Synthesis Using Spectrum and Noise from ATPG Patterns," *Proc. 27th IEEE Asian Test Symp.*, pp. 69–74.
- [14] N. Yogi and V. D. Agrawal (2006), "Spectral RTL Test Generation for Gate-Level Stuck-at Faults," *Proc. 15th IEEE Asian Test Symp.*, pp. 83–88.