Spectral RTL Test Generation for Gate-Level Stuck-at Faults

Nitin Yogi* and Vishwani D. Agrawal
Auburn University, Department of Electrical and Computer Engineering, Auburn, AL 36849, USA
yoginit@auburn.edu, *vagrawal@eng.auburn.edu*

Abstract

We model RTL faults as stuck-at faults on primary inputs, primary outputs, and flip-flops. Tests for these faults are analyzed using Hadamard matrices for Walsh functions and random noise level at each primary input. This information then helps generate vector sequences. At the gate-level, a fault simulator and an integer linear program (ILP) compact the test sequences. We give results for four ITC'99 and four ISCAS'89 benchmark circuits, and an experimental processor. The RTL spectral vectors performed equally well on multiple gate-level implementations. Compared to a gate-level ATPG, RTL vectors produced similar or higher coverage in shorter CPU times.

1. Introduction

Conventionally, test vectors are generated at the gate-level, i.e., after synthesis has been performed. Though this methodology has the advantage of being able to generate reliable, high fault coverage test vectors due to its direct use of the stuck-at fault model, it suffers from several disadvantages. For large circuits, the large number of faults and the algorithm complexity make the gate-level test generation time consuming and expensive. Since gate-level test generation is performed at a later stage in the design process it is difficult to deal with testability issues, revealed during test generation, in an already verified design. Also, the gate-level ATPG cannot be used for cores or circuits for which only the functional information is available. This scenario is frequently encountered in commercial environments.

RTL (register transfer level) or synthesis independent test generation eliminates the disadvantages of gate-level test generation discussed above. Several RTL test generation methods have been proposed. Ravi and Jha [22], Ghosh and Fujita [8], Kim and Hayes [18] and Goloubeva *et al.* [11] use pre-computed test sets for RTL constructs like adders, multiplexers etc. and derive test vectors for the whole RTL circuit. Pre-computed test sets either make some assumptions about the synthesis of the design or use a superset of the actually required test vectors. All of them use some kind of data structure or metrics to derive the RTL test sets, which have implications of large memory and computation overheads. Thaker *et al.* [24] show that a set of stuck-at faults of variables in high-level synthetic operators and at the boundaries of RTL modules serve as a statistical sample for the gate-level coverage analysis.

Recent work by Kang *et al.* [14] uses a set of stuck-at faults called the *sensitization faults* which consider the detection of each primary input fault separately at every primary output. The coverage of sensitization faults is shown to correlate well with the stuck-at fault coverage in any gate-level implementation. However test generation effort is higher for such faults as compared to ordinary stuck-at faults.

The gate-level spectral methods of test generation are relevant to the present research. In 1983, Susskind [23] showed that Walsh spectrum can be used for testing a digital circuit. General properties and applications of digital spectra can be found in the published literature [1, 6, 13, 25]. Hsiao and Seth [12] further expanded that work to compact testing. More recently, Giani *et al.* [9, 10] have reported spectral techniques for sequential ATPG and built-in self-test. Hsiao's group at Virginia Tech has published further work on spectrum-based self test and core test [2, 3, 16]. Khan and Bushnell [17] have designed hardware signature analyzers using spectral components. Zhang *et al.* [29] further refined the method of extracting the spectra from a digital signal using a selfish gene algorithm. Recent work suggests that wavelet transforms can also be used for similar application [4].

In this paper, we present a spectral method of generating test vectors for sequential circuits using only RTL faults, which are faults on the inputs and outputs of the circuit and inputs and outputs of the flip-flops (since they remain invariant through synthesis). Our spectral analysis determines the prominent digital function components and the noise level in the RTL vectors. Vector sequences generated from these properties are found to detect almost as many faults as any gate-level

*Parts of this work were presented as a student paper at *15th IEEE North Atlantic Test Workshop* [27].

ATPG. Besides, the sequences can be compacted to about the same size as that produced by the gate-level ATPG. In the RTL method the use of gate-level ATPG is eliminated and only a fault simulator is used.

In this paper, Section 2 gives an overview of bit-stream analysis in the spectral domain. Section 3 presents our method of RTL ATPG using spectral analysis, with results discussed in Section 4. We conclude in Section 5.

2. Background

Our method of test generation is based on the premise that the spectrum of vectors that detect high-level faults of the circuit reflects important characteristics of the circuit. These characteristics may include spatial and temporal correlations among the bits of primary input vectors and the necessary vector sequence length to sensitize paths between primary inputs and outputs of a sequential circuit. However, any high level test sequence has, besides the relevant spectra, some amount of noise, which corresponds to the don't care bits in the tests of target faults. We analyze the spectrum and the noise level, and generate new vectors using the spectrum, to which noise samples are added.

We use frequency decomposition, in which any bit-stream or signal can be projected on or represented using a set of orthogonal functions. The projections of the signal on each of the functions give the contribution of the corresponding functions to the original signal. We shall use Walsh functions [26] because they have been used for testing with effective results.

Walsh functions are a set of orthogonal functions. They consist of trains of square pulses having +1s and -1s as the allowed states and can only change at fixed intervals of a unit time step. For an order n, i.e., for a sequence of n time steps, there are 2^n Walsh functions given by the rows of a $2^n \times 2^n$ Hadamard matrix $H(n)$ [26], when arranged in the so-called "sequency" order [26]. The Hadamard matrix is a symmetric matrix with each row being a unique Walsh orthogonal function, also called as the basis function *bit-streams*. Since it consists of only +1s and −1s, it is a good choice for the signals in VLSI testing (+1 = logic 1, −1 = logic 0). Also multiplications can essentially be computed using additions and subtractions only. Hadamard matrices are square matrices containing only +1 and −1 elements and can be generated using the following recurrence relation:

$$H(n) = \begin{bmatrix} H(n-1) & H(n-1) \\ H(n-1) & -H(n-1) \end{bmatrix} \qquad (1)$$

where $H(0) = 1$ and 2^n is the dimension of the nth order Hadamard matrix, $H(n)$. For example, for $n = 1$ and $n = 2$, we have:

$$H(1) = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \text{ and } H(2) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \qquad (2)$$

The Hadamard matrix is an *orthogonal matrix*, which gives $H(n) \times H(n)^T = nI_n$, where I_n is the $2^n \times 2^n$ identity matrix. This simplifies reconstruction of the test vectors from the spectral domain. Any bit-stream of k bits can be represented as a linear combination of the basis bit-streams from the Hadamard matrix, $H(\log_2 k)$. The multiplicands used are the projections of the object bit-stream on the basis bit-streams. We shall refer to them as coefficients. By analyzing these coefficients we will be able to determine the major contributing basis bit-streams in an original signal, which we shall regard as important basis bit-stream functions.

3. Spectral RTL ATPG

Our approach to RTL test generation consists of two principal steps described in this section.

3.1 Spectral Characterization

The RTL faults considered are the stuck-at faults on primary inputs and outputs of the circuit and on inputs and outputs of all flip-flops. These faults remain invariant through logic synthesis. We obtain test vectors to detect RTL faults. These vectors are analyzed using Hadamard matrix to find the major spectral components. To analyze a vector sequence, the bit-streams entering various inputs are analyzed separately. The 0s and 1s in a bit-stream are represented as −1s and +1s. To find the coefficients for the bit-stream corresponding to an input, the bit-stream is multiplied with the Hadamard matrix. The multiplication operation is basically a correlation of the bit-stream with each of the basis bit-streams. A high value of the coefficient corresponds to a high correlation of the bit-stream to the corresponding basis bit-stream and vice-versa. Hence basis bit-streams exhibiting high coefficient values are considered as important or essential components and others are considered as noise.

Figure 1 shows an example of generation of coefficients by projecting a bit-stream onto the basis bit-streams and determining the essential component(s). In the example, an 8 bit-stream (0s and 1s in the original sequence being represented as −1s and +1s) is analyzed by multiplying by a third order 8×8 Hadamard matrix. The corresponding result gives the coefficients. As shown, we obtained a single coefficient with high correlation, which we shall treat as an essential component and others will be treated as noise. Figure 2

$$\begin{array}{ccccccccc}
\begin{matrix}1\\0\\1\\1\\1\\0\\1\\0\end{matrix} & 0 \rightarrow -1 & \begin{matrix}1\\-1\\1\\1\\1\\-1\\1\\-1\end{matrix} & \begin{matrix}Spectral\\ decomposition:\end{matrix} & \begin{bmatrix}1&1&1&1&1&1&1&1\\1&-1&1&-1&1&-1&1&-1\\1&1&-1&-1&1&1&-1&-1\\1&-1&-1&1&1&-1&-1&1\\1&1&1&1&-1&-1&-1&-1\\1&-1&1&-1&-1&1&-1&1\\1&1&-1&-1&-1&-1&1&1\\1&-1&-1&1&-1&1&1&-1\end{bmatrix} & \begin{bmatrix}1\\-1\\1\\1\\1\\-1\\1\\-1\end{bmatrix} & = & \begin{bmatrix}2\\\mathbf{6}\\-2\\2\\2\\-2\\-2\\2\end{bmatrix} \\
\text{bit stream} & & \text{modified bit stream} & & \text{Hadamard matrix } H(3) & \text{modified bit stream} & & \text{spectral component magnitudes}
\end{array}$$

Figure 1. Spectral analysis of a stream of 8-bits. The essential Walsh component in this bit-stream has magnitude 6 and is represented by the second row of Hadamard matrix, $H(3)$.

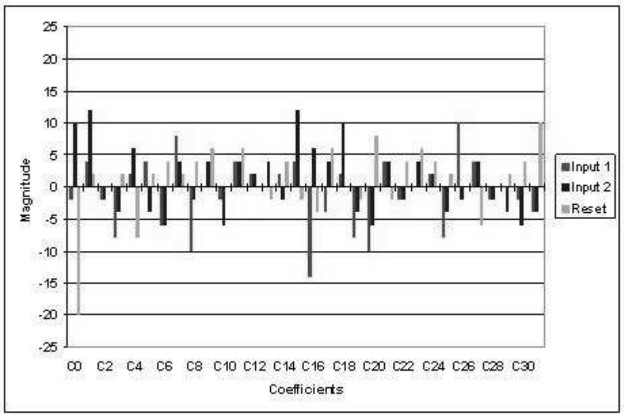

Figure 2. Walsh spectral coefficients for b01.

shows the spectral coefficients for the circuit b01. The high rising bars show high correlation with the corresponding basis bit-streams and these are considered essential. To determine the threshold, which separates the essential components from the noise components, each coefficient value considered is compared with the mean of the total spectrum. For *white noise* we shall have all equal valued coefficients and their magnitudes will be equal to the mean of any other arbitrary spectrum. Hence after squaring the coefficients, their magnitudes are compared with the mean of all coefficients. This compares the fraction of power in the coefficients to the mean power level. If this ratio is greater than some constant K, then the coefficient and hence the corresponding basis bit-stream is considered to be an essential component or else is considered non-essential or noise. The constant K affects which coefficients are being considered as essential ones. A high value of K selects only a few components as essential and a low value selects many components.

3.2 Spectral Vector Generation

After spectral analysis of the RTL vectors, the spectral coefficients are obtained. To generate test vectors

$$\text{(a) Perturbing spectra}: \begin{bmatrix}2\\\mathbf{6}\\-2\\2\\2\\-2\\-2\\2\end{bmatrix} \rightarrow \begin{bmatrix}1\\\mathbf{6}\\2\\-1\\3\\-2\\3\\-1\end{bmatrix}$$

$$\text{(b) New bit} - \text{stream obtained from perturbed spectra}:$$

$$Sign\{[1\ \ \mathbf{6}\ \ 2\ -1\ \ 3\ -2\ \ 3\ -1] \times H(3)\}$$
$$= [1\ \ 1\ \ 1\ -1\ \ 1\ -1\ \ 1\ -1] \rightarrow [1\ 1\ 1\ 0\ 1\ 0\ 1\ 0]$$

Figure 3. Bit-steam generation by perturbing the spectra. Note that the essential component having a magnitude 6 is not perturbed.

for gate-level faults, the essential spectral coefficients decided by the threshold are retained and others, being considered noise, are perturbed in a confidence range in terms of magnitude and/or in phase to generate new coefficients. The confidence levels correspond to the amount of randomness to be added.

Test vectors can easily be generated from the coefficients by multiplying the coefficients with the Hadamard matrix again. Figure 3 shows an example of reconstruction of test vectors. We generate test vector sets with different values of K. Also since we are adding noise randomly, this variation gives different characteristics to each vector set. We generate multiple sets of vectors for each value of K.

Suppose the number of spectral components obtained for a circuit is n. We generate perturbation vectors sequences, $V_1,\ V_2,\ \cdots,\ V_M$, each of length n, such that their coverage as determined by fault simulation of the gate-level circuit either reaches some target value or simply saturates. Next, we compact the test by selecting the smallest number of these sequences without reducing the coverage.

Compaction is done by an integer linear program (ILP), in a similar way as has been reported in the literature [5, 15, 19]. During fault simulation, at the beginning of each vector sequence the complete fault list is restored and the circuit is set to an unknown state. Thus, the coverage obtained for a sequence remains valid irrespective to the order in which it is applied. On fault simulation, the fault simulator provides a complete list of vector sequences that detect each fault. The vector sequence V_i is assigned an integer variable, $x_i \epsilon [0, 1]$ ($x_i = 1$: select the ith sequence or else discard it. Suppose kth fault is detected by sequences V_3, V_4, and V_{11}. The following ILP constraint picks at least one sequence that detects this fault:

$$x_3 + x_4 + x_{11} \ \geq \ 1 \qquad (3)$$

The number of such constraints equals the number of faults. The ILP then determines the values of variables x_i's that satisfy all constraints of the type 3 with the following objective function:

$$\text{Minimize} \sum_{i=1}^{i=M} x_i \qquad (4)$$

where M is the total number of vector sequences generated. For the results given in the next section, we used the ILP software contained in the AMPL mathematical programming package [7]. In the ILP solution, the smallest possible number of x's is assigned the value 1 and all others are assigned 0. The sequences with their x's set to 1 form the compacted test set.

4. Results

The spectral technique of RTL-ATPG was applied to four ITC'99 RTL benchmark circuits, four ISCAS'89 benchmark circuits and an experimental processor, PARWAN [21]. Three ITC'99 RTL benchmark circuits were synthesized in two ways, by optimizing area and by optimizing delay.

Test vectors for RTL faults were obtained using the Mentor Graphics tool FlexTest [20] which is a sequential ATPG system with a built in fault simulator. Those RTL vectors were analyzed for their spectrum, new vector sequences were generated using the technique discussed above and finally they were compacted. Results were obtained on Sun Ultra 5 machines with 256MB RAM. Table 1 shows the characteristics of the RTL test vectors generated for the circuits. Column 1 lists the circuit name. Here b01-A and b01-D are the area and delay optimized implementations of the b01 ITC'99 benchmark. ISCAS'89 benchmarks are already at the gate-level. For s5378 and s9234, we created additional versions by adding a global reset input in the

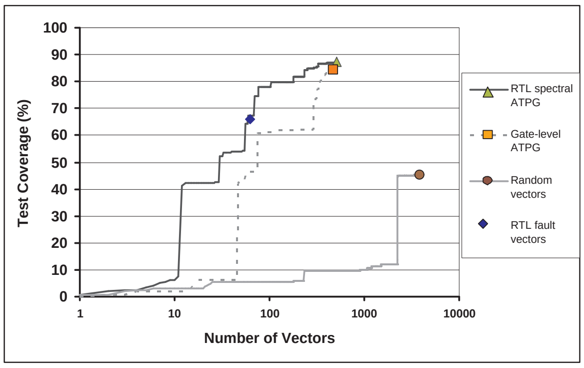

Figure 4. Test coverage of RTL ATPG (spectral vectors) for area optimized b11-A circuit.

original circuits. These are denoted with an asterisk (*) in Tables 1 and 2.

Column 3 of Table 1 lists the number of RTL faults, which are the faults at the primary inputs, primary outputs, and the inputs and outputs of flip-flops. Next in Table 1 appear the number of RTL test vectors, test generation time (CPU s) and the number of spectral components. In the absence of a true RTL ATPG program, we used FlexTest [20] to derive tests just for the selected stuck-at faults we designate as RTL faults. For the ITC'99 benchmarks this was done for two gate-level versions, one synthesized with area optimization and the other with delay optimization. The number of spectral components in the sixth column is the number of RTL vectors rounded off to the nearest power of 2.

The last two columns of Table 1 show the fault coverages of the RTL vectors. RTL Coverage is the coverage of just the RTL faults and gate-level coverage is the coverage of all stuck-at faults in the implementation. As expected, the gate-level coverage is lower than the normal requirement of being close to 100%. We will use our spectral technique to enhance this coverage. The low coverage of the RTL faults, however, may indicate a testability problem, which could limit our ability to increase the coverage either by the spectral technique or by gate-level ATPG.

Table 2 gives a comparison of the proposed RTL ATPG method with gate-level sequential ATPG. The first two columns give the circuit name and the number of gate level single stuck-at faults. The performances of RTL ATPG–spectral tests, gate-level ATPG, and random vectors can be compared by examining the data in the subsequent columns. For RTL ATPG, the number of vectors is the total number vectors in the compacted test sequences. Gate-level test coverage provided by the fault simulator of FlexTest [20] is shown in column 2 of Table 2. Note that the test coverage of FlexTest is an upward adjusted coverage, accounting for faults that are found to be untestable. The ATPG time in column 5 includes the times for RTL characterization (Table 1), perturbed spectral sequence generation, fault simulation, and ILP compaction. Of these, RTL characteri-

Table 1. Spectral characterization of circuits by RTL vectors.

Circuit		RTL Characterization					
Name	Gate-level synthesis	No. of faults	No. of vectors	CPU s	No. of spectral Components	RTL Cov. (%)	Gate-level Cov. (%)
b01-A	Area optimized	62	38	< 1	64	94.57	96.33
b01-D	Delay optimized	62	31	< 1	32	94.57	85.45
b09-A	Area optimized	248	109	519	128	75.22	78.18
b09-D	Delay optimized	248	193	418	256	75.22	72.69
b11-A	Area optimized	340	224	530	256	76.16	74.09
b11-D	Delay optimized	340	174	767	256	76.32	84.14
b14-A	A part of Viper processor	2566	110	1684	128	63.53	47.14
s1488	ISCAS'89	104	38	1	64	83.12	64.34
s5378	ISCAS'89	1602	115	1185	128	55.36	68.82
s5378*	ISCAS'89+Reset	1962	82	444	64	69.22	65.04
s9234	ISCAS'89	1840	16	706	16	18.48	16.45
s9234*	ISCAS'89+Reset	2264	59	2495	64	49.85	43.58
s35932	ISCAS'89	14536	92	50	64	68.80	94.03
PARWAN [21]	An experimental processor	434	80	156	64	70.41	69.64

zation and fault simulation are the dominant components, the other two being negligible. As we move down in Table 2, circuits become larger and we observe that RTL ATPG provides about the same test coverage and vector lengths as the gate-level ATPG, but its time increases slower. Moreover, the RTL faults used for circuit characterization and vector generation are implementation independent. Notably, the test coverage of random vectors tends to drop as circuits become larger.

Figures 4 and 5 give test coverages against the number of vectors for two circuits, b11-A and the PARWAN processor [21], for RTL spectral ATPG vectors, gate-level ATPG vectors and random vectors. The gate-level coverages of RTL vectors (generated to cover RTL faults only) are also shown by a point in each graph. For these circuits, the coverages of RTL ATPG are about 2 to 4% higher, vector lengths about double and CPU times about 30 to 50% when compared to the gate-level ATPG.

5. Conclusion

We have presented a new method of RTL test generation using spectral techniques. Test vectors generated for RTL faults are analyzed using Hadamard matrix to extract important features and new vectors are generated retaining those features. Generation of different types of test sets and performing compaction on them is found to be an efficient and reliable method for test generation. Results show that as circuits become larger the RTL method may have advantages over gate-level ATPG. This reveals a promise in generation of test vectors at RTL by spectral analysis.

RTL test generation has advantages of reduced memory and CPU time complexity. It enables the testability appraisal at RTL, and hence efforts can be made to improve testability when the design is conceptualized at higher levels of abstraction. This aspect is not explored in the present work. Further, RTL ATPG enables the testing of cores for whom only the functional information is known. An alternative method of spectral characterization of a core from functional vectors has been explored in a recent paper [28].

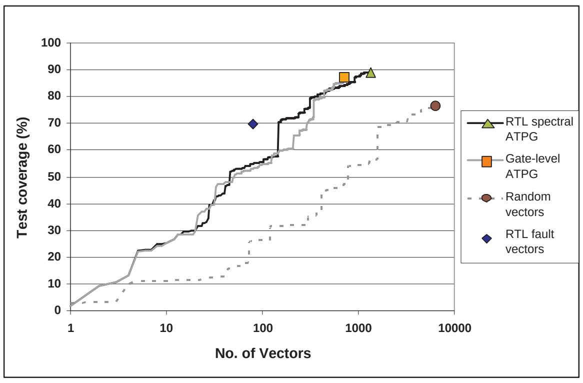

Figure 5. Test coverage of RTL ATPG (spectral vectors) for PARWAN processor [21].

References

[1] K. G. Beauchamp, *Applications of Walsh and Related Functions with an Introduction to Sequency Theory.* Orlando, FL: Academic Press, 1984.

[2] X. Chen and M. S. Hsiao, "Characteristic Faults and Spectral Information for Logic BIST," in *Proc. Int. Conf. on CAD*, Nov. 2002, pp. 294–298.

[3] X. Chen and M. S. Hsiao, "Testing Embedded Sequential Cores in Parallel Using Spectrum-Based BIST," *IEEE Trans. Computers*, vol. 55, no. 2, pp. 150–162, Feb. 2006.

[4] S. K. Devanathan and M. L. Bushnell, "Sequential Spectral ATPG Using the Wavelet Transform and Compaction," in *Proc. 19th International Conference on VLSI Design*, 2006, pp. 407–412.

Table 2. Comparison of RTL ATPG and Sequential gate-level ATPG results.

Circuit name	No. of gate-level faults	RTL ATPG – spectral tests			Gate-level ATPG			Random inputs	
		Cov. %	No. of vectors	Compaction CPU s	Cov. %	No. of vectors	ATPG CPU s	No. of vectors	Cov. %
b01-A	228	99.57	128	19	99.77	75	1	640	97.78
b01-D	290	98.77	128	19	99.77	91	1	640	95.80
b09-A	882	84.68	640	730	84.56	436	384	3840	11.71
b09-D	1048	84.21	768	815	78.82	555	575	7680	6.09
b11-A	2380	88.84	768	737	84.62	468	1866	3840	45.29
b11-D	3070	89.25	1024	987	86.16	365	3076	3840	41.42
b14-A	25894	85.09	6656	5436	68.78	500	6574	12800	74.61
s1488	4184	95.65	512	103	98.42	470	131	1600	67.47
s5378	15584	76.49	2432	2088	76.79	835	4439	3840	67.10
s5378*	15944	73.59	1399	718	73.31	332	22567	2880	62.77
s9234	28976	17.36	64	721	20.14	6967	18241	160	15.44
s9234*	29400	49.47	832	2734	48.74	12365	4119	2176	33.06
s35932	103204	95.70	256	1801	95.99	744	3192	320	50.70
PARWAN [21]	5380	89.11	1344	1006	87.11	718	3626	6400	76.63

[5] P. Drineas and Y. Makris, “Independent Test Sequence Compaction through Integer Programming,” in *Proc. Int. Conf. Computer Design*, 2003, pp. 380–386.

[6] B. J. Falkowski, “Spectral Testing of Digital Circuits,” *VLSI Design*, vol. 14, no. 1, pp. 83–105, 2002.

[7] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming.* South San Francisco, California: Scientific Press, 1993.

[8] I. Ghosh and M. Fujita, “Automatic Test Pattern Generation for Functional RTL Circuits Using Assignment Decision Diagrams,” in *Proc. 35th Design Automation Conf.*, 1999, pp. 43–48.

[9] A. Giani, S. Sheng, M. S. Hsiao, and V. D. Agrawal, “Efficient Spectral Techniques for Sequential ATPG,” in *Proc. Design, Autom. & Test in Europe (DATE) Conf.*, 2001, pp. 204–208.

[10] A. Giani, S. Sheng, M. S. Hsiao, and V. D. Agrawal, “Novel Spectral Methods for Built-In Self-Test in a System-on-a-Chip Environment,” in *Proc. 19th IEEE VLSI Test Symp.*, 2001, pp. 163–168.

[11] O. Goloubeva, G. Jervan, Z. Peng, M. Sonza Reorda, and M. Violante, “High-level and Hierarchical Test Sequence Generation,” in *Proc. of HLDVT*, 2002, pp. 169–174.

[12] T.-C. Hsiao and S. C. Seth, “An Analysis of the Use of Rademacher-Walsh Spectrum in Compact Testing,” *IEEE Trans. Comput.*, vol. 33, pp. 934–938, Oct. 1984.

[13] S. L. Hurst, D. M. Miller, and J. C. Muzio, *Spectral Techniques in Digital Logic.* Orlando, FL: Academic Press, 1985.

[14] J. Kang, S. C. Seth, and V. Gangaram, “A Functional Fault Model for Coverage Analysis and Test Generation,” 2006. Manuscript in preparation.

[15] K. R. Kantipudi and V. D. Agrawal, “On the Size and Generation of Minimal N-Detection Tests,” in *Proc. 19th Int. Conf. VLSI Design*, 2006, pp. 425–430.

[16] G. Kasturirangan and M. S. Hsiao, “Spectrum-Based BIST in Complex SOCs,” in *Proc. 20th IEEE VLSI Test Symp.*, Apr. 2002, pp. 111–116.

[17] O. I. Khan and M. L. Bushnell, “Aliasing Analysis of Spectral Statistical Response Compaction Techniques,” in *Proc. 19th International Conference on VLSI Design*, 2006, pp. 801–806.

[18] H. Kim and J. P. Hayes, “High-Coverage ATPG for Datapath Circuits with Unimplemented Blocks,” in *Proc. International Test Conf.*, 1998, pp. 577–586.

[19] J. P. Marques-Silva, “Integer Programming Models for Optimization Problems in Test Generation,” in *Proc. IEEE Asia-South Pacific Design Automation Conf.*, 1998, pp. 481–487.

[20] Mentor Graphics, *FastScan and FlexTest Reference Manual*, 2004.

[21] Z. Navabi, *Analysis and Modeling of Digital Systems.* New York: McGraw-Hill, 1993.

[22] S. Ravi and N. K. Jha, “Fast Test Generation for Circuits with RTL and Gate-Level Views,” in *Proc. International Test Conf.*, 2001, pp. 1068–1077.

[23] A. K. Susskind, “Testing by verifying Walsh coefficients,” *IEEE Trans. Computers*, vol. 32, no. 2, pp. 198–201, Feb. 1983.

[24] P. A. Thaker, V. D. Agrawal, and M. E. Zaghloul, “A Test Evaluation Technique for VLSI Circuits Using Register-Transfer Level Fault Modeling,” *IEEE Trans. CAD*, vol. 22, no. 8, pp. 1104–1113, Aug. 2003.

[25] M. A. Thornton, R. Drechsler, and D. M. Miller, *Spectral Techniques in VLSI CAD.* Boston: Springer, 2001.

[26] E. W. Weisstein. From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com.

[27] N. Yogi and V. D. Agrawal, “High-Level Test Generation for Gate-Level Fault Coverage,” in *15th IEEE North Atlantic Test Workshop*, May 2006.

[28] N. Yogi and V. D. Agrawal, “Spectral Characterization of Functional Vectors for Gate-Level Fault Coverage Tests,” in *Proc. 10th VLSI Design and Test Symposium*, Aug. 2006, pp. 407–417.

[29] J. Zhang, M. L. Bushnell, and V. D. Agrawal, “On Random Pattern Generation with the Selfish Gene Algorithm for Testing Digital Sequential Circuits,” in *Proc. International Test Conf.*, 2004, pp. 617–626.