Virtual Scan Chains: A Means for Reducing Scan Length in Cores

[1]Abhijit Jas, [2]Bahram Pouya, and [1]Nur A. Touba

[1]Computer Engineering Research Center
University of Texas at Austin
Austin, TX 78712-1084

[2]Test Technology Group
Advanced Products Research & Development
Motorola, Austin, TX 78721

Abstract

A novel design-for-test (DFT) technique is presented for designing a core with a "virtual scan chain" which looks (to the system integrator) like it is shorter than the real scan chain inside the core. The I/O pins of a core with a virtual scan chain are identical to the I/O pins of a core with a normal scan chain. For the system integrator, testing a core with a virtual scan chain is identical to testing a core with a normal scan chain. The only difference is that the virtual scan chain is much shorter so the size of the scan vectors and output response is smaller resulting in less test data and fewer scan shift cycles. The process of mapping the virtual scan vectors to real scan vectors is handled inside the core and is completely transparent to the system integrator. It is done by using LFSRs to "expand" the shorter virtual test vector into a full test vector. Results indicate that virtual scan chains can be designed which are several times shorter than the real scan chains inside the core.

1. Introduction

Core-based design is emerging as a new paradigm for the design of integrated circuits. System integrators construct a system-on-a-chip using pre-designed and pre-verified cores as building blocks. System integrators can purchase cores from various core vendors. This creates a competitive environment where multiple core vendors are trying to sell cores with similar functionality. As the complexity of systems-on-a-chip continues to increase, the difficulty and cost of testing such chips is escalating rapidly [Zorian 98]. One characteristic of a core that emerges as an important distinguishing factor is test complexity. Given two cores with similar functionality, the core that can be thoroughly tested with the smallest amount of test data and the simplest tester program has a significant competitive advantage because it reduces manufacturing test costs.

In this paper, a novel design-for-test (DFT) technique that allows core vendors to reduce the test complexity of the core they are trying to market is presented. The idea is to create a core with a "virtual scan chain" which looks (to the system integrator) like it is shorter than the real scan chain inside the core (as illustrated in Fig. 1). The I/O pins of the core with the virtual scan chain are identical to the I/O pins of a core with a real scan chain. There is a "scan data in" pin (SDI), "scan data out" pin (SDO), and a "scan enable" (SE) pin to control the scan chain. Without loss of generality, this paper will describe a single virtual scan chain replacing a single real scan chain, but obviously if the core has multiple real scan chains then they could be replaced with multiple virtual scan chains. For the system integrator, testing a core with a virtual scan chain is identical to testing a core with a normal scan chain. The only difference is that the virtual scan chain is shorter so the size of the scan vectors and output response is smaller resulting in less test data as well as less test time (fewer scan shift cycles). The tester program that is required for testing a core with a virtual scan chain is no different than that required for testing a core with a normal scan chain. Thus, from the system integrator's point of view, a core with a virtual scan chain is identical to a core with a normal scan chain in all respects expect that it is much shorter. The real scan chain inside the core, however, is longer than the virtual scan chain. The process of mapping the virtual scan vectors to real scan vectors is handled inside the core and is completely transparent to the system integrator. One nice feature of a virtual scan chain is that it hides Intellectual Property (IP) because it encodes the core's scan vectors and disguises the real number of scan cells.

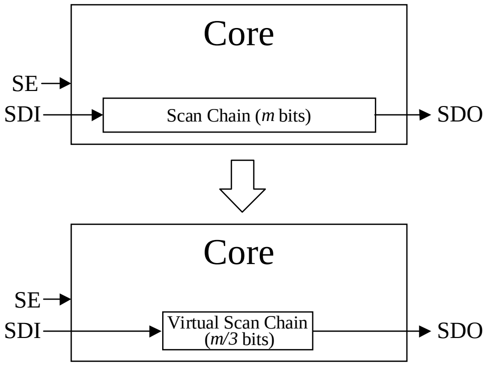

Figure 1. Concept of Virtual Scan Chain

0-7695-0613-5/00 $10.00 © 2000 I

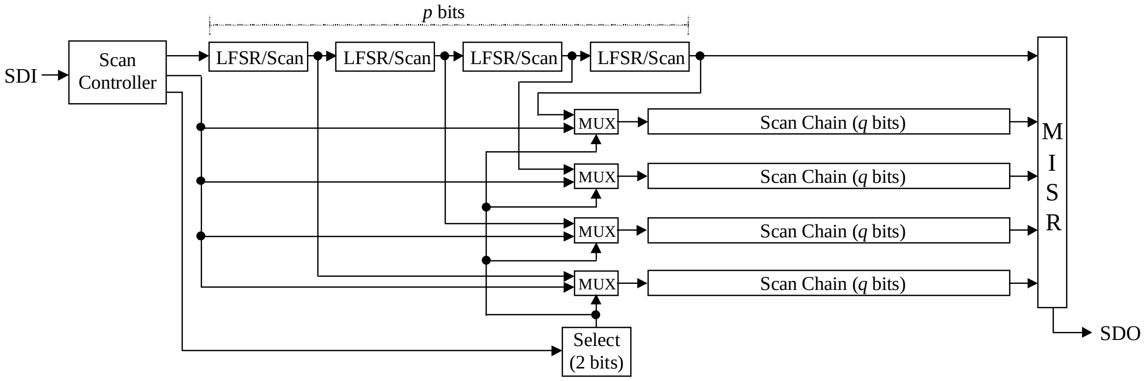

Figure 2. Virtual Scan Chain that is $p+q+2$ Bits Long

The problem of reducing the test time for cores has been attacked from several different angles in recent literature [Aerts 98], [Jas 98], [Hamzaoglu 99], [Rajski 98], [Sugihara 98]. Scan chain architectures for core-based designs that maximize bandwidth utilization are presented in [Aerts 98]. A technique for compression/decompression of scan vectors using cyclical decompressors and run-length coding is described in [Jas 98]. Both of these techniques apply to cores with scan chains, and hence also apply to cores with virtual scan chains. Since a core with a virtual scan chain is fully compatible to a core with a normal scan chain from the system integrator's point of view, techniques for optimal testing of cores with normal scan can be applied to cores with virtual scan.

Recently, Hamzaoglu and Patel [Hamzaoglu 99] presented an approach called "Parallel Serial Full Scan (PSFS)" for reducing test time in cores. The idea is to have two modes for loading a scan chain from a single scan data in (SDI) pin: parallel and serial. In parallel mode, the same test vector is shifted into multiple scan chains. This allows some reduction in the total amount of test data.

Another approach for reducing test time is to use built-in self-test (BIST). A modular BIST approach that allows sharing of BIST control logic among multiple cores is presented in [Rajski 98]. A novel technique for combining BIST and external testing across multiple cores is described in [Sugihara 98].

Designing a core with BIST is an alternative to designing a core with a virtual scan chain. However, there are several advantages to developing a core with a virtual scan chain compared with BIST:

- It is non-trivial to achieve high fault coverage with BIST. Inserting test points to improve fault coverage degrades performance. In many cases, it may be undesirable to modify the function logic.
- Pseudo-random BIST vectors can cause problems with illegal states and bus conflicts.
- BIST requires long test lengths which can add to tester socket time (i.e., the time that the chip sits in the tester socket).
- Developing a tester program for handling a core with BIST may be more complicated for the system integrator. It may not be compatible with the system integrator's standard test methodology.
- A core with a virtual scan chain is compatible with all other cores with scan chains, hence the same test integration methodologies and tools can be used.

For these reasons, the system integrator may prefer a core with a virtual scan chain to one with BIST.

2. Implementing a Virtual Scan Chain

Having described the concept of a virtual scan chain, now the details of how it can be implemented inside the core will be discussed. It is best explained with an example. Figure 2 shows an m-bit long real scan chain which is implemented as a $(p+q+2)$-bit long virtual scan chain. The real scan chain (consisting of m scanned flip-flops in the core) is divided into 5 smaller scan sub-chains. One scan sub-chain is p bits long and the other 4 are q bits long $(m=p+4q)$. Only $p+q+2$ bits will be shifted in from the SDI pin (since that is the size of each virtual scan vector), and after that the system clock will be applied to capture the response back into the scan chain. During those $p+q+2$ scan cycles, all $m=p+4q$ scan elements must be filled with the real scan vector. The way this is done will now be described.

Select	LFSR Seeds	Vector for Selected Sub-Scan Chain
2-bits	p-bits	q-bits

Figure 3. Format of Virtual Scan Vector

There is a scan controller which is a simple finite state machine that controls which of the scan sub-chains the SDI input is being shifted into during each scan cycle. During the first two scan cycles, the SDI input is shifted into a 2-bit select (SEL) register. During the next *p* scan cycles, the SDI input is shifted into the *p*-bit scan sub-chain. During the last *q* scan cycles, the SDI input is shifted into one of the 4 *q*-bit scan sub-chains (selected by the 2-bits shifted into the SEL register). The format of the virtual scan vector is shown in Fig. 3. The remaining 3 *q*-bit scan sub-chains are loaded in the following way. After the scan controller loads the *p*-bit scan sub-chain, it configures the *p*-bit scan sub-chain as 4 separate LFSRs (each of which is serially connected to one of the 4 *q*-bit scan sub-chains). These LFSRs are then run in autonomous mode during the final *q* scan cycles. Thus, during the final *q* scan cycles, 3 of the *q*-bit scan sub-chains are concurrently loaded from their corresponding autonomous LFSR while the remaining *q*-bit scan sub-chain is directly loaded from the SDI input. Thus, at the end of $p+q+2$ scan cycles, all $m=p+4q$ scan elements are loaded with a test vector. What happens is that the seeds that are loaded into the LFSRs are “expanded” by running the LFSRs in autonomous mode. The process of finding a virtual scan vector that will map to a desired test vector is described in the next section.

As the next virtual scan vector is shifted in, the response of the previous vector gets shifted out. The response of the multiple sub-chains needs to be compacted as it is shifted out since there is only one SDO output. This can be done using a multiple-input signature register (MISR) with the feedback line connected to the SDO output. This will make the output response of the virtual scan chain look like that of a normal scan chain. For the last test vector in the test set, the last few bits of the output response information may get stuck in the flip-flops of the MISR and not get shifted out. This problem can be solved by adding an extra dummy test vector to the end of the virtual scan vector test set to effectively “flush” the contents of the MISR out. Using a MISR introduces the possibility of losing fault coverage due to aliasing. This can be avoided by either doing fault simulation with the MISR when generating the test vectors, or by choosing the size of the MISR so that the probability of aliasing is sufficiently low.

Note that while the example in Fig. 2 shows 4 *q*-bit scan sub-chains, any number of such sub-chains can be used (e.g., 8, 16, 32, etc.). Selecting the number of sub-chains and the size of the LFSRs will be discussed in Sec. 4 after the test generation procedure is described.

3. Constructing Virtual Scan Vector Test Set

In this section, a procedure for finding a minimal set of virtual scan vectors that provides the desired fault coverage is described. It is assumed that the virtual scan chain architecture (i.e., the number of scan sub-chains and sizes of the LFSRs) has already been selected. The process of choosing a virtual scan chain architecture for a particular core will be discussed in the next section as many of the issues for that relate to the test generation procedure described in this section.

Each virtual scan vector gets mapped to a test vector in the real scan chain by using LFSRs. The idea of expanding an LFSR seed into a scan vector was first proposed in [Koenemann 91]. If the LFSR has *k*-stages, then a system of linear equations can be solved to find a seed that will generate a particular scan vector. Because of linear dependencies in the LFSR, it is not always possible to find a solution for any arbitrary scan vector (this problem will be addressed shortly).

3.1 Test Generation

The procedure for finding a virtual scan vector test set that provides a desired fault coverage is as follows. First, random test generation is used to find virtual scan vectors that detect the easy-to-detect faults. This is done by simply simulating random virtual scan vectors, and those that detect previously undetected faults are added to the test set. For the hard-to-detect faults, normal ATPG is done to find test cubes (i.e., the unspecified inputs are left as X’s). The linear equations for the LFSRs are then solved to find a virtual scan vector that will map to the test cube. For some of the test cubes with a large number of specified inputs, it may not be possible to solve the linear equations for all of the LFSRs (due to linear dependencies in the LFSR). This means that there is no seed for the LFSR that will generate the needed scan vector. If only one LFSR is unsolvable, then that is okay because the corresponding *q*-bit scan sub-chain can be selected as the one to be directly loaded from the SDI input. However, if multiple LFSRs are unsolvable, then the size of the LFSRs can be increased until a solution is found. When the LFSR sizes are changed, then the mapping of virtual scan vectors to test vectors is changed, so a second pass is required (i.e., new seeds for the altered LFSR need to be computed for the test cubes). Thus, a good strategy is to first try to find a seed for the test cube with the largest number of specified bits. If the LFSRs need to be made bigger to generate the largest test cube, then that can be done right away. An alternate approach to making the LFSRs bigger is to use multiple-polynomial LFSRs [Hellebrand 95] or mapping logic [Touba 96], [Wunderlich 96]. This allows the size of the LFSRs to remain small. The drawback is that it adds some additional overhead.

3.2 Static Compaction

For a virtual scan chain, the amount of static compaction (merging of compatible test cubes) that can be done is limited because static compaction specifies additional X's which may cause a test vector to no longer be mappable to a virtual scan vector. As a result, the number of virtual scan vectors required for a particular fault coverage may be more than the number of normal scan vectors. However, because each virtual scan vector is much shorter (has fewer number of bits), the overall amount of test data is still greatly reduced (as will be shown in the experimental results).

The static compaction procedure for a virtual scan chain must check when two test cubes are combined that the additional specified bits do not cause the system of linear equations for more than one LFSR to become unsolvable. Some threshold on the total number of specified bits in each scan sub-chain can be used as a heuristic on whether the linear equations will be solvable. Static compaction of test cubes can proceed until the number of specified bits for more than one scan sub-chain exceeds the threshold.

After static compaction, the linear equations for each test cube can be solved to find a virtual scan vector that maps to the test cube. The virtual scan vector for each test cube can then be expanded (by simulating the LFSRs) to the corresponding fully specified test vector which can then be fault simulated to possibly drop additional undetected faults.

4. Selecting Virtual Scan Chain Architecture

The two important parameters in designing a virtual scan chain are how many scan sub-chains there will be and how big will the LFSRs for each sub-chain be. As was discussed in the previous section, the LFSRs must be large enough to allow the system of linear equations to be solved for all the test cubes. Moreover, the size of the LFSRs also affects how much static compaction can be done. Larger LFSRs can handle more specified bits in the test cubes which means more static compaction can be done resulting in fewer virtual scan vectors. However, larger LFSRs also increase the size of each virtual scan vector. So there is a tradeoff on the number of virtual scan vectors versus the size of each virtual scan vector. The product of the two determines the total amount of test data and total number of scan cycles required for testing the core.

The tradeoff is illustrated in the graph in Fig. 4. The horizontal axis is the ratio of the scan elements configured as LFSRs to the total number of scan elements. As the graph goes from left to right, the LFSR sizes grow larger. The virtual scan length grows as the LFSR sizes grow. However, the constraints on static compaction go down as the LFSR sizes grow, so the number of virtual scan vectors needed to achieve a particular fault coverage goes down because more compaction can be done. As can be seen in the graph, initially on the far left, increasing the LFSR sizes really helps a lot to reduce the number of virtual scan vectors, however, a point is reached where increasing the LFSR size does not have much effect on test vector compaction. Hence, the product of the virtual scan length and the number of vectors is minimized somewhere near that point. Note that in Fig. 4, the total test data curve was scaled down to fit in the graph with the other values (scaling leaves the shape of the graph unchanged).

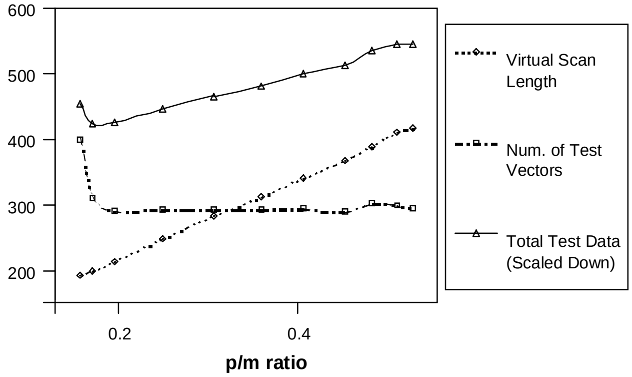

Figure 4. Variation of Parameters with *p*/*m* Ratio for *s13207* with 8 scan sub-chains

In our experiments, we found that having the LFSRs be around 25% of the size of each scan sub-chain was generally a good value for minimizing the amount of test data. If the size of the LFSRs are kept at 25%, then having more sub-chains will tend to reduce the amount of static compaction that is possible. It is better to have fewer sub-chains to increase static compaction for two reasons: one is that it gives more flexibility for handling specified bits (because the sub-chains are larger), and the other is that one sub-chain is fed directly from the SDI pin so it is better to have that be as large as possible. So again there is a tradeoff. Increasing *n* reduces the size of each virtual scan vector, but it also increases the number of virtual scan vectors because it constrains static compaction. The reduction in the size of the virtual scan chain tapers off rapidly as *n* becomes greater than 16. In our experiments, we did not see any cases where having more than 16 sub-chains gave better results.

Figure 5 show an example of how the total test data varies with the virtual scan length. As can be seen, when the virtual scan length gets very small, the constraints on static compaction imposed by the LFSRs cause the number of test vectors to become very large so that the test data increases.

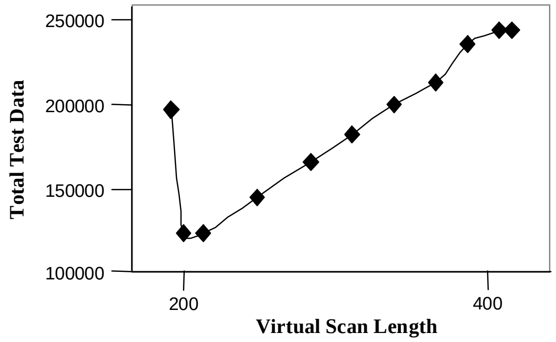

Figure 5. Total Test Data vs. Virtual Scan Length for *s13207* with 8 scan sub-chains

5. Experimental Results

Experiments were performed for the largest ISCAS 89 benchmark circuits. Table 1 shows the results comparing a virtual scan chain with a normal scan chain. The fault coverage in both cases is 100% of detectable faults. A virtual scan chain with 4, 8 and 16 sub-chains was designed for each circuit. For the normal scan chain, the following are shown: the size of the normal scan chain, the number of test vectors with full (unconstrained) static compaction, and the total amount of test data (that must be stored on the tester). For the virtual scan chain, the following are shown: the number of scan sub-chains *n*, the size of the virtual scan chain (which is much shorter than the real scan chain), the number of test vectors which results from static compaction under the constraints imposed by the linear dependencies of the LFSRs, and the total amount of test data. Lastly, the percentage reduction in the number of scan cycles required for testing each circuit is shown. The percentage is computed as follows:

[(Normal Scan Test Data) - (Virtual Scan Test Data)] /
(Normal Scan Test Data) x 100

As can be seen from the results, the number of test vectors is larger for the virtual scan chain because of the constraints on static compaction, however, the number of bits in each vector is much less than that of a normal scan chain. Consequently, the total amount of test data is reduced and the number of scan cycles for testing the circuit is also reduced.

Table 1. Results Comparing Virtual Scan Chain with Normal Scan Chain - Using Same ATPG Software

Circuit Name	Normal Scan Chain			Virtual Scan Chain				% Reduction of Scan Cycles
	Scan Size (bits)	Num. of Test Vectors	Test Data (bits)	*n*	Scan Size (bits)	Num. Of Test Vectors	Test Data (bits)	
s9234	247	201	99294	4	126	215	54180	45.4
				8	124	213	52824	**46.8**
				16	131	231	60522	39.0
s13207	700	293	410200	4	264	300	158400	61.4
				8	199	306	121788	**70.3**
				16	194	317	122996	70.0
s15850	611	173	211406	4	235	192	90240	57.3
				8	208	185	76960	63.6
				16	210	181	76020	**64.0**
s38417	1664	271	901888	4	652	301	392504	56.5
				8	547	282	308508	**65.8**
				16	573	307	351822	60.1
s38584	1464	220	644160	4	605	246	297660	53.8
				8	452	253	228712	64.5
				16	343	295	202370	**68.6**

Table 2. Results Comparing Virtual Scan Chain with Normal Scan Chain - Using Compactest Test Vectors from [Pomeranz 93] in Normal Scan Chain

Circuit Name	Normal Scan Chain			Virtual Scan Chain				% Reduction of Scan Cycles
	Scan Size (bits)	Num. of Test Vectors	Test Data (bits)	*n*	Scan Size (bits)	Num. Of Test Vectors	Test Data (bits)	
s9234	247	135	66690	8	124	213	52824	20.8
s13207	700	237	331800	8	199	306	121788	63.3
s15850	611	123	150306	16	210	181	76020	49.4
s38417	1664	95	316160	8	547	282	308508	2.4
s38584	1464	125	366000	16	343	295	202370	44.7

In Table 1, the same ATPG and static compaction procedures were used for generating the test vectors for both scan chains. The only difference was the constraints on static compaction for the virtual scan chain which is why the number of test vectors is higher for the virtual scan chain. One way to reduce the test data for the normal scan chain would be to use a more powerful dynamic compaction procedure such as Compactest [Pomeranz 93]. For comparison, results are shown in Table 2 where the test vectors for the normal scan chain were generated using Compactest whereas the vectors for the virtual scan chain were generated using static compaction as before. The percentage reduction in the number of scan cycles for the virtual scan chain is of course less in this case because the reference point is now the smaller set of test vectors generated by Compactest (although the reduction is still very significant in most cases). Note, however, that this is not a completely fair comparison because a more powerful ATPG tool is being used for generating test vectors for the normal scan chain than what is used for the virtual scan chain. One way to reduce this discrepancy would be to modify Compactest to generate compacted test vectors under the constraints imposed by the linear dependencies in the LFSRs for the virtual scan chain. That would bring down the number of virtual scan vectors and hence further reduce the test data.

6. Conclusion

The key feature of designing a core with a virtual scan chain is that it allows full compatibility with normal scan chains (test I/O pins and tester program are identical). The test data and test time is reduced without adding any additional complications for the system integrator. The system integrator can use all of the standard integration methodologies as would be used for a core with a normal scan chain.

A core with a virtual scan chain will reduce test costs for the system integrator. Thus, core vendors may find virtual scan chains a means to achieve a competitive advantage in selling their cores.

In this work, the default ordering of the scan chains was used. It was assumed that the core designer would want to choose the ordering of the scan chain based on other criteria such as minimizing routing. However, one way to improve the results would be to specially order the scan chain. The scan cells could be partitioned into scan sub-chains in a way that equally distributes the specified bits in the test cubes in order to minimize the size of the LFSRs and/or maximize the amount of static compaction that can be performed.

Acknowledgements

This material is based on work supported in part by the National Science Foundation under Grant No. MIP-9702236, and in part by the Texas Advanced Research Program under Grant No. 003658-0644-1999.

References

[Aerts 98] Aerts, J., and E.J. Mariniseen, “Scan Chain Design for Test Time Reduction in Core-Based ICs,” *Proc. of International Test Conference*, pp. 448-457, 1998.

[Hamzaoglu 99] Hamzaoglu, I., and J. Patel, ”Reducing Test Application Time for Full Scan Embedded Cores,” *International Symposium on Fault Tolerant Computing*, pp. 260-267, 1999.

[Hellebrand 95] Hellebrand, S., J. Rajski, S. Tarnick, S. Venkataraman and B. Courtois, ”Built-In Test for Circuits with Scan Based on Reseeding of Multiple-Polynomial Linear Feedback Shift Registers,” *IEEE Transactions on Computers*, Vol. 44, No. 2, pp. 223-233, Feb. 1995.

[Jas 98] Jas, A., and N.A. Touba, “Test Vector Compression via Cyclical Scan Chains and Its Application to Testing Core-Based Designs ,” *Proc. of International Test Conference*, pp. 458-464, 1998.

[Koenemann 91] Koenemann, B., “LFSR-Coded Test Patterns for Scan Designs,” *Proc. of European Test Conference*, pp. 237-242, 1991.

[Pomeranz 93] Pomeranz, I., L.N. Reddy, and S.M. Reddy, “COMPACTEST: A Method to Generate Compact Test Sets for Combinational Circuits,” *IEEE Transactions on Computer-Aided Design*, Vol. 12, No. 7, pp. 1040-1049, Jul. 1993.

[Rajski 98] Rajski, J., and J. Tyszer, “Modular Logic Built-In Self-Test for IP Cores,” *Proc. of International Test Conference*, pp. 313-321, 1998.

[Sugihara 98] Sugihara, M., H. Date, and H. Yasuura, “A Novel Test Methodology for Core-Based System LSIs and a Testing Time Minimization Problem,” *Proc. of International Test Conference*, pp. 465-472, 1998.

[Touba 96] Touba, N.A., and E.J. McCluskey, “Altering a Pseudo-Random Bit Sequence for Scan-Based BIST,” *Proc. of International Test Conference*, pp. 167-175, 1996.

[Wunderlich 96] Wunderlich, H.-J., and G. Kiefer, “Bit-Flipping BIST”, *Prof. of International Conference on Computer-Aided Design (ICCAD)*, pp. 337-343, 1996.

[Zorian 98] Zorian, Y., E.J. Marinissen, and S. Dey, “Testing Embedded-Core Based System Chips,” *Proc. of International Test Conference*, pp. 130-143, 1998.