Efficient Seed Utilization for Reseeding based Compression*

Erik H. Volkerink[1,2] and Subhasish Mitra[3]

[1]Center for Reliable Computing (CRC)
Stanford University, Stanford, CA

[2]Agilent Laboratories
Palo Alto, CA

[3]Intel Corporation
Sacramento, CA

Abstract

The conventional LFSR reseeding technique for test data compression generates one test pattern from each LFSR seed. The seed size is determined by the maximum number of specified bits in a test pattern belonging to a given test set. However, for most practical designs the majority of test patterns have significantly fewer specified bits compared to the maximum. This limits the amount of compression that can be achieved with conventional reseeding. This paper presents a new reseeding technique that overcomes this problem by generating a single test pattern from multiple seeds and multiple test patterns from a single seed. The new reseeding technique is applied to two industrial designs resulting in significant reduction in tester memory requirement and test application time compared to the conventional reseeding technique.

1. Introduction

As the design complexity increases so does the test data volume [ITRS 99]. This increase in test data volume results in three major test problems: (1) Increased tester pattern memory requirements, (2) Long upload times, and (3) Limited I/O bandwidth. Recently, several compression techniques to reduce test data volume have been published. Most techniques exploit the fact that most bits in test patterns generated by ATPG tools are don't cares – only a very small proportion of all bits in a test pattern are specified to be 0 or 1 (~ 1%-2% [Barnhart 01]).

It has been shown in several publications [Koenemann 91, Koenemann 01] that test patterns can be efficiently compressed by reseeding a linear feedback shift register (LFSR). In conventional reseeding, the LFSR is loaded with a binary combination of 1s and 0s, called a *seed*, corresponding to every test pattern from the tester. The LFSR is then run in an autonomous mode to fill the scan chains. The seed corresponding to a test pattern is computed such that the bit sequence produced by the LFSR starting from that seed state correctly generates all specified bits in that test pattern. The size of the seed, which is the same as the number of flip-flops in the LFSR, is generally determined by the number of specified bits in the *worst test pattern*, the test pattern with the maximum number of specified bits. However, as shown in Fig. 1, the majority of patterns have significantly fewer number of specified bits compared to the number of specified bits in the worst test pattern. Thus, although a significantly shorter seed would have sufficed for most test patterns, the number of seed bits that are stored on the tester for every test pattern is equal to the number of seed bits required for the worst test pattern. This significantly limits the amount of compression that can be achieved with conventional reseeding.

This paper overcomes the above problem through a very simple yet effective modification of the conventional reseeding architecture so that multiple test patterns can be generated from a single seed or a single test pattern can be generated from multiple seeds. Application of this technique to industrial designs shows significant reduction in tester memory requirement and test time compared to conventional reseeding.

Several previous publications have addressed the above problem [Zacharia 95, Krishna 01, Krishna 02, Koenemann 01, Rajski 02]. Unlike the schemes described in [Zacharia 95, Krishna 01], our technique achieves similar end result through a very simple modification to the conventional reseeding architecture. In principle, our approach has similarities with the technique described in [Krishna 02]. Our technique doesn't use any additional seed compression and is capable of reducing both hardware overhead and test data volume. More detailed differences between these two approaches are discussed later in this paper.

This paper is organized as follows. A brief overview of conventional LFSR reseeding is presented in Sec. 2. Section 3 describes the new LFSR reseeding technique and discusses various scenarios to demonstrate the effectiveness of this technique.

* This work was done at CRC, Stanford University

Proceedings of the 21st IEEE VLSI Test Symposium (VTS'03)
1093-0167/03 $17.00 © 2003 IEEE

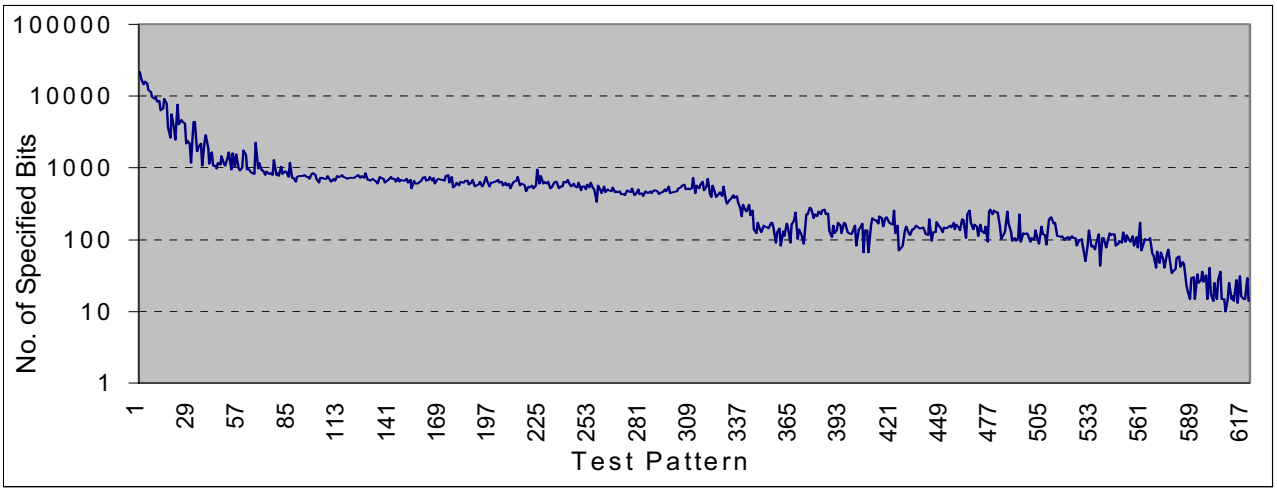

Figure 1: The number of specified bits in test patterns of an industrial design (ASIC 1)

Results obtained from the application of this technique to industrial designs are presented in Sec. 4. We conclude in Sec. 5.

2. Conventional LFSR Reseeding

In the reseeding architecture shown in Fig. 2, for every test pattern a seed is stored on the tester and loaded onto the LFSR. The LFSR is then run in autonomous mode for a number of cycles equal to the length of the longest scan chain. For details about LFSR and phase-shifter, please refer to [Bardell 87]. Suppose that the number of specified bits in the worst pattern is s. The number of LFSR flip-flops (also equal to the seed size) is generally set to be equal to $s + 20$. Since the LFSR and the phase-shifter are linear circuits, for every test pattern a set of linear equations corresponding to the specified bits in that test pattern can be generated where the variables correspond to the bits in the initial seed to be loaded onto the LFSR to generate that pattern. A solution to these linear equations generates the seed for that test pattern. It is proved in [Koenemann 91] that for a seed size of $s + 20$, the probability that the linear equations cannot be solved to generate a seed is less than or equal to 10^{-6}.

Generally, when the LFSR is run in the autonomous mode to fill the scan chains with a test pattern, the seed corresponding to the next test pattern is shifted into the on-chip seed buffer shown in Fig. 2. This is done in such a way that the next seed is ready by the time the LFSR finishes filling the scan chains with the current test pattern. Thus, if the LFSR size is 1,000 bits and the length of the longest scan chain is 500 bits, then at least 2 inputs are required to scan in a seed into the seed buffer. Note that, if 5 inputs are used then it will take 200 cycles to scan in a seed into the seed buffer.

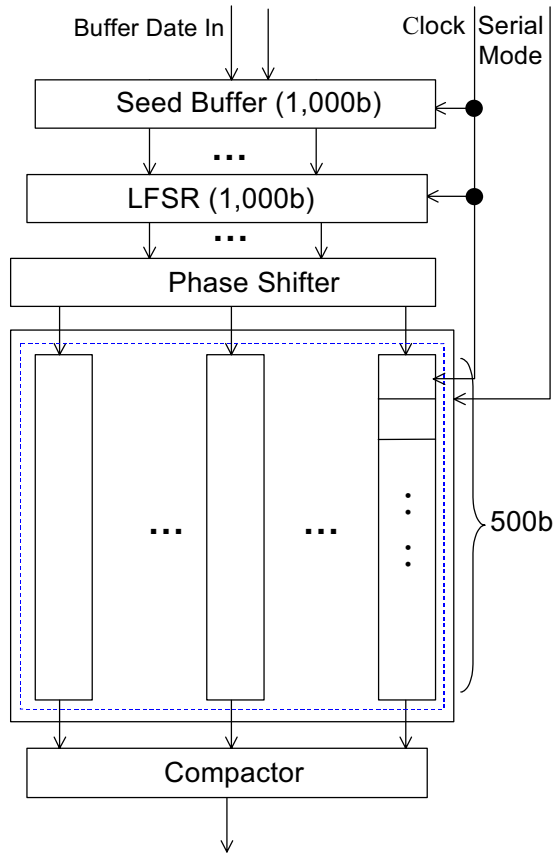

Figure 2: Conventional Reseeding Architecture

Since it takes 500 cycles to scan in a test pattern into the scan chains, the contents of the tester pattern memory associated with the 5 inputs are essentially wasted for 300 (= 500-200) cycles, unless a tester with repeat count compression capabilities is used [Barnhart 01, Volkerink 02]. This increases the effective tester memory requirement.

It is easy to see from Fig. 1 that it may not be efficient to choose the seed size based on the number of specified bits in the worst pattern. Instead, the seed size may be chosen based on the number of specified bits in a "typical" test pattern, e.g., approximately 1,000 instead of 10,000 in the worst test pattern. In that case, a serial shift mode is required. In this mode the LFSR and the phase shifter are bypassed, and test patterns that cannot be generated by the LFSR from a seed state (e.g., a test pattern with the number of specified bits greater than the seed size-20) are directly scanned in by reconfiguring all scan chains into fewer chains. With serial mode, the optimum seed size is a tradeoff between (1) the advantage of compressing test patterns into shorter seeds and (2) the disadvantage of shifting in a set of uncompressed patterns using the serial shift mode.

Reseeding with serial mode, also limits the amount of compression that can be achieved. For example, referring to Fig. 1, if the seed size is 1,020 bits, test patterns with 100 specified bits will be generated from a seed 1,020 bits long although theoretically 120 bits should be enough to generate this test pattern. This wastage is quantified using the efficiency metric discussed in more details in Sec. 4. Also, when a pattern with 10,000 specified bits is scanned in using the serial mode, the entire test pattern has to be stored in the tester instead of a seed 10,000 bits long. Our new reseeding technique, described in Sec. 3, overcomes these problems using a very simple modification to the reseeding architecture of Fig. 2.

3. New Reseeding Architecture

The new reseeding architecture, one of the major contributions of this paper, is shown in Fig. 3. In addition to the architecture in Fig. 2, we have another input from the tester, the *stall* signal. When the stall signal is 1, the seed buffer is only clocked by the input clock and the LFSRs and the scan chains aren't clocked. When the stall signal is 0 the seed buffer, the LFSR and the scan chains are all clocked by the input clock. Depending on the tester architecture and scan design, the same capability may be achieved by having two clocks from the tester – one driving the clock inputs of the seed buffer and the other driving the clock inputs of the LFSR and the scan chains.

The stall signal is the key to the efficiency of the new reseeding architecture in terms of achieving significant data compression and test time reduction. A *scan cycle* is defined to be the application of a scan clock to the scan chains. A *bit slice* is defined to be the set of inputs applied to the scan chain inputs at a scan cycle. The number of entries in a bit slice is equal to the number of scan chains. It must be obvious to the reader that an entire set of test patterns can be represented by a sequence of several bit slices. Referring to Fig. 3, a single test pattern can be imagined to be a sequence of 500 bit slices. The usefulness of this concept is illustrated using the following examples. Note that, since the LFSR is now only 500 bits long, the area overhead of this architecture is also smaller than that of Fig. 2.

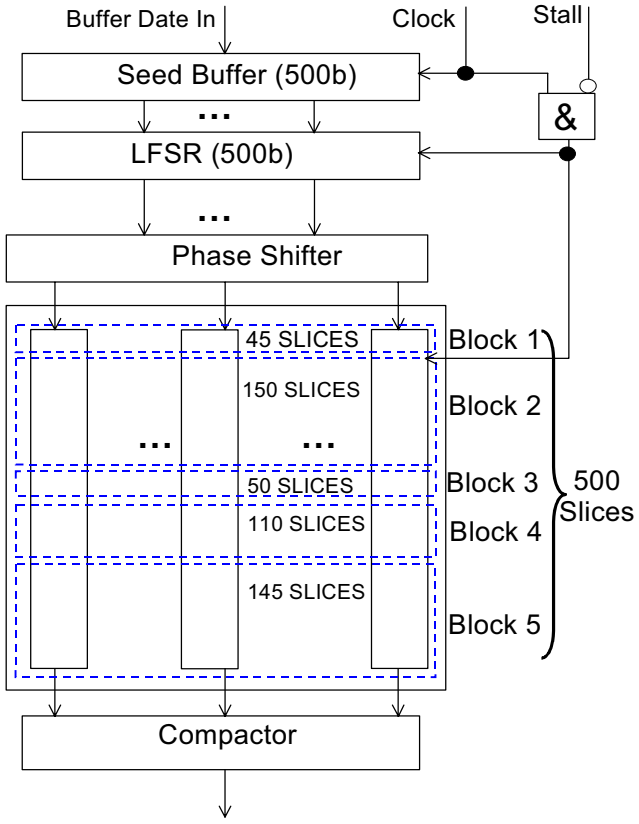

Figure 3: Architecture for Input Channel Reduction

Example 1: A test pattern with 2,000 specified bits

In the conventional architecture, we need the serial mode to scan in this test pattern. In the new architecture, the test pattern is divided into blocks of consecutive bit slices such that the specified bits in each block can be generated by a seed 500 bits long. As shown in Fig. 3, Block 1 contains 145 bit slices, Block 2 contains 150 bit slices, Block 3 contains 50 bit slices,

Block 4 contains 110 bit slices and Block 5 contains 145 bit slices. Next, 5 seeds corresponding to these 5 blocks are generated such that the LFSR loaded with a seed generates the specified bits in the corresponding block. For example, the LFSR is loaded with Seed 5 first. After 145 scan cycles with stall = 0 the scan chains contain the specified bits of Block 5. During these 145 scan cycles, Seed 4 was being scanned into the seed buffer. Next, the stall signal is set to 1 for 355 (= 500-145) cycles so that the entire Seed 4 is scanned into the seed buffer. Then the stall signal is again set to 0 for 110 cycles so that the specified bits of Block 4 are scanned into the scan chains, followed by 390 (= 500-110) cycles of stall = 1 so that Seed 3 is scanned into the scan buffer. This process is continued until the entire test pattern is scanned into the scan chains. Note that, only 2,000 bits had to be stored on the tester instead of the entire test pattern (~ 500,000 bits for a design with 500,000 flip-flops) that has to be stored in the serial mode with the conventional architecture. The time required to scan in this test pattern is equal to 500 scan cycles + 2,000 (= 355 + 390 + 450 + 350 + 455) stall cycles = 2,500 cycles. Note that, with a serial mode for a design with 500,000 flip-flops and 2 scan inputs (Fig. 2), it would have taken 250,000 cycles to scan in this test pattern with the conventional architecture.

Example 2: A test pattern with 800 specified bits

In the conventional architecture (Fig. 2), we apply a seed 1,000 bits long and it takes 500 scan cycles to generate the test pattern on-chip. In the new architecture of Fig. 3, the test pattern is divided into two blocks such that the specified bits in the block can be generated by a 500 bits long seed. Suppose that Block 1 contains 300 bit slices and Block 2 contains 200 bit slices. In this case, the tester memory requirement will be 1,000 bits (two seeds) and the application time is equal to 500 scan cycles + 500 (200 + 300) stall cycles = 1,000 cycles.

Example 3: A test pattern with 100 specified bits

In the conventional architecture of Fig. 2, we need a seed 1,000 bits long and 500 scan cycles. Using the new architecture, we require a seed 500 bits long and 500 scan cycles resulting in additional 2x compression for this test pattern.

Example 4: A set of 5 test patterns with a total of 400 specified bits with repeat count compression capability on the tester

The conventional architecture (Fig. 2) requires 5 seeds each being 1,000 bits long and 2,500 scan cycles. Since the total number of specified bits is only 400, we require a single seed of length 500. The number of required cycles is 2,500. There will be 2,000 (= 2,500-500) cycles during which it doesn't matter what is scanned into the seed buffer. In the presence of repeat count compression feature [Volkerink 02, Barnhart 01], these 2,000 bits of tester memory aren't wasted and the tester memory requirement will be only 500 bits for these 5 test patterns resulting in additional 10x compression over the conventional architecture. However, the phase shifter must be appropriately designed in this case so that there is no temporal dependency between the bit sequences applied to the scan chains for a period of 5 test patterns [Bardell 87].

Example 5: A set of 5 test patterns with a total of 400 specified bits with no repeat count compression capability on the tester

Following Example 4, only 2,500 bits are required for these 5 test patterns compared to 5,000 bits using the conventional architecture.

It must be very clear to the reader that, given a design with scan, the new reseeding architecture allows the designer to optimize test data volume and test time based on profiles of specified bits in test patterns generated for this design. Note that, with the concept of bit slices, we can apply reseeding techniques to any set of consecutive bit slices not necessarily restricted to test pattern boundaries. For example, if every test pattern has 500 bit slices, we can use a single seed to generate the last 200 consecutive bit slices of a test pattern followed by 400 consecutive bit slices of the next test pattern from a single seed. The number of bit-slices constituting a block isn't fixed but varies from one block to another. This makes our technique distinct from the compression scheme described in [Krishna 02]. The optimum solution for a given design depends on several factors such as the number of scan chains, the number of flip-flops, the number of test patterns, the profile of specified bits in test patterns, number of available scan pins, phase shifter design, the tester memory organization and the repeat count compression capability of the tester. The general guidelines are: (1) use as short scan chains as possible (to reduce test time), (2) cover as many test patterns as possible with a single seed, and (3) use stall signal instead of a serial mode to scan in test patterns with a lot of specified bits beyond the capability of what the LFSR can produce from a single seed.

Table 2: Experimental Results

	ASIC 1		ASIC 2	
	Compression Ratio	Efficiency	Compression Ratio	Efficiency
Conventional reseeding	2x	7%	8	7%
Reseeding with serial shift mode	8x	24%	18x	15%
New reseeding technique	**32x**	**91%**	**125x**	**93%**

4. Experimental results

We present results obtained from the application of the new reseeding architecture to two industrial designs ASIC1 and ASIC2. The characteristics of these designs are described in Table 1.

Table 1: Characteristics of ASIC1 and ASIC2

Parameters	ASIC1	ASIC2
Gate count	~300K	~7M
Flip-flop count	~30K	~105K
Test Data Volume	~18M	~1.6G

For the above designs, single-stuck fault test patterns with don't cares (also referred to as test cubes in many publications) were generated using TetraMax, a commercial ATPG tool. No constraint was set on the number of specified bits per test pattern and the ATPG was performed with maximum dynamic and static compaction. The scan test patterns were post-processed to generate the results.

It was assumed that given a test pattern it is possible to find a seed such that the LFSR generates the specified bits of that test pattern from the seed state as long as the seed size exceeds the number of specified bits in the test pattern by 20 or more [Koenemann 91]. It was also assumed that the tester allows repeat count compression per scan-in channel.

The compression software was executed on a HP 9000/785/J5600 machine with 2 550MHz PA-RISC processors and 2G of physical RAM.

The obtained results are presented in Table 2. We evaluated two metrics for comparison purposes. The first metric is the *compression ratio*, which is the ratio of the total number of tester memory bits required with no compression to the total number of tester memory bits required with compression. The other metric is *efficiency*, which is defined to be equal to the ratio of the total number of specified bits in all test patterns to the total number of tester memory bits required with compression. When efficiency is equal to 1, the number of tester memory bits required with compression is equal to the total number of specified bits in all test patterns. Note that, the objective of any compression scheme should be to maximize both compression ratio and efficiency.

By conventional reseeding we mean that the seed size is chosen to be equal to the number of specified bits in the worst test pattern. For example, for ASIC1, the worst test pattern has 20,000 specified bits (Fig. 1). For reseeding with serial shift mode, we chose a seed size such that all test patterns with the number of specified bits more than seed size-20 are serially shifted in using the serial mode. All other test patterns are generated by the LFSR from the corresponding seed.

The chosen seed size was obtained after numerous experiments to obtain the one that achieves maximum compression ratio. For ASIC1, the seed size for reseeding with serial mode was chosen to be equal to 1,200. For our new reseeding technique, the seed size was chosen such that we have maximum compression ratio. It took around 4 hours to find the best seed size. For ASIC1, the seed size was 400 bits, the number of seed buffer inputs is equal to 5 and the number of scan chains is 350 for our new reseeding technique with maximum compression ratio. For this architecture, the test time reduction was almost 28 times compared to the test time for the same design with 5 scan chains (with 5 scan-ins) and no compression. As shown in table 2, better results are obtained for ASIC 2.

We also collected results on the application of our technique to the reseeding scheme described in [Hellebrand 95] which is a variant of Koenemann's scheme [Koenemann 91] because it uses multiple LFSR polynomials. The results are very similar to those in Table 2. Hence, these results are not presented in a separate table.

5. Conclusions

A new technique for test data compression using LFSR-reseeding is presented in this paper. The technique is very practical, can be easily implemented in existing designs and doesn't require any special tester features. Experimental results on industrial designs clearly demonstrate the effectiveness of this technique over other reseeding techniques. We're currently investigating techniques for automated synthesis of optimum scan architectures with this new reseeding technique.

6. Acknowledgements

Erik Volkerink was supported by Agilent Technologies, Inc.

7. References

[ITRS 99] International Technology Roadmap for Semiconductors (ITRS), 1999.

[Bardell 87] Bardell, P.H., W. H. McAnney and J. Savir, "*Built-in Test For VLSI: Pseudo-random Techniques*", Wiley Inter-Science, 1987.

[Barnhart 01] Barnhart, C., B. Keller, B. Koenemann and R. Walther, "OPMISR: The Foundation for Compressed ATPG Vectors", *Proc. Intl. Test Conf.*, pp. 748-757, 2001.

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

[Koenemann 91] Koenemann, B., "LFSR-Coded Test Patterns for Scan Designs", *Proc. European Test Conf.*, pp. 237-242, 1991.

[Koenemann 01] Koenemann, B., C. Barnhart, B. Keller, T. Snethen, O. Farnsworth, D. Wheater, "A SmartBIST Variant with Guaranteed Encoding", *Proc. Asian Test Symp.*, pp.325-330, 2001.

[Krishna 01] Krishna, C.V., A. Jas and N.A. Touba, "Test Vector Encoding Using Partial LFSR Reseeding", *Proc. Intl. Test Conf.*, pp. 885-893, 2001.

[Krishna 02] Krishna, C.V., and N. A. Touba, "Reducing Test Data Volume Using LFSR Reseeding with Seed Compression", *Proc. Intl. Test Conf.*, pp. 321-330, 2002.

[Rajski 02] Rajski, J., J. Tyszer, M. Kassab, N. Mukherjee, R. Thompson, H. Tsai, A. Hertwig, N. Tamarapalli, G. Mrugalski, G. Eidel, J. Qian, "Embedded Deterministic Test for Low Cost Manufacturing Test", *Proc. Intl. Test Conf.*, pp. 301-310, 2002.

[Volkerink 02] E. Volkerink, A. Khoche, S. Mitra, "Packet-based Test Vector Compression," *Proc. Intl. Test Conf.*, pp. 154-163, 2002.

[Zacharia 95] N. Zacharia, J. Rajski, J. Tyszer, "Decompression of Test Data using Variable-Length Seed LFSRs," *Proc. IEEE VLSI Test Symp.*, pp. 426-433, 1995.