LOW POWER AND TEST DATA COMPRESSION IN VLSI TESTING USING NEW ENCODING SCHEME

V.Govindaraj[1], B.Jaishankar[2], S.Balamurali[3], P.Balamurugan[4]
[1 2 3]Assistant Professor, ECE, KPR Institute of Engineering and Technology , Tamilnadu, India
[4] Associate Professor, ECE, KSR College of Engineering , Tamilnadu, India

---

Abstract -*Power dissipation during test is a significant problem as the size and complexity of systems-on-chip (SOCs) continue to grow. During scan shifting, more transitions occur in the flip-flops compared to what occurs during normal functional operation. This problem is further compounded when pseudorandom filling of the unassigned input values is employed. Excessive power dissipation during test can increase manufacturing costs by requiring the use of a more expensive chip packaging or causing unnecessary yield loss. In this project, a new test-data-compression scheme based on linear feedback shift register (LFSR) reseeding that significantly reduces power consumption during test is proposed. Test-data volume has also increased dramatically as the size and the complexity of chips grow. A large number of test pattern bits being assigned randomly cause a large number of transitions in the scan chains thereby increasing power dissipation during test drastically. To overcome this a new encoding algorithm is Proposed to achieve test data compression and low power dissipation.*

Keywords –power dissipation, test data compression,vlsi,LFSR.

1. INTRODUCTION

As the size and complexity of systems-on-a-chips (SOCs) continue to grow, the cost of VLSI test is increasing drastically. Larger chips require a larger amount of test data and dissipate a larger amount of power during test. Moreover, they are typically harder to test because they tend to have more hard-to-detect faults. Test time is a critical part of test cost and increases as the size and complexity of a chip increase. Reducing test cost is becoming an increasingly critical issue. This dissertation focuses on two important sources of the test cost, namely test data volume and test power. Test time depends on both test data volume and test power. Large test data volume increases test time because it requires more time to transfer the data to and from the chip.

Test power can slow down test speed, thereby increasing test time. If the average power Consumption during test is higher than the chip package's capability to dissipate heat; the test must be run at a lower frequency. Therefore, both test power and test data should be considered to reduce test time effectively. Conventional test data compression schemes generally dissipate high power. Most conventional compression schemes exploit the fact that a test set has a large number of don't cares and only 1~5% of specified (care) bits. The don't cares are assigned to maximize compression. In this process, a large number of transitions may occur in test patterns. The larger the number of transitions in the test patterns, the larger the power dissipation. This dissertation addresses these two important problems in the VLSI testing area, namely test data volume and test power.

1.1 TEST POWER VS. TEST DATA

Conventional test data compression schemes generally increase test power. Most conventional test data compression techniques are based on the fact that a large percentage of test set, typically 90%~95%, is filled with don't care bits. The don't care bits are assigned in a way that minimizes test data volume and not test power. To reduce the number of transitions in a chip during test, the don't care bits should be set to constant values.

Then, test power dissipation will be minimized, but the don't care bits would not be used for test data compression. On the other hand, if the don't care bits are used for test data compression, then the don't care bits cannot be used for test power reduction. For example, in linear feedback shift register (LFSR) reseeding scheme that is used in several commercial tools including TestKompress by Mentor Graphics and DBIST by Synopsys, the don't care bits are assigned almost randomly, which results in large power dissipation. This is why test power can be a serious problem in test data compression techniques

1.2 LFSR RESEEDING

The basic idea in LFSR reseeding is to generate deterministic test cubes by expanding seeds. A seed is an initial state of the LFSR that is expanded by running the LFSR in autonomous mode. Since typically only 1-5% of the bits in a test vector are specified, most bits in a test cube do not need to be considered when a seed is computed because they are don't care bits. Therefore, the size of a seed is much smaller than the size of a test vector. Consequently, reseeding can significantly reduce test data storage and bandwidth [12]. Many test data compression schemes are based on LFSR reseeding. Causing high power dissipation. We present a new encoding scheme that can be used in Conjunction with any LFSR reseeding scheme to significantly reduce test power and even further reduce test storage.

2. PROPOSED METHODOLOGY

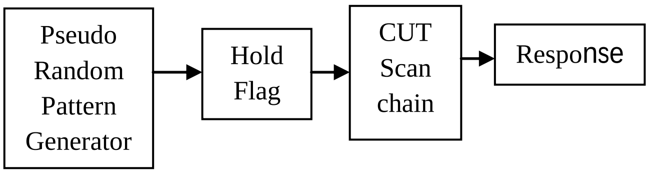

Fig 1. Basic Block Diagram

The proposed encoding scheme acts as a second stage of compression after LFSR reseeding. It accomplishes two goals. First, it reduces the number of transitions in the scan chains (by filling the unspecified bits in a different manner), and second it reduces the number of specified bits that need to be generated via LFSR reseeding. Experimental results indicate that the proposed method significantly reduces test power and in most cases provides greater test data compression than LFSR reseeding alone.

2.1. ENCODING ALGORITHM

Let a transition in a test cube be defined as a specified 0 (1) followed by zero or more X's followed by a specified 1 (0). The key idea of the proposed encoding algorithm is to take advantage of the fact that number of transitions in a test cube is always less than the number of specified bits in a test cube. Thus, rather than using LFSR reseeding to directly encode the specified bits as in conventional LFSR reseeding, the proposed encoding algorithm divides the test cube into blocks and only uses LFSR reseeding to produce the blocks that contain transitions. For the blocks that do not contain transitions, the logic value fed into the scan chain is simply held constant.

This approach reduces the number of transitions in the scan chains and in most cases also reduces the total number of specified bits that must be generated by the LFSR as compared with conventional LFSR reseeding

2.2 Basic Concept

	BLOCK	BLOCK	BLOCK	BLOCK
original	0 X X 1	X 1 1 1	1 X 1 X	X X X X
encoded	0 0 X X 1	1 X 1 1 1	11X1 X	X X X XX

The proposed encoding scheme encodes each test cube with two kinds of data: hold flags and data bits. Each test cube is divided into several blocks and each block has a one-bit hold flag. The hold flag indicates whether a transition occurs in a block.

There are three types of blocks:

1) Transition block (Hold flag = 0)

One or more transitions exist in the block. Either both 0 and 1 are present in the block (e.g., XX1X0X), or only 0 or 1 is present but the last specified bit from a previous block was opposite.

2) Non-transition block (Hold flag = 1)

No transition occurs in current block. Only 0 or 1 is present in the block, and the last specified bit from a previous block is same (e.g., X0XX0X).

Table 1. Example of encoding test data

3)Don't care block (Hold flag = X)

No specified bits occur in the block, all are don't cares. If the hold flag for a block is 1, then the data bits in the block are simply held constant from the last data bit in the previous block. If the hold flag is 0, then the data bits are loaded directly from the LFSR. If the hold flag is X, then it can be either treated as a non-transition block or as a transition block with all X data. Both the hold flags and the data bits are generated from a single LFSR using reseeding.

An example of the proposed encoding is shown in Table 1. The test sequence in the example is composed of 4 blocks and each block has 1 hold flag and 4 data bits. The hold flags are shown in bold in the "Encoded" bit sequence row. The original test cube contains 7 specified bits. However, using the proposed encoding scheme, the encoded data has only 3 specified hold flags and 2 specified data bits giving a total of only 5 specified bits.

	BLOCK	BLOCK	BLOCK	BLOCK
Original	X 0 1 X	X 0 X 0	X X X X	1 1 1 X
Encoded	0 X 0 1 X	1 X 0 X 0	0 XX X 1	1 1 1 1 X

Table 2. Example of encoding test data

Thus, the proposed encoding scheme reduces the number of specified bits that need to be generated using LFSR reseeding. As shown in table2 the 1's in block 2 and block 3 don't need to be generated directly by the LFSR, but are rather generated as a by-product of the fact that the hold flags keep the input to the scan chain held constant at 1. Thus, test data compression can be achieved in this way.

Moreover, no transitions will occur when generating block 2 and block 3 because the hold flags are 1 thus keeping all the bits in the blocks constant. This would not be the case in conventional LFSR reseeding where the X's in blocks 1 and 2 get filled with random data which may results in many more transitions. Thus, a reduction in the number of transitions can be achieved in this way.

2.3 Conversion Procedure

It is possible to increase the number of non-transition blocks by converting some transitions blocks into non-transition blocks. There are two requirements that must be satisfied in order to convert a transition block into a non-transition block. The first is that it cannot contain both specified 0's and specified 1's. The second is that the last bit of the previous block must be an X. Two examples of this are shown in table3.1.

Block 2 is initially a transition block even though it only contains specified 0's because the last specified bit in block 1 was a 1. However, the very last bit of block 1 is a don't care, so a conversion

Table 1. Example of encoding test data

procedure can be used to specify that don't care as a 0 and thereby convert block 2 into a non-transition block. Even though this conversion required adding an extra specified data bit, the net result is still a reduction in the total number of specified bits because now block 2 is a non-transition block and thus none of its data bits need to be generated by the LFSR.

This same conversion procedure can also be used to convert block 4 in Table 1. In to a non-transition block. By increasing the number of non-transition blocks, the conversion procedure can help to reduce both test storage (since it can reduce the total number of specified bits) as well as test power (since it can reduce the number of transitions by enabling all the X's in the converted non-transition block to be filled with the same logic value Convert blocks 2 and 4 into non-transitions blocks) .

Table 2. Example of conversion procedure (last bit of blocks 1 and 3 are specified to convert blocks 2 and 4 in to non-transitions blocks

2.4. Partitioning into Hold Cube Compatible Sets

The test storage for LFSR reseeding depends on the number of specified bits. For each block that is not a don't care block, the hold flag for that block is specified. If the number of specified hold flags becomes larger than the number of the specified test data bits that are reduced by using the proposed encoding scheme, then the encoding scheme would be reducing test power dissipation at the cost of test storage. The test storage would increase because the total number of specified data bits plus specified hold bits would exceed the total number of specified bits in the original test cubes. However, in this chapter, a method for reducing the number of specified hold flags is introduced. The key idea is to take advantage of the fact that many test cubes may have compatible assignments in their corresponding hold flags. We will denote the set of hold flags for one test cube as a hold cube since each hold flag can be either a 1, 0, or don't care (X). If several consecutive test cubes have the same hold cube, it is not necessary to change any of the hold flags. Thus, the hold flags could be loaded once and then reused when applying subsequent test cubes.

The hold cubes for a pair of test cubes are compatible if they do not conflict in any specified bit positions. In other words, for every bit position where one hold cube has a specified value, the other hold Cube has either the same specified value or a don't care (and vice versa). Let a hold cube compatible set be defined as a set of test cubes with mutually compatible hold cubes. Since typically only around 1-5% of the data bits in a test cube are specified, the corresponding hold cube will typically have a large number of don't cares

present for LFSR reseeding anyway) and some small combinational logic. The size of the HF-SR dominantly determines the hardware overhead in this scheme. It depends on the number of scan chains and the total number of blocks.

3. SIMULATION RESULTS

The code for LFSR and LFSR Reseeding with and without Encoding Algorithm code is written in VHDL and simulated using MODELSIM 5.7G

3.1 LFSR Reseeding Without Algorithm

International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 02 Issue: 01 | March-2015 www.irjet.net p-ISSN: 2395-0072

By observing the result when the clock is high the LFSR generates the Test patterns and it shifted to the scan chain, the scan chain data are fed to input of the test circuit .In this method the test patterns are high, they take more time to process the data and it's power consumption is measured with the help of XILINX power analysis tool.

Fig 2 simulation result of S27 without Algorithm

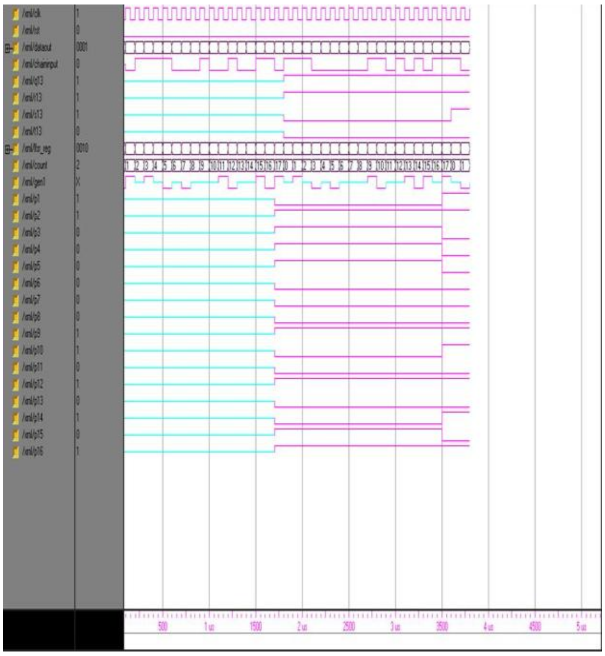

Fig 3 simulation result of C17 without Algorithm

3.2 LFSR Reseeding With Algorithm

By observing the result when the clock is high the LFSR generates the Test patterns, if the hold flag is set, the encoded data are fed to the scan chain otherwise generated LFSR test patterns are fed to the scan chain. The scan chain data are fed to input of the test circuit, and it's power consumption is measured with the help of XILINX power analysis tool.

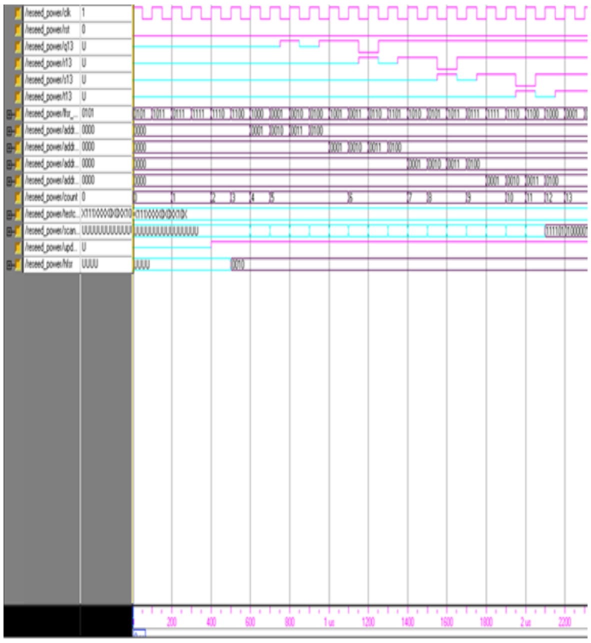

Fig4.simulation result of S27 with Algorithm

Table 3. Comparison of with and without algorithm

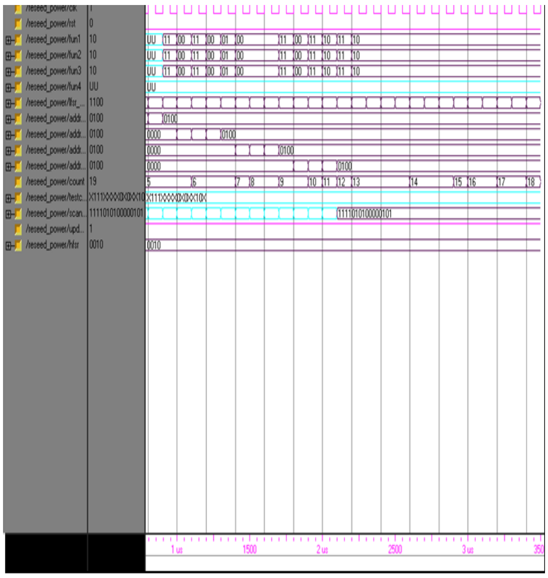

Fig 5. simulation result of C17 with Algorithm

Criteria	Circuit name	LFSR reseeding without algorithm	LFSR reseeding with algorithm
Power in Mw	S27	21	16
Power in Mw	C17	31	22
Test data volume	S27	543	256
Test data volume	C17	634	342

3.3 Comparison of LFSR reseeding without algorithm along with LFSR reseeding with algorithm

The Method LFSR Reseeding without algorithm consumes more power and the device utilization is more, compare to LFSR Reseeding with circuit

4. CONCLUSION

LFSR reseeding is a powerful approach for reducing test storage. The proposed Encoding scheme provides a way to reduce test power for LFSR reseeding. It acts as a second stage of compression after LFSR reseeding. By employing hold flags, not only is test power reduced, but also test storage can be reduced. Further the Encoding scheme will be implemented in BIST Environment and will be tested with the help of Benchmark Circuits using MODELSIM and XILINX tool.

REFERENCES

[1] Harikrishna Parmar and Prof. Usha Mehta "A statistical test data compression technique with adaptive bit filling and AI based reordering: optimization for compression and scan power" International Journal of VLSI and Signal Processing Applications, Vol. 1, Issue 2 , May 2011,(15-24).

[2] Chandra, A., and K. Chakrabarty(2002), "Reduction of SOC Test Data Volume, Scan Power and Testing Time Using Alternating Run-length Codes," Proc. Of Design Automation Conference, pp. 673-678.

[3] Rosinger, P. M., B.M. Al-Hashimi, and N. Nicolici(2002), "Low Power Mixed- Mode BIST Based on Mask Pattern Generation Using Dual LFSR Re-seeding," Proc. of Int. Conference on Computer Design, pp. 474-47.

[4] Samaranayake, S., N. Sitchinava, R. Kapur, M.B. Amin(2002), and T.W. Williams, "Dynamic Scan: Driving Down the Cost of Test," Computer, Vol. 35, Issue 10, pp. 63 - 68,

[5] Sankaralingam, R., R.R. Oruganti, and N.A. Touba(2000), "StaticCompaction techniques to control scan vector power dissipation," Proc. of VLSI Test Symp., pp. 35-40

[6] N. Zacharia, J. Rasjski, and J. Tyszer(1995), "Decompression of test data using variable-length seed LFSRs," in Proc. VLSI Test Symp., pp. 426–433.

[7] Hellebrand, S., S. Tarnick, J. Rajski, and B. Courtois(1992), "Generation of Vector Patterns through Reseeding of Multiple-Polynomial Linear Feedback Shift Register," Proc. of International Test Conference, pp. 120-129.

[8] Krishna, C.V., A. Jas, and N.A. Touba(2001), "Test Vector Encoding Using Partial LFSR Reseeding", Proc. of IEEE International Test Conference, pp. 885-893.

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

[10] Krishna, C.V., and N.A. Touba(2001), "Reducing Test Data Volume Using LFSR Reseeding with Seed Compression ", Proc. of IEEE International Test Conference, pp. 321-330.

[11] Koenemann, B(1991)., "LFSR-Coded Test Patterns for Scan Designs," Proc. of European Test Conference, pp. 237-242.

[12] Jinkyu Lee and Nur A. Touba(2007),LFSR-Reseeding Scheme Achieving Low-Power Dissipation During Test", Proc. of European Test Conference, pp. 237-242

