P. Vasantha* et al.
(IJITR) INTERNATIONAL JOURNAL OF INNOVATIVE TECHNOLOGY AND RESEARCH
Volume No.5, Issue No.3, April – May 2017, 6125-6131.

A Modified Fault Coverage Architecture For A Low Power BIST Test Pattern Generator Using LP-LFSR

P.VASANTHA
Avanthi Institute of Engineering and Technology, Bhogapuram Mandal, Tagarapuvalasa, Vizianagaram Dist

ADIGARLA MADHUSUDHANA RAO
Assistant Professor, Avanthi Institute of Engineering and Technology, Bhogapuram Mandal, Tagarapuvalasa, Vizianagaram Dist

Abstract: **This paper proposes low power pseudo random Test Pattern generation .This test pattern is run on the circuit under test for desired fault coverage. The power consumed by the chip under test is a measure of the switching activity of the logic inside the chip which depends largely on the randomness of the applied input stimulus. Reduced correlation between the successive vectors of the applied stimulus into the circuit under test can result in much higher power consumption by the device than the budgeted power. A new low power pattern generation technique is implemented using a modified conventional Linear Feedback Shift Register which can perform fault analysis and reduce the power of a circuit during test by generating three intermediate patterns between the random patterns by reducing the hardware utilization. The goal of having intermediate patterns is to reduce the transitional activities of Primary Inputs (PI) which eventually reduces the switching activities inside the Circuit under Test (CUT) and hence power consumption is also reduced without any penalty in the hardware resources.**

I. INTRODUCTION

The main challenging areas in VLSI are performance, cost, and power dissipation. Due to switching i.e. the power consumed testing, due to short circuit current flow and charging of load area, reliability and power. The demand for portable computing devices and communications system are increasing rapidly. These applications require low power dissipation VLSI circuits. The power dissipation during test mode is 200% P more than in normal mode. Hence it is important aspect to optimize power during testing. Power optimization is one of the main challenges.

Although over the next years, the primary objective of manufacturing test will remain essentially the same to ensure reliable and high quality semiconductor products—conditions and consequently also test solutions may undergo a significant evolution. The semiconductor technology, design characteristics, and the design process are among the key factors that will impact this evolution. With new types of defects that one will have to consider to provide the desired test quality for the next technology nodes such as 3-D, it is appropriate to pose the question of what matching design-for-test(DFT) methods will need to be deployed. Test compression, introduced a decade ago, has quickly become the main stream DFT methodology. However, it is unclear whether test compression will be capable of coping with the rapid rate of technological changes over the next decade. Interestingly, logic built-in self-test (LBIST), originally developed for board, system, and in-field test, is now gaining acceptance for production test as it provides very robust DFT and is used increasingly often with test compression. This hybrid approach seems to be the next logical evolutionary step in DFT. It has potential for improved test quality; it may augment the abilities to run at-speed power aware tests, and it can reduce the cost of manufacturing test while preserving all LBIST and scan compression advantages.

Attempts to overcome the bottleneck of test data bandwidth between the tester and the chip have made the concept of combining LBIST and test data compression a vital research and development area. Thorough surveys of relevant test compression techniques can be found, for example, As with conventional scan-based test, hybrid schemes, due to the high data activity associated with scan-based test operations, may consume much more power than a circuit under test was designed to function under. With overstressing devices beyond the mission mode, reductions in the operating power of ICs in a test mode have been of concern for years. Full toggle scan patterns may draw several times the typical functional mode power, and this trend continues to grow, particularly over the mission mode's peak power.

II. PROJECT DESCRIPTION

The test pattern generator produces test vectors that are applied to the tested circuit during pseudo-random testing of combinational circuits. The nature of the generator thus directly influences the fault coverage achieved. the influence of the type of pseudo-random pattern generator on stuck-at fault coverage. Linear feedback shift registers (LFSRs) are mostly used as test pattern generators, and the generating polynomial is primitive to ensure the maximum period. We have shown that it

is not necessary to use primitive polynomials, and moreover that their using is even undesirable in most cases. This fact is documented by statistical graphs.

The necessity of the proper choice of a generating polynomial and an LFSR seed is shown here, by designing a mixed-mode BIST for the ISCAS benchmarks as the complexity of VLSI circuits constantly increases, there is a need of a built-in self-test (BIST) to be used. Built-in self-test enables the chip to test itself and to evaluate the circuit's response. Thus, the very complex and expensive external ATE (Automatic Test Equipment) may be completely omitted, or its complexity significantly reduced. Moreover, BIST enables an easy access to internal structures of the tested circuit, which are extremely hard to reach from outside. There have been proposed many BIST equipment design methods. In most of the state-of-the-art methods some kind of a pseudorandom pattern generator (PRPG) is used to produce vectors to test the circuit. These vectors are applied to the circuit either as they are, or the vectors are modified by some additional circuitry in order to obtain better fault coverage. Then the circuit's response to these vectors is evaluated in a response analyzer. Usually, linear feedback shift registers (LFSRs) or cellular automata (CA) are used as PRPGs, for their simplicity. Patterns generated by simple LFSRs or CA often do not provide a satisfactory fault coverage. Thus, these patterns have to be modified somehow. One of the method that is designing low power lfsr that can perform fault analysis and reduce the power of a circuit during test by generating three intermediate patterns between the random patterns by reducing the hardware utilization. The goal of having intermediate patterns is to reduce the transitional activities of Primary Inputs (PI) which eventually reduces the switching activities inside the Circuit under Test (CUT) and hence power consumption is also reduced without any penalty in the hardware resources.

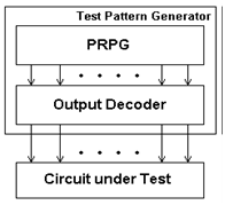


Figure 2.1. Test-per-clock BIST design

Now the LFSR width may be arbitrarily scaled. The effectiveness of this scaling and its possible extents are documented on BIST equipment design examples for some of the standard ISCAS benchmarks.

Mixed-Mode Column-Matching In the column-matching BIST design method the test pattern generator (TPG) consists of two parts: the pseudorandom pattern generator (PRPG), which is usually an LFSR and the output decoder. The output decoder is a combinational block transforming pseudorandom vectors into deterministic test patterns pre-computed by an ATPG tool. The method is designed for a test-per-clock BIST, i.e., the test patterns are fed to the circuit in parallel. Thus, the output decoder has as many inputs, as there are the PRPG outputs (LFSR bits) and as many outputs as there are CUT inputs. Figure 2.1. Test-per-clock BIST design the decoder is constructed using the column-matching algorithm, the inputs of the decoder is the PRPG patterns, the outputs are deterministic test vectors. The algorithm is designed to test combinational circuits only, thus the order, in which are the test patterns applied to the circuit, is insignificant. Thus, the vectors may be reordered in any way, i.e., we can freely decide, which PRPG vector will be "decoded" to obtain a particular deterministic vector. The main principle of the algorithm consists in trying to "match" as many decoder outputs with its inputs, by finding a suitable vector ordering. If an output is matched with an input, there will be no logic needed to implement this output; it will be implemented as a mere wire. Finding these matches is a simple permutation problem. Let us have an n-bit PRPG and an m-output CUT.

The decoder will be an n-input and m-output combinational block. There are n possibilities for a column match for each of the m outputs. Thus, there are n m combinations to test, to obtain an optimum matching. Such an algorithm complexity is prohibitively large, thus some heuristic must be used instead of a brute force approach. We use a "thorough search" algorithm, having an asymptotic complexity $O(n \cdot m\ 2 \cdot p \cdot s\ 2)$, where p is the number of PRPG patterns and s the number of deterministic vectors. Then the algorithm has been extended to support a mixed-mode BIST . Here the BIST is divided into two phases: the pseudorandom and deterministic one. The difference between our mixed-mode BIST method and the others is that the two phases are disjoint. First, the easy-to-detect faults are covered in the pseudo-random phase. Then, a set of deterministic test vectors covering the undetected faults is computed and these tests are then generated by a transformation of the subsequent PRPG patterns. This significantly reduces the decoder logic. A general scheme of the column-matching mixed-mode BIST is shown in Fig. 2.2. For sake of simplicity the number of LFSR bits (and thus the Decoder inputs) was set equal to the number of CUT inputs (m) here. Column-matching BIST scheme

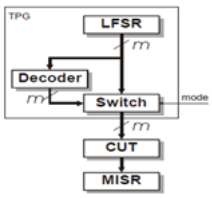


Figure 2.2. Column-matching BIST scheme

The whole mixed-mode column-matching based TPG design process can be summarized as follows:

1. Simulate several (PR) pseudo-random patterns for the CUT and determine the undetected faults (by fault simulation)

2. Compute deterministic test patterns detecting these faults by an ATPG tool

3. Perform the column-matching using the subsequent LFSR pseudo-random patterns and the deterministic tests

4. Synthesize the unmatched decoder outputs using a two-level Boolean minimizer.

BIST ARCHITECTURE:

A typical BIST architecture consists of

- TPG - Test Pattern Generator
- TRA – Test Response Analyzer
- Control Unit

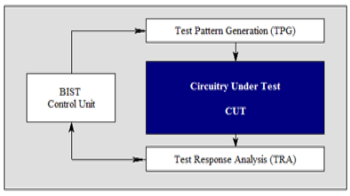


Fig 2.3 Test Pattern Generator

It generates test pattern for CUT. It will be dedicated circuit or a micro processor. Pattern generated may be pseudo random numbers or deterministic sequence. Here we are using a Low Power Linear Feedback Shift Register (LP-LFSR) for generating random numbers.

Test Response Analyzer (TRA): TRA will check the output of MISR & verify with the input of LFSR & give the result as error or not.

BIST Control Unit: Control unit is used to control all the operations. Mainly control unit will do configuration of CUT in test mode/Normal mode, feed seed value to LFSR, Control MISR & TRA. It will generate interrupt if an error occurs. You can clear interrupt by interrupt_clear_i signal.

Circuit under Test (CUT): CUT is the circuit or chip in which we are going to apply BIST for testing stuck at zero or stuck at one error.

Need for using BIST technique

Today's highly integrated multi-layer boards with fine-pitch ICs are virtually impossible to be accessed physically for testing. Traditional board test methods which include functional test, only accesses the board's primary I/Os, providing limited coverage and poor diagnostics for board-network fault. In circuit testing, another traditional test method works by physically accessing each wire on the board via costly "bed of nails" probes and testers. To identify reliable testing methods which will reduce the cost of test equipment, a research to verify each VLSI testing problems has been conducted. The major problems detected so far are as follows:

- Test generation problems
- Gate to I/O pin ratio

- **Test Generation Problems**

The large number of gates in VLSI circuits has pushed computer automatic-test-generation times to weeks or months of computation. The numbers of test patterns are becoming too large to be handled by an external tester and this has resulted in high computation costs and has outstripped reasonable available time for production testing.

- **The Gate to I/O Pin Ratio Problem**

As ICs grow in gate counts, it is no longer true that most gate nodes are directly accessible by one of the pins on the package. This makes testing of internal nodes more difficult as they could neither no longer be easily controlled by signal from an input pin (controllability) nor easily observed at an output pin (observe ability). Pin counts go at a much slower rate than gate counts, which worsens the controllability and observe ability of internal gate nodes.

III. BASIC ARCHITECTURE:

An n-bit PRPG connected with a phase shifter feeding scan chains forms a kernel of the generator producing the actual pseudorandom test patterns. A linear feedback shift register or a ring generator can implement a PRPG. More importantly, however, n hold latches are placed between the PRPG and the phase shifter. Each hold latch is individually controlled via a corresponding stage of an n-bit toggle control register. As long as its enable input is asserted, the given latch is transparent for data going from the PRPG to the phase shifter, and it is said to be in the toggle mode. When the latch is disabled, it captures and saves, for a number of clock cycles, the corresponding bit of PRPG, thus feeding the phase shifter (and possibly some scan chains) with a constant value. It is now in the hold mode. It is worth noting that each phase shifter

output is obtained by XOR-ing outputs of three different hold latches. Therefore, every scan chain remains in a low-power mode provided only disabled hold latches drive the corresponding phase shifter output the toggle control register supervises the hold latches. Its content comprises 0s and 1s, where 1s indicate latches in the toggle mode, thus transparent for data arriving from the PRPG. Their fraction determines a scan switching activity.

The control register is reloaded once per pattern with the content of an additional shift register. The enable signals injected into the shift register are produced in a probabilistic fashion by using the original PRPG with a programmable set of weights. The weights are determined by four AND gates producing 1s with the probability of 0.5, 0.25, 0.125, and 0.0625, respectively. The OR gate allows choosing probabilities beyond simple powers of 2. A 4-bit register Switching is employed to activate AND gates, and allows selecting a user-defined level of switching activity. Given the phase shifter structure, one can assess then the amount of scan chains receiving constant values, and thus the expected toggling ratio.

While preserving the operational principles of the basic solution, this approach splits up a shifting period of every test pattern into a sequence of alternating hold and toggle intervals. To move the generator back and forth between these two states, we use a T-type flip-flop that switches whenever there is a 1 on its data input. If it is set to 0, the generator enters the hold period with all latches temporarily disabled regardless of the control register content. This is accomplished by placing AND gates on the control register outputs to allow freezing of all phase shifter inputs. This property can be crucial in SoC designs where only a single scan chain crosses a given core, and its abnormal toggling may cause locally unacceptable heat dissipation that can only be reduced due to temporary hold periods. If the T flip-flop is set to 1 (the toggle period), then the latches enabled through the control register can pass test data moving from the PRPG to the scan chains. Two additional parameters kept in 4-bit Hold and Toggle registers determine how long the entire generator remains either in the hold mode or in the toggle mode, respectively. To terminate either mode, a 1 must occur on the T flip-flop input. This weighted pseudorandom signal is produced in a manner similar to that of weighted logic used to feed the shift register. The T flip-flop controls also four 2-input multiplexers routing data from the Toggle and Hold registers. It allows selecting a source of control data that will be used in the next cycle to possibly change the operational mode of the generator .For example, when in the toggle mode, the input multiplexers observe the Toggle register. Once the weighted logic outputs 1,the flip-flop toggles, and as a result all hold latches freeze in the last recorded state. They will remain in this state until another 1 occurs on the weighted logic output. The random occurrence of this event is now related to the content of the Hold register, which determines when to terminate the hold mode.

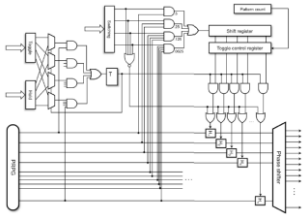


Fig. 3.1 Fully operational version of PRESTO.

IV. PROPOSED METHOD

LOW POWER PATTERN GENERATION:

One way to improve the correlation between the bits of the successive vectors is to avoid frequent transitioning of the logic levels of the primary inputs. The new approach entails inserting 3 intermediate vectors between every two successive vectors. The total number of signal transitions between these 5 vectors is equal to the total number of signal transitions between the 2 successive vectors generated using the conventional approach. This reduction of signal transition activity in the primary inputs reduces the switching activity inside the design under test and therefore results in reduced power Consumption by the device under test. The additional circuitry used to accomplish the generation of the 3 intermediate vectors is minimal at best consisting of few logic gates

The technique of inserting 3 intermediate vectors is achieved by modifying the conventional LFSR circuit with two additional levels of logic between the conventional flip-flop outputs and the low power outputs as shown in Figure 4.1. The first level of hierarchy from the top down includes logic circuit design for propagating either the present or the next state of the flip-flops to the second level of hierarchy. The second level of hierarchy is a multiplexer function that provides for selecting between the two states (present or next) to be propagated to the outputs as low power output. inimal at best consisting of few logic gates.

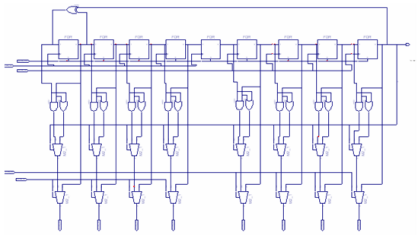


Fig 4.1 LP-LFSR

In the simulation environment, the outputs of the flip-flops are loaded with the seed vector. The feedback taps are selected pertinent to the characteristic polynomial $x^8 + x + 1$. Only 2 inputs pins, namely test enable and clock are required to activate the generation of the pattern as well as simulation of the design circuit. It is also noteworthy here that the intermediate vectors in addition to aiding in reducing the number of transitions can also empirically assist in detecting faults just as good as the conventional LFSR patterns. Description of the technique to produce low power pattern for BIST The following is a description of a low power test pattern generation technique as depicted in the 9-bit LFSR based schematic in Figure 4.1. Verilog based test bench as shown in Appendix B is used in assigning the initial output states (0100 1011) of the 9-bit LFSR. The feedback taps are designed for maximal length LFSR generating all zeros and all one's as well. The first step is to generate T1, the first vector by enabling (clocking) the first 4-bits of the LFSR and disabling (not clocking) the last 4 bits. This Shifts the first 4 bits to the right by one bit. The feedback bits of the LFSR are the outputs of the 8th and the first flip-flop. The output of the 8^{th} flip-flop is 1 and the output of the first flip-flop is 0. The exclusive-or of the 8^{th} -flip-flop (logic 1 in this case) and the first flip-flop(logic 0 in this case) is input (1 EXOR 0 = 1 into the first D flip-flop. The new pattern in the first four bits of the LFSR is 1010. Note that the shaded register is clocked along with the first 4 bits of the LFSR. So the input of the shaded flip-flop is the output of the 4^{th} flip-flop which in this case is 0. Also note that prior to the first clock, the input of the shaded register was the seed value of the 4^{th} flip-flop at the output of the 4^{th} flip-flop which in this case is 0. So after the first clock this value of 0 will now appear at the output of the shaded flip-flop. In other words the value of the 4^{th} output is stored in this shaded register and is used in the next few steps. The first 4 shifted bits of the LFSR and the last 4 un-shifted bits (i.e. the seed value) are propagated as T1 (1010 1011) to the final outputs. Next few steps involve generating the 3 intermediate patterns from T1. These patterns are defined as Ta, Tb and Tc shown in below flow.

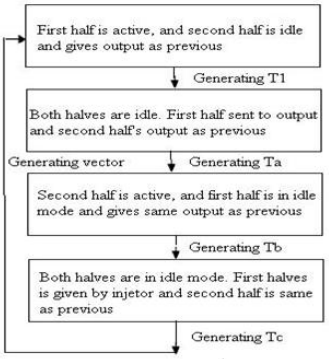


Fig 4.2 Proposed algorithm for low power LFSR

Ta is generated by maintaining (disabling the clock to the first 4 bits) the first four bits of the LFSR outputs (as is from T1) as the final first four low power outputs 1010. Note that the clock to the last four bits of the LFSR is also disabled.

The last four bits however are the outputs from the injector circuits. The injector circuit compares the next value (the input of the D-flip-flop) with the current value (the output of the D-flip-flop). According to T1, the outputs (current values) of the last 4 bits of the LFSR are 1011. The next values are the values at the inputs of the D-flip-flops which in this case are 0101**. C**ompare the current values (1011) bit by bit with the next values (0101). If the values bit by bit are not the same then use the random generator feedback R (in this case is logic 1) as the bit value as shown in the schematic above. If however both values bit by bit are the same then propagate that bit value to output as opposed to the R bit. This bit by bit comparison gives us the last four bits of Ta to be 1111. Therefore Ta = 1010 1111. Next step is to generate Tb. Shift the last 4 flip-flops to the right one bit but do not shift the first 4 flip-flops to the right. The clock to the first 4 bits plus the shaded flipflop is disabled. The clock to the last 4 bits is enabled. Propagate the outputs of the flip-flops of the entire LFSR as opposed to the outputs of the injection circuit to the outputs (low power). The injection circuits are disabled. As in Ta, maintain the first four LFSR outputs (1010) as the low power outputs. Again from Ta, the inputs of the last four D flip-flops from the previous step (generating Ta) are 0101. Also note that the output of the shaded register is 0 from the previous step (generating Ta). Therefore the input of the 5^{th} flip-flop is a 0. The outputs of the last 4 flip-flops are 0101 resulting in Tb = 1010 0101. The 3^{rd} intermediate vector Tc is generated via disabling the clock to the entire LFSR. Propagate the first 4 outputs from the injection circuit as the first 4 low power outputs and maintain the last 4 low power outputs the same as Tb. Generating injection circuit outputs for Tc is conceptually the same as explained above in

generating Ta. Current values (the outputs of the flip-flops) of the first four flip-flops are compared with the next values (the inputs of the flip-flops) of the flip-flops. The feedback from the 8th flip-flop is 1 (please see generating Tb). Therefore the logical feed forward value of R is 1. The feedback value from the first flip-flop is also 1 as per the current values above. The exclusive or of two ones is a 0. Therefore the input to the first flip-flop is a 0 which is also the next state of the first flip-flop. Hence the next values are 0 for the first flip-flop and 101 for the 2nd, 3rd and 4th flip-flop respectively. The next values are 0101. The first four outputs from the injection circuit are 1111. The last 4 outputs are the same as Tb which are 0101 resulting in the 3rd and final intermediate vector Tc = 1111 0101. Generating T2 is quite similar to generating T1. As in Tc the outputs of the last four LFSR flops are 0101. The outputs of the first 4 flip-flops of the LFSR are the current values which are 1010. Therefore the seed vector for generating T2 is 1010 0101. Shift the first four bits of the LFSR plus the shaded flip-flop. Do not clock the last four flip-flops. Propagate the outputs of the entire LFSR to the final low power outputs. The output of the 8th flip-flop from the previous step (generating Tc) is a 1 and the output of the first flip-flop from the previous step (generating Tc) is also a 1. The exclusive or of the output of the 8th flip-flop and the first flip-flop is 0. Therefore the input to the first flip-flop will be a 0. The inputs to the 2nd, 3rd, 4th and the shaded flip-flops are 1010. These are also the current values from the previous step (generating Tc). Shifting the first four flip-flops of the LFSR to the right by one bit results in 0101 as the outputs of the first four flip-flops. Therefore T2 generated is 0101 0101.

C17 bench mark circuit:

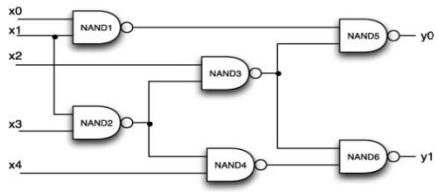
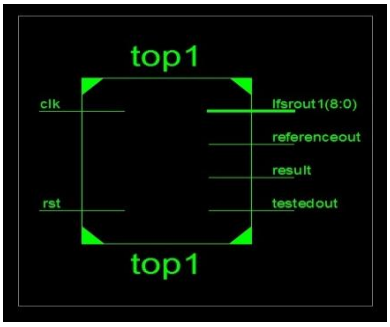


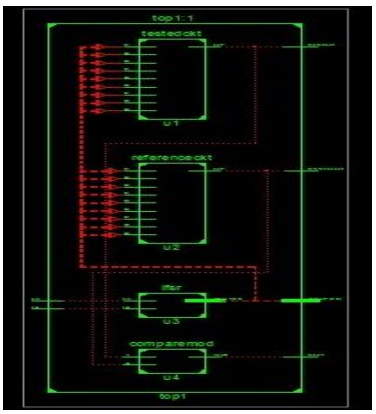
Fig 3.3 C17 Benchmark Circuit

V. RESULT

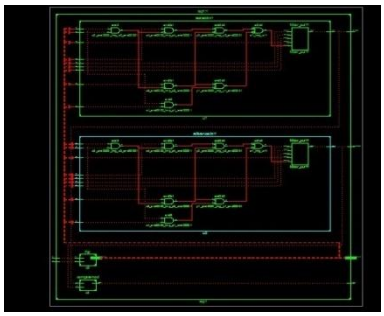
RTL SCHEMATIC DIAGRAM:



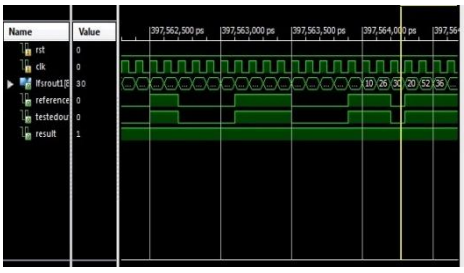
INTERNAL RTL SCHEMATIC:



Technology schematic:



WAVE FORMS:



DESIGN AND SUMMARY REPORTS:

top1 Project Status			
Project File:	prpg.xise	**Parser Errors:**	No Errors
Module Name:	top1	**Implementation State:**	Placed and Routed
Target Device:	xc3s500e-4fg320	**•Errors:**	No Errors
Product Version:	ISE 12.3	**•Warnings:**	11 Warnings (11 new)
Design Goal:	Balanced	**•Routing Results:**	All Signals Completely Routed
Design Strategy:	Xilinx Default (unlocked)	**•Timing Constraints:**	All Constraints Met
Environment:	System Settings	**•Final Timing Score:**	0 (Timing Report)

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Total Number Slice Registers	19	9,312	1%
Number used as Flip Flops	13		
Number used as Latches	6		
Number of 4 input LUTs	21	9,312	1%
Number of occupied Slices	17	4,656	1%
Number of Slices containing only related logic	17	17	100%
Number of Slices containing unrelated logic	0	17	0%
Total Number of 4 input LUTs	21	9,312	1%
Number of bonded IOBs	14	232	6%
Number of BUFGMUXs	1	24	4%
Average Fanout of Non-Clock Nets	3.43		

COMPARISON RESULTS:

By using low power LFSR, the area utilization is decreased with compared to the PRESTO method then the number of look up tables is decreased to 21 than existing method of 30 look up tables. Hence the Power is also decreased to 0.173mw from existing method of 0.249mw power and memory utilization is 187084kb with compared to memory of existing PRESTO (187724kb) and the delay is 5.492ns for existing prpg and 9.377ns for lowpower LFSR.

VI. CONCLUSION AND FUTURESCOPE

The proposed approach shows the concept of reducing the transitions in the test pattern generated. The transition is reduced by increasing the correlation between the successive bits. The simulation results shows that how the patterns are generated for the applied seed vector. This paper presents the implementation with regard to verilog language. Synthesizing and implementation (i.e. Translate, Map and Place and Route) of the code is carried out on Xilinx - Project Navigator, ISE 12.3i suite. The power reports shows that the proposed low power LFSR consumes less power during testing by taking the benchmark circuit C17. In future there is a chance to reduce the power somewhat more by doing modifications in the proposed architecture.

VII. REFERENCES

[1]. Michał Filipek, Grzegorz Mrugalski, Senior Member, IEEE, Nilanjan Mukherjee, Senior Member, IEEE, Benoit Nadeau-Dostie, Senior Member, IEEE, Janusz Rajski, Fellow, IEEE, Je˛drzej Solecki, and Jerzy Tyszer, Fellow, IEEE "Low-Power Programmable PRPG With Test Compression Capabilities",2014.

[2]. Michael L.Bushnell, Vishwani D.Agawal," Essentials of electronic testing for digital, memory and mixed-signal VLSI circuits," Kluwer Academic Publishers, 2000.

[3]. Mohammad Tehranipoor, Mehrdad Nourani, Nisar Ahmed," Low- Transition LFSR for BIST-Based Applications," 14th Asian Test Symposium, pp. 138- 143, 18-21 Dec. 2005.

[4]. F. Corno, P. Prinetto, M. Rebaudengo, M. Sonza Reorda," A Test Pattern Generation methodology for low power onsumption," pp.1-5, 2008.

[5] P.Glard et al "Survey 0 F Low-Power Testing Of Vlsi Circuits"IEEE Design & Test Of Computers, vol. 19, no. 3. (2002), pp. 80-90.

[6]. Dr.K.Gunavathi, Mr. K. ParamasivaM, Ms.P.Subashini Lavanya, M.Umamageswaran," A novel BIST TPG for testing of VLSI circuits", IEEE International Conference on Industrial and Information Systems, pp. 8 - 11, August 2006.

