The Power Optimization of Linear Feedback Shift Register Using Fault Coverage Circuits

K.YARRAYYA1, K CHITAMBARA RAO 2

Abstract— A new fault coverage test pattern generator using a linear feedback shift register (LFSR) called FC-LFSR 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. The experimental results for c17 benchmark, with and without fault confirm the fault coverage of the circuit being tested.

Index Terms — C17 BENCHMARK, CUT, FAULTCOVERAGE, LFSR.

1 INTRODUCTION

Test Pattern generation has long been carried out by using conventional Linear Feedback Shift Registers (LFSR's). LFSR's are a series of flip-flop's connected in series with feedback taps defined by the generator polynomial. The seed value is loaded into the outputs of the flip-flops. The only input required to generate a random sequence is an external clock where each clock pulse can produce a unique pattern at the output of the flip-flops.

The number of inputs required by the circuit under test must match with the number of flip-flop outputs of the LFSR. 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 [1].

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.

2 TESTING SCHEMES

Power dissipation is a challenging problem for today's System-on-Chips (SoCs) design and test. The power dissipation in CMOS technology is either static or dynamic. Static power dissipation is primarily due to the leakage currents and contribution to the total power dissipation is very small. The dominant factor in the power dissipation is the dynamic power which is consumed when the circuit nodes switch from 0 to 1. During switching, the power is consumed due to the short circuit current flow and the charging of load capacitances [1].

The power dissipation of a system in test mode is more than in normal mode. Low correlation between consecutive test vectors (e.g. among pseudorandom patterns) increases switching activity and eventually power dissipation in the circuit. The same happens when applying low correlated patterns to scan chains.Increasing switching activity in scan chain results in increasing power consumption in scan chain and combinational block. The extra power (average or peak) can cause problems such as instantaneous power surge causes circuit damage, formation of hot spots, difficulty in performance verification and reduction of the product yield and lifetime [2].

Large and complex chips require a huge amount of test data and dissipate a significant amount of power during test, which greatly increases the system cost. There are many test parameters should be improved in order to reduce the test cost. Parameters include the test power, test length (test application time), test fault coverage, and test hardware area overhead.

Automatic test equipment (ATE) is the instrumentation used in external testing to apply test patterns to the CUT, to analyze the responses from the CUT, and to mark the CUT as good or bad according to the analyzed responses. External testing using ATE has

a serious disadvantage, since the ATE (control unit and memory) is extremely expensive and cost is expected to grow in the future as the number of chip pins increases. As the complexity of modern chips increases, external testing with ATE becomes extremely expensive. Instead, Built-In Self-Test (BIST) is becoming more common in the testing of digital VLSI circuits since it overcomes the problems of external testing using ATE. BIST test patterns are not generated externally as in case of ATE [3].

BIST perform self-testing and reducing dependence on an external ATE. BIST is a Design-for-Testability (DFT) technique makes the electrical testing of a chip easier, faster, more efficient and less costly. The important to choose the proper LFSR architecture for achieving appropriate fault coverage and consume less power.Every architecture consumes different power for same polynomial. Applications of LFSR: Pattern generator, Low power testing, Data compression, and Pseudo Random Bit Sequences (PRBS).

2.1 BIST ARCHITECTURE

A typical BIST architecture consists of

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

As shown in Fig.:1 below.

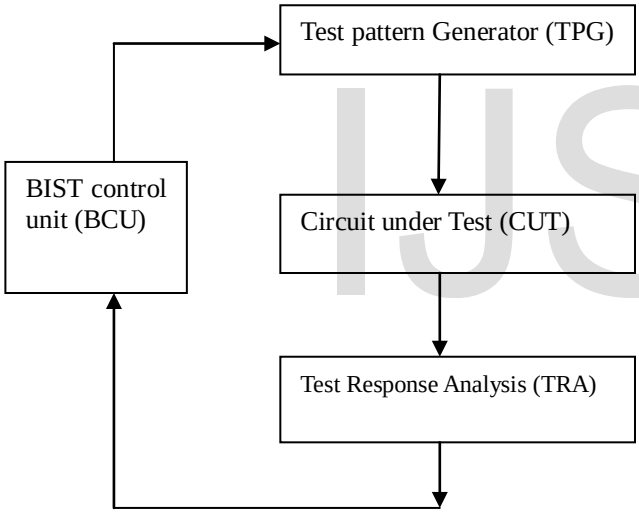

Fig.:1 Test Pattern Generator in BIST.

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 Linear Feedback Shift Register for generating random number. The Architecture for LFSR is as shown Fig. :2.

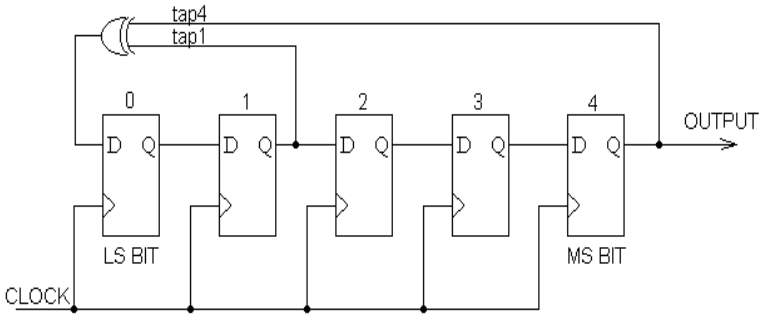

Fig. : 2 The Architecture for LFSR

Tapping can be taken as we wish but as per taping change the LFSR output generate will change & as we change in no of flip-flop the probability of repetition of random number will reduce. The initial value loading to the LFSR is known as seed value.

2.1.1 Test Response Analyzer (TRA)

TRA will check the output of MISR & verify with the input of LFSR & give the result as error or not.

2.1.2 BIST Control Unit (BCU)

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 signal.

2.1.3 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.

3 LOW TRANSITION PATTERN GENERATIONS

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 Fig.:3 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 [4]. 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. Minimal at best consisting of few logic gates.

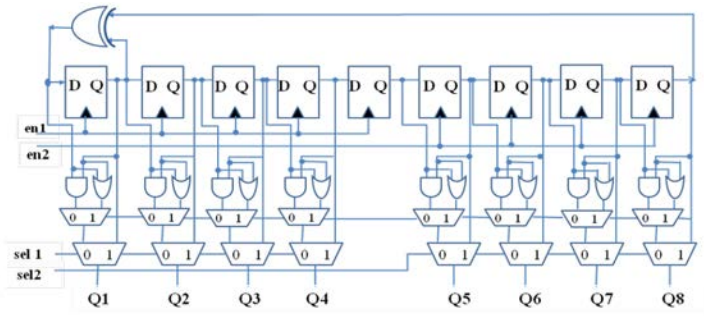

Fig.:3 Low power Test Pattern LFSR

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 [1] [5]. 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 3. Verilog based test bench as shown in Fig.:5 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.

3.1 ALGORITHM FOR LOW POWER LFSR

The first step is to generate Ti, 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 $8^{th}$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 Ti (1010 1011) to the final outputs. Next few steps involve generating the 3 intermediate patterns from Ti. These patterns are defined as Tk1, Tk2 and Tk3 shown in Fig.:4 below flow. Tk1 is generated by maintaining (disabling the clock to the first 4 bits) the first four bits of the LFSR outputs (as is from Ti) 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 Ti, 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 Tk1 to be 1111. Therefore TK1 = 1010 1111. Next step is to generate Tk2. 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 flip-flop 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 Tk1, maintain the first four LFSR outputs (1010) as the low power outputs. Again from Tk1, the inputs of the last four D flip-flops from the previous step (generating Tk1) are 0101. Also note that the output of the shaded register is 0 from the previous step (generating Tk1). Therefore the input of the 5^{th} flip-flop is a 0. The outputs of the last 4 flip-flops are 0101 resulting in Tk2 = 1010 0101.

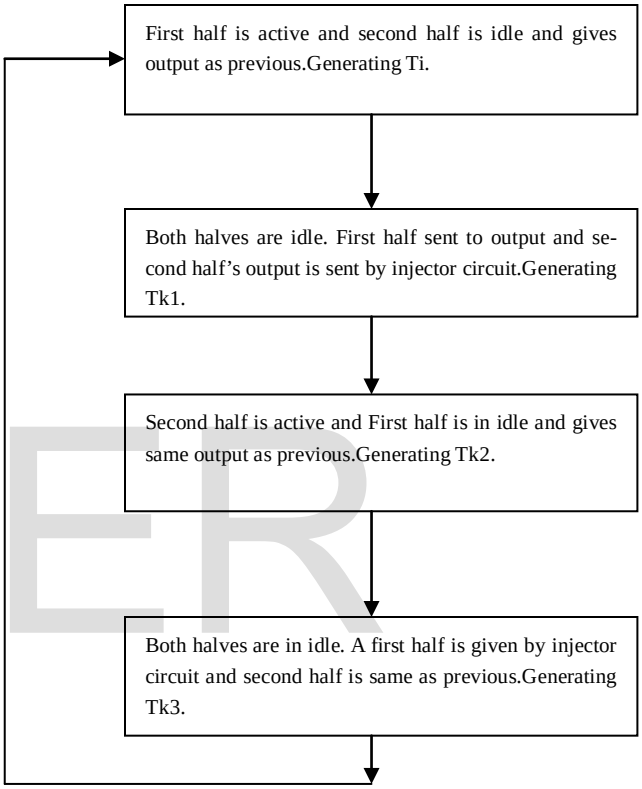

Fig.:4 Algorithms for low power LFSR

The 3^{rd} intermediate vector Tk3 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 Tk2. Generating injection circuit outputs for Tk3 is conceptually the same as explained above in generating Tk1. 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 8^{th}flip-flop is 1 (please see generating Tk2). 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 2^{nd}, 3^{rd} 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 Tk2 which are 0101 resulting in the 3rdand final intermediate vector Tk3 = 1111 0101. Generat-

ing Ti2 is quite similar to generating Ti. As in Tk3 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 Ti2 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 8^{th}flip-flop from the previous step (generating Tk3) is a 1 and the output of the first flip-flop from the previous step (generating Tk3) is also a 1. The exclusive or of the output of the 8^{th} 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 2^{nd}, $3^{rd}$and 4^{th} the shaded flip-flops are 1010. These are also the current values from the previous step (generating Tk3).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 Ti2 generated is 0101 0101.

The test patterns which are gesnerated using the low transition algorithm [6] [7] are applied to the C17 circuit shown in Fig.:5, whose outputs should be compared with the benchmark design by using the comparator. The comparator compares the obtained one with the original one and decides the tested IC is faulty or not [8].

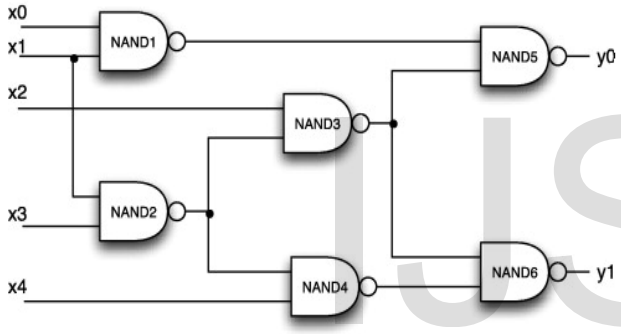

Fig.:5 C17 bench mark circuit.

4 SIMULATION RESULTS

The design is verified by performing behavioral simulation in Fig.:6, Fig.:7 and Fig.:8. To do, a high-level or behavioral simulator is used, which executes the design by interpreting the verilog cod like any other programming language, i.e. regardless of the target architecture. At this stage, FPGA development is much like software development signals and variables may be watched, procedures and functions may be traced, and breakpoints may be set. The entire process is very fast, as the design is not synthesized, thus giving the developer a quick and complete understanding of the design. The downside of behavioral simulation is that specific properties of the target architecture [1], namely timing and resource usage are not covered.

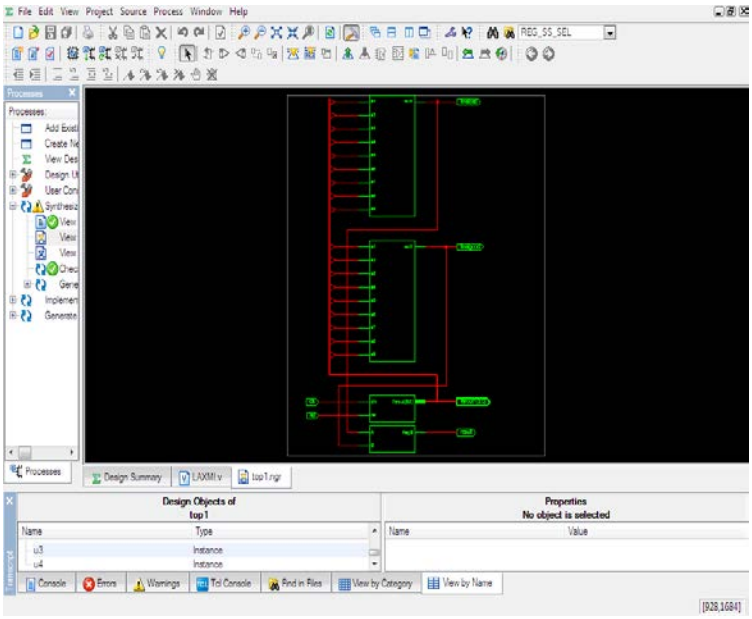

Fig.:6 RTL Schematic of LT-LFSR.

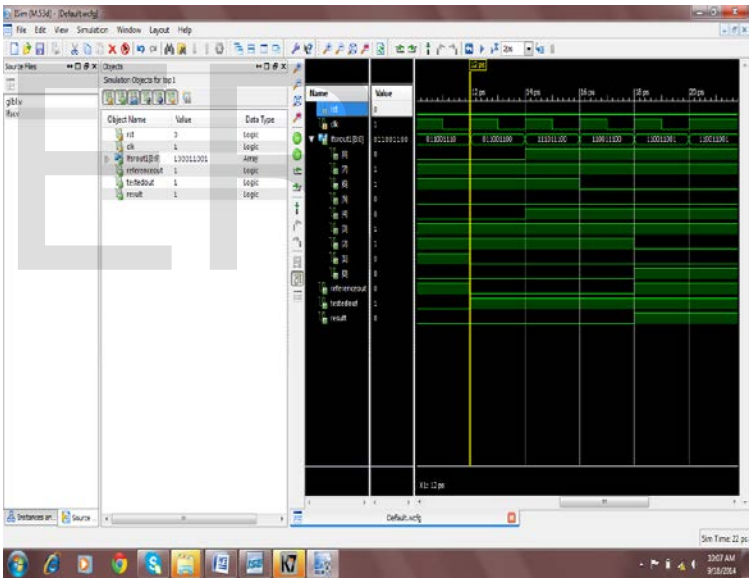

Fig.:7 Simulation report of LT-LFSR with fault.

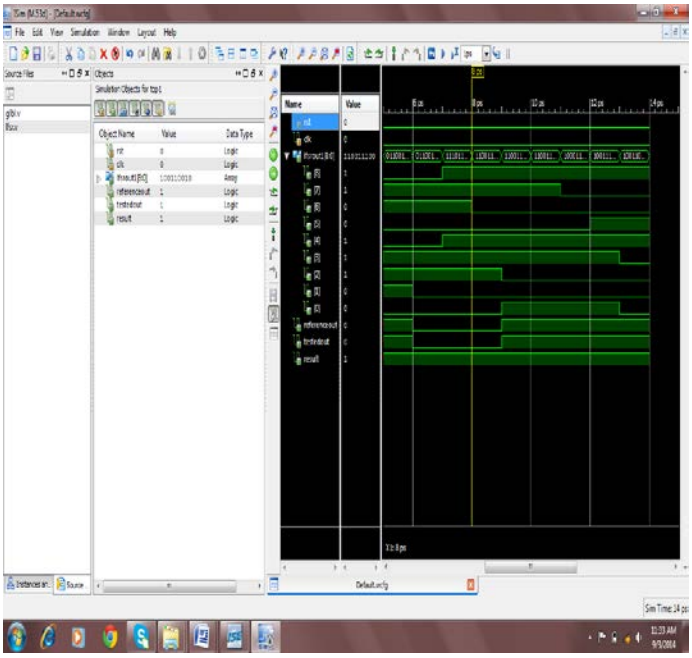

Fig.:8 Simulation report of LT-LFSR without fault.

5 SYNTHESIS RESULT

Synthesis is the process of translating Verilog HDL to a net list, which is built from a structure of micros, e.g. adders, multiplexers, and shift registers. Chip synthesizers perform optimizations, especially hierarchy flattening and optimization of combinational paths. Specific cores, like RAMs or ROMs are treated as black boxes. Recent tool can duplicate registers, perform re-timing or optimize thir results according to given constraints shown in Fig.:9, TableI and TableII.

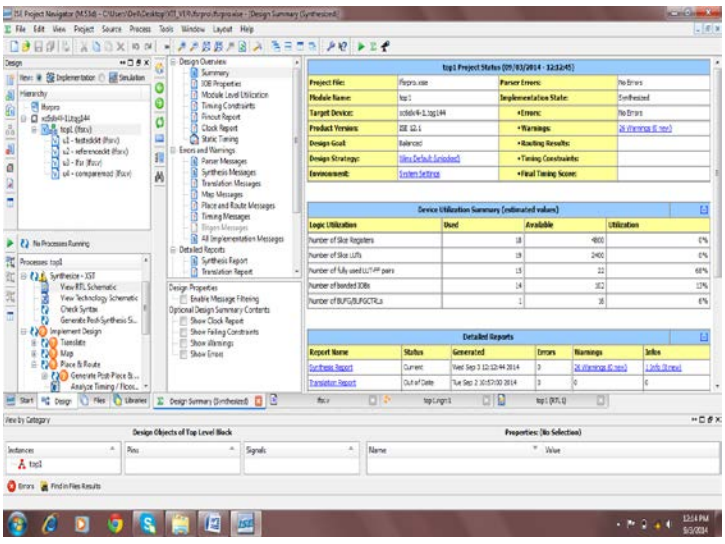

Fig.:9 Synthesis results for LT-LFSR.

TableI: SYNTHESIS AND PAR REPORT

	Existing results	Proposed result
Logic Utilization	Used(Available)	Used(Available)
Number of slice Flip-Flops	251 (3,840)	18 (4,800)
Number of 4 input LUTs	12 (3,840)	16 (2,400)
Number of occupied slices	45 (1,920)	9 (600)
Number of slices containing only related logic	45(21)	9 (20)
Number of slices containing Unrelated logic	0 (21)	7 (20)
Number of bounded IOBS	64 (173)	14 (102)
Power Consumption	14mw	12.19 mw
Fault Coverage	100 %	100 %

TableII: **POWER ANALYSIS REPORT**

Circuit	Frequency (MHZ)	Total power (mw)	Dynamic power (mw)	Quiescent power (mw)
C17	139	12.19	0.96	11.23

6 CONCLUSION

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 12.1i suite. The power reports shows that the proposed low power LFSR consumes less power (12.19 mw) 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.

7 REFERENCES

[1]M.Aubertin,*Member,IEEE,* J. M. Pierre Langlois, *Member, IEEE*, Munshin,*Member,IEEE* "A New Approach to Design Fault Coverage Circuit with Efficient Hardware Utilization for Testing Applications" IEEE transactions on very large scale integration (vlsi) systems, vol. 21, no. 5, may 2013

[2]Crouch.A,Design-for-Test for Digital IC's and Embedded Core Systems,"" Prentice Hall, 1999.

[3] Afzali.A, Alisaface.M, Atoofian.E, Hatami.S and Navabi.Z,A New Low-Power Scan-Path Architecture,"" IEEE International Symposium, Vol.5, pp.5278 - 5281, 23-26 May 2005.

[4] P. Girard, “Survey of low-power testing of VLSI circuits,” IEEE Design and Test of Computers, vol. 19, no. 3, pp. 80–90, 2002.
[5] M. ElShoukry, C. P. Ravikumar, and M. Tehranipoor, “Partial gating optimization for power reduction during test application,” in Proceedings of the 14th Asian Test Symposium (ATS '05), pp. 242– 245, ind, December 2005.
[6] S. Wang, “Generation of Low Power Dissipation and High Fault Coverage Patterns for Scan-Based BIST,” in Proc. Int. Test Conf. (ITC‟02), pp. 834-843, 2002.
[7] N. Basturkmen, S. Reddy and I. Pomeranz, “A Low power Pseudo-Random BIST Technique,” in Proc. Int. Conf. on Computer Design (ICCD‟02), pp. 468-473, 2002
[8] Mehrdad Nourani, Mohammad Tehranipoor, Nisar Ahmed,„„„Low- Transition LFSR for BIST-Based Applications,‟‟‟14th Asian Test Symposium, pp. 138- 143, 18-21 Dec. 2005.176