Implementation of Low Power BIST for 32 bit Vedic Multiplier

T.Srinivasa Rao
Department of Electronics and Communication Engineering,
Aditya Engineering College, Surampalem, Andhra Pradesh, India.

V Bharathi Devarakonda
Department of Electronics and Communication Engineering,
JNTUK, Kakinada, Andhra Pradesh, India.

Abstract - In this paper, low power built-in self test (BIST) is designed for 32 bit Vedic multiplier. The objective of this work is to reduce power consumption in BIST with increased fault coverage. Various methods of pattern generation are compared keeping in view of power consumption. In this test pattern generation the seed value is changed every 2m cycles. For this purpose m bit binary counter & gray code generator is used. Signature analysis is done with the help of Multiple Input Signature Register (MISR). The signature of MISR will indicate whether the circuit under test (CUT) i.e. Vedic multiplier is faulty or not. The results are tabulated and compared. From the implementation results, the low power BIST shows better power reduction than other methods. Simulation is carried out in Xilinx ISE and the design is implemented using Vertex 5 Field Programmable Gate Array (FPGA).

Keywords- Vedic multiplier, Test Pattern Generation, MISR, CUT

I. INTRODUCTION

The main challenging areas in VLSI are performance, cost, testing, area, reliability and power. The demand for portable computing devices and communication system are increasing rapidly. These applications require low power dissipation. The main aim of these devices is to reduce the power dissipation with high fault coverage. Generally power dissipation of a system in test mode is more than in normal mode.

Testing of integrated circuits is important to ensure high level of quality in products. The Built-In-Self-Test (BIST) is one of most popular test solutions to test the embedded cores. Test pattern generation is vital in any BIST circuit. Since off-chip communication between the FPGA and a processor is bound to be slower than on chip communication and in order to minimize the time required for adjustment of the parameters, the built in self test approach is proposed for this method.

Review of Previous Work

For Test Pattern Generation, Chakrabarty et.al, proposed a deterministic Built-in Test Pattern Generation using Twisted –Ring Counters (TRC). It embeds a precomputed deterministic test set for the circuit under test (CUT) in a short test sequence produced by TRC. The patterns derived from the seeds are applied test-per-clock to the circuit under test. This is a combination of BIST with external slow testers [1].R.S.Katti et.al, proposed a multiple output low power LFSR that produces the output of several clock cycles of a serial LFSR at once. This allows for a reduction in the power-supply voltage [3].

Sybille Hellebrand et.al, proposed pattern generation for a deterministic BIST scheme in which it targets test-per-scan architecture combining pseudo random and deterministic BIST. The amount of bits to be stored is reduced compared to others by 1-30% [9].Y.Zorian et.al, proposed a distributed BIST control scheme for complex VLSI circuits in which a generic BIST scheduling process and BIST control architecture is presented. The control architecture provides an autonomous BIST activation and a diagnostic capability to identify failed blocks [13].

II. PROPOSED ALGORITHM

BIST is a design for testability (DFT) technique in which testing is carried out using built in hardware features. Since testing is built into the hardware, it is faster and efficient. BIST techniques are aimed at overcoming the problems and limitation of external testing. Here additional circuitry is placed on the chip to facilitate testing of internal modules and hence access to internal points is easy. Further testing can be done at the normal operating speed. With advances in integration the costs of putting extra circuitry on chip is decreasing, making BIST an attractive and feasible alternative to external testing. At system level, BIST is a low cost test solution.

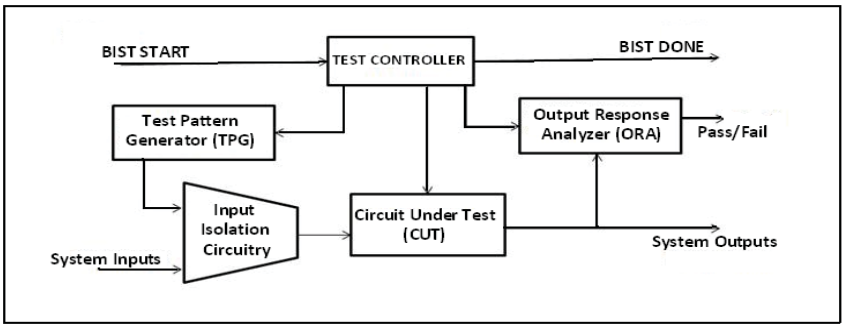


Figure1: BIST basic block diagram

The generic BIST is shown in Fig. 1. BIST solution consists of several blocks given below.

a) Circuit under test (CUT): It is the portion of the circuit tested in BIST mode. It can be combinational, sequential or a memory.

b) Test pattern Generator (TPG): This is a circuit to be tested, a way to compress those results & way to analyze them. It generates the test patterns for CUT. Here a Linear feedback shift register is used to generate patterns. Patterns are generated in pseudo random fashion.

c) Test controller: It controls the test execution. It provides the control signal to activate all blocks. If control signal is 0, then BIST is said to be in test mode & if 1, normal mode.

Design of LFSR

BIST architecture is based on linear feedback shift register whose input bit is logic function of its previous state. An LFSR basically consists of an interconnection of D-flip-flops, XOR gates, forming a shift register with feedback. Mainly LFSRs are used for pseudo random generation such as TPGs, ATPG, code convolution techniques. The initial value of LFSR is called seed value. This seed value is always represented in Galois field format or normal binary format also called characteristic polynomial expression of a unit. The initial value of LFSR should be non zero value i.e. any one of the bit should be high. If it is zero value then LFSR will be in zero lock state where it produces only zero value to all the clock pulses. The selection of characteristic polynomial is based on the number of faults to be covered.

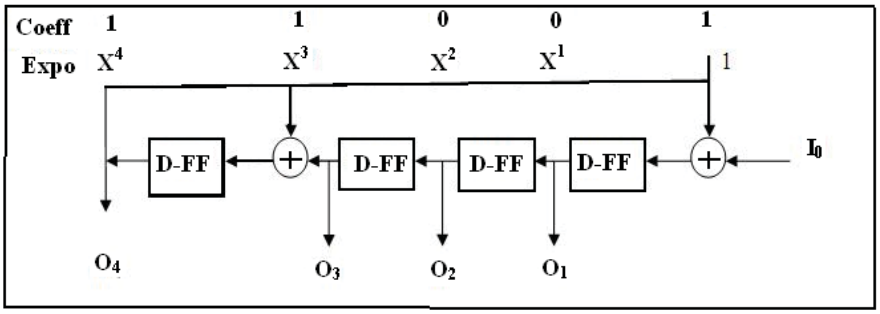


Figure 2 The division type LFSR with polynomial x^4+x^3+1

III. Implementation Of Different Test Pattern Generators

Before the overall design is synthesized, the four LFSR based test pattern generations is incorporated into CUT. There are different test pattern generators as follows.

A. LFSR Type I
B. LFSR Type II
C. Multiple polynomial
D. Cellular Automaton LFSR

All these circuits are described in Verilog HDL and implemented on Vertex 5 FPGA.

A.LFSR Type I:

LFSR type I is commonly used TPG and is called „External LFSR". It consists of D flip flops and XOR gates. The XOR gates are in external feedback. LFSR can cycle through 2^{n-1} distinct states, all zero state is omitted. LFSR type I is shown in Fig 3.

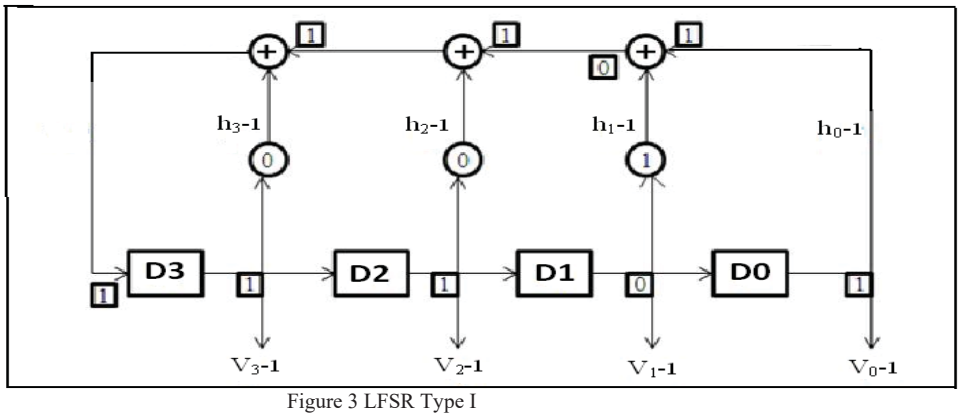


Figure 3 LFSR Type I

B.LFSR Type II:

LFSR type II is called „Internal LFSR". It has the linear elements interspersed between flip flops. There are n flip-flops, so it is called an n-stage LFSR. LFSR can cycle through 2^{n-1} distinct states, all zero state is omitted. The main difference between LFSR type I & type II is that in type I the XOR gates are in external feedback and in type II the XOR gates are internal i.e. in between the flip-flops.

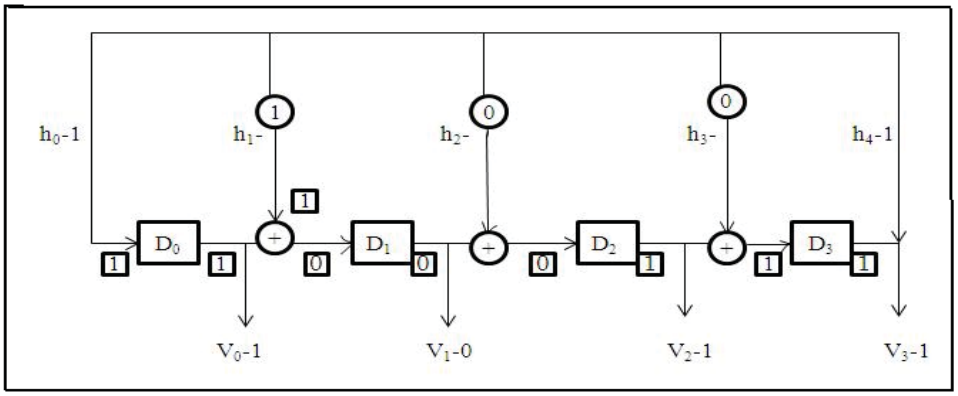


Figure 4: LFSR Type II

C. Multiple Polynomial LFSR:

Multiple polynomial LFSR can change characteristic polynomial in determined time. Here a decoding logic is present and its input bits are given to three AND gates. Using this different seed values are generated. This decoding logic is used to select the particular pattern in a row of patterns. Circuit for Multiple polynomial is shown in Fig 5.

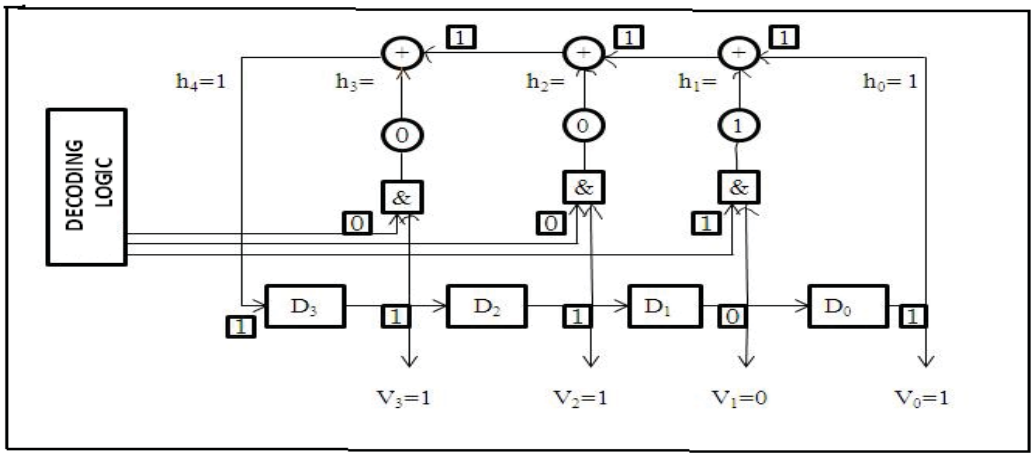


Figure 5 Multiple polynomial LFSR

D. Cellular Automation LFSR:

Cellular Automation (CA) consists of cells. Each cell is build from memory element (Flip-flop) and combinational element. Input to the combinational part of cell is driven from neighboring cells and the cell itself. It will generate all 2^{n-1} patterns for an n cell CA. The all zero pattern is not generated by the CA, just like an LFSR, without adding additional hardware. CA is used in the pattern generator and response analyzer. LFSRs are more popular because of their compact and simple design. Cellular Automaton LFSR is more complex to design but provide patterns with higher randomness.

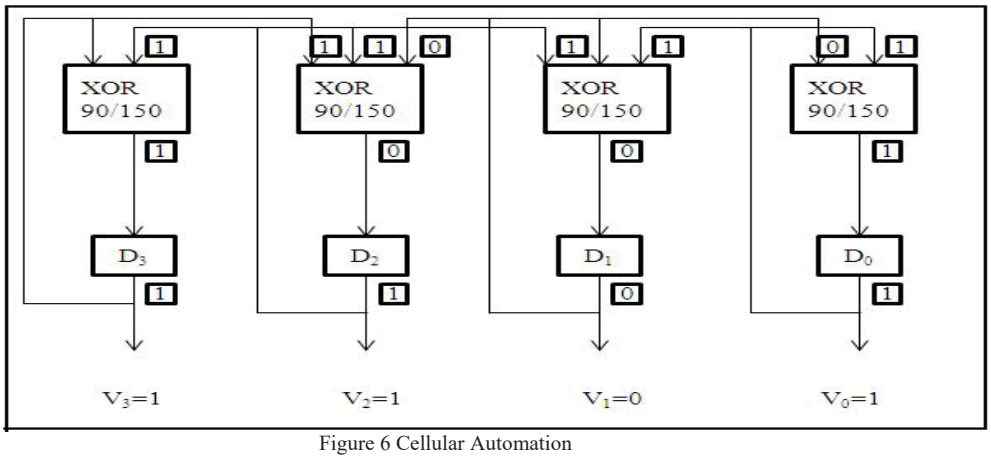


Figure 6 Cellular Automation

IV LATEST METHOD

In this section BIST is designed using the low power Test Pattern Generator as shown in the BIST block diagram in Fig 1. Several blocks i.e. Circuit Under test, Response Analyzer are been discussed.

A. Low power Test Pattern Generator:

Here a Linear feedback shift register is used for generating test patterns with reduced switching activities. The LP-TPG consists of m bit counter, gray code generator, LP-LFSR, NOR gate structure and XOR gate as shown in Fig. 7. The m bit counter is initialized with zeros, which generates 2^m test patterns. Counter and gray code generator are synchronized with common clock.

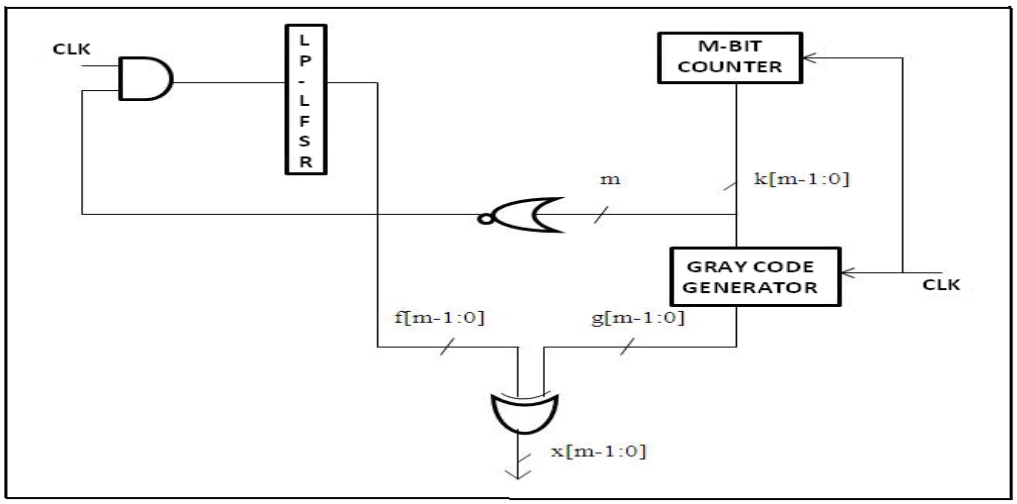


Figure 7 Low Power Test Pattern Generator

When counter output is all zero pattern, NOR gate output is one. Only when the NOR gate output is one, the clock signal is applied to activate the LP-LFSR to generate the next seed. This seed and the output sequence

from the gray code generator are exclusive ORed to generate the final output. This effectively reduces the switching activities which results in low power. For every 2^m clock cycles the seed value is changed and here no decoding logic is used. Also the selection of polynomial depends on number of faults to be covered which is not the case for other existing methods. So fault coverage is high and high randomness is introduced.

B. Circuit under test:

This work mainly evaluates the speed and power of Vedic Multiplier with latest type of BIST and compares with those built with other types of BIST. In this paper a 16*16 Vedic multiplier is designed. Adders are used to perform the addition of bits and generate the final output.

AlgorithmforVedicmultiplier:

The algorithm for N*N bit Vedic multiplier is given below [6]:

Consider for any number of bits in input ,let the multiplication of two N- bit binary numbers (where N=1,2,3…N) A and B where A=A_N…A_3,A_2,$A_1$and B=B_N…B_3,B_2,B_1 The final multiplication result will be N+N bits as S=$S_{(N+N)}$…S_3,S_2,S_1.

Step1: Divide the multiplicand A and multiplier B into two equal parts each consisting of [N-to-(N/2) +1] bits and [N/2to1] bits, respectively where first part represents MSB and other is LSB.

Step2: Now represent parts of A as A_M and A_L So that A = {$A_M A_L$}. B as B_M and B_L So that B = {$B_M B_L$}.

Step3: Using the Vedic multiplication fundamentals, taking 4bits at a time and 4multiplier blocks, multiplication operation is performed.

Step4: The outputs of [(N/2)*(N/2)] bits are added (i.e. 8*8bitsfora16bitmultiplier) accordingly to obtain the final output .For a 16*16multiplier, 3ripple carry adders are used.

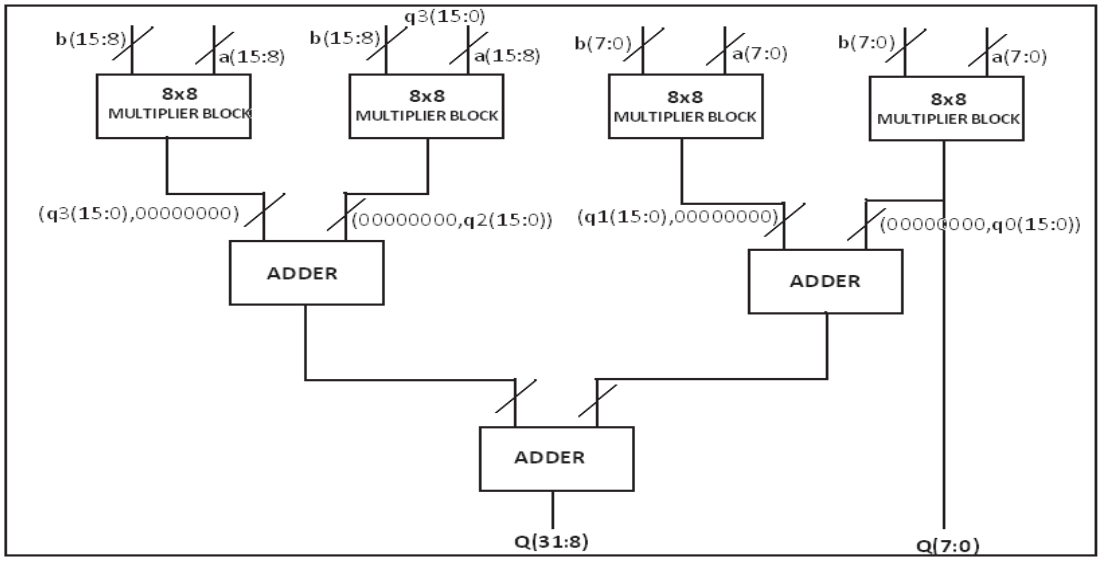


Figure8:16*16Vedic Multiplier

Response Analyzer:

A response analyzer is a comparator with stored responses or an LFSR used as signature analyzer. It analyses the value sequence on primary output and compares it with expected output. Signature analysis is done with the help of multiple input signature register (MISR).Here we assume that the CUT has no outputs .It is seen that this circuit operates as n single-input signature analyzers. An n-stage MISR has the property that the parity overall the bits in the input streams equals the parity of the final signature.After16clock cycles the seed value is changed. The signature of MISR will indicate whether the circuit under test i.e. Vedic multipliers faulty or not. The polynomial used in the BIST is given as

$P^*(x)=x^{31}+x^{29}+x^{28}+x^{20}+x^{18}+x^{16}+x^{13}+x^9+x^7+x^4+x^3+x^2$&G(x)=0000000000000101.

The input sequence can be represented by the polynomial G(x) and the output sequence by Q(x). The highest degree of polynomials G(x) and Q(x) correspond, respectively, to the first input. Bit to enter the LFSR and the first output bit produced n clock periods later, and where n is the degree of the LFSR. If the initial state of the LFSR is

all)s, let the final state of the LFSR be represented by the polynomial R(x). The n bit can be shown that these polynomials are related by the equation [5]

$$G(x)/P^*(x) = Q(x) + \{R(x)/P^*(x)\}$$

Where $P^*(x)$ is the reciprocal characteristic polynomial of the LFSR. Hence an LFSR carries out (polynomial) division on the input stream by the characteristic polynomial, producing an output stream corresponding to the quotient Q(x) and a remainder R(x) .Here now Fig9(a) shows the 32 bit multiple input signature register $P^*(x)=x^{31}+x^{29}+x^{28}+x^{20}+x^{18}+x^{16}+x^{13}+x^9+x^7+x^4+x^3+x^2$ and G(x)istakenas0000000000000101(16bits) which is the input sequence given. Fig 9(b) shows a simulation of the processing of this input by the LFSR.

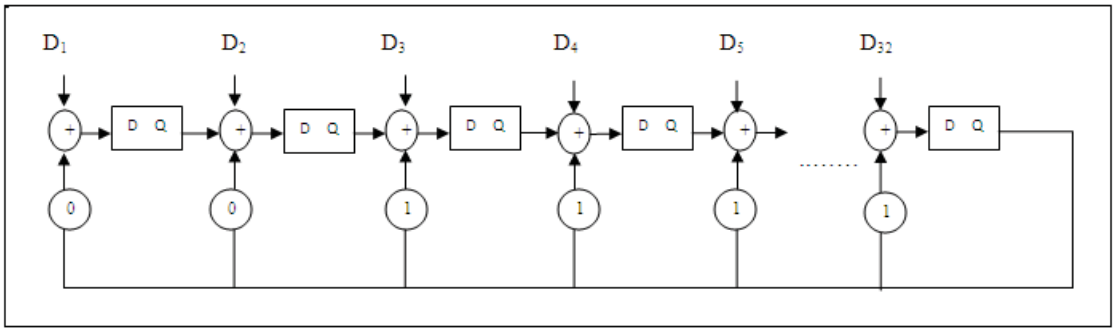


Figure9 (a) Multiple Input Signature Register

Time	Input Stream	Register contents	Output Stream
		1 2 3 4 5	
0	1010000000000000	0 0 0 0 0 ⟸	Initial state
1	101000000000000	0 0 0 0 0	
2	10100000000000	0 0 0 0 0	
3	1010000000000	0 0 0 0 0	
4	101000000000	0 0 0 0 0	
5	10100000000	0 0 0 0 0	
6	1010000000	0 0 0 0 0	0
7	101000000	0 0 0 0 0	00
8	10100000	0 0 0 0 0	000
.	.	.	.
.	.	.	.
.	.	.	.
14	10	1 0 0 0 0	000000000
15	1	0 1 0 0 0	0000000000
16		1 0 1 0 0	00000000000
		Remainder R(x)	Quotient Q(x)

Figure9 (b) Polynomial Division

Remainder R(x) =$1+x^2$ and Q(x) =0. To check this result we have performed this polynomial division in the Matlab using the function 'deconv'.As it is shown in Fig 9(c), 'a' and 'b' are inputs where 'a' is the signature and 'b'

is the polynomial. Outputs are ‘q’ and ’r’.’ ‘q’ is the quotient and ‘r’ is the remainder.

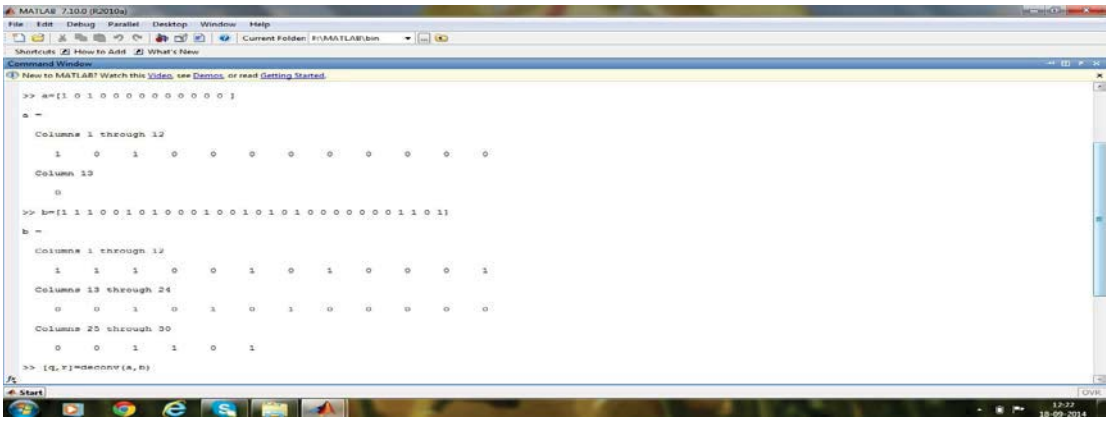


Figure9(c): Polynomial division in Matlab

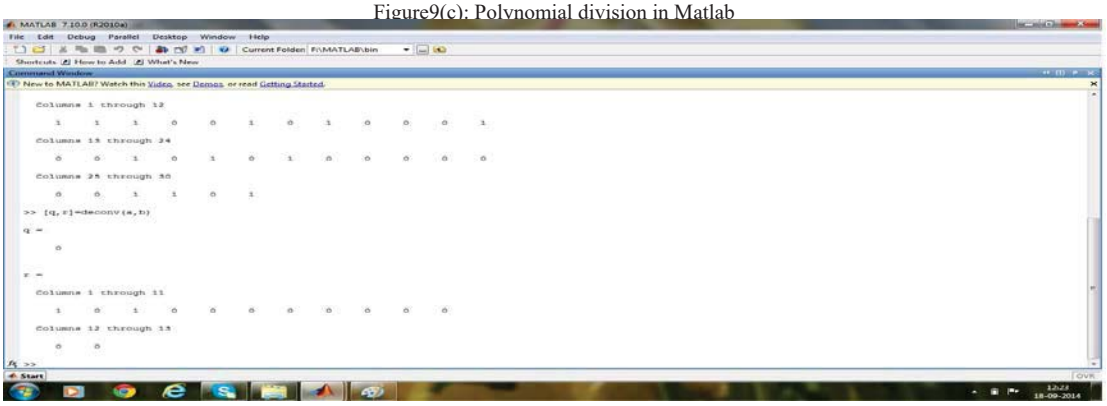


Figure9 (d): Polynomial division in Matlab.

So we can see from the Fig9 (d) that theremainderis1+X^2.

V IMPLEMENTATION DETAILS&RESULTS

Simulation and analysis were carriedoutwithXilinx13.2version.XilinxplanAheadtoolwas used for the power analysis. The output patterns for different LFSR’s are shown below:

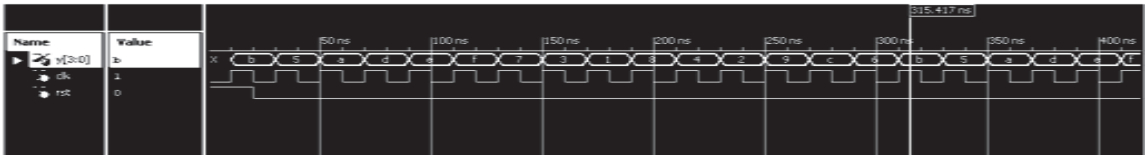


Figure10: Waveforms of LFSR Type-I

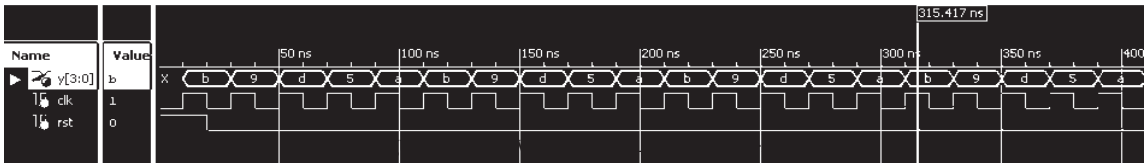


Figure11: Waveforms of LFSR Type-II

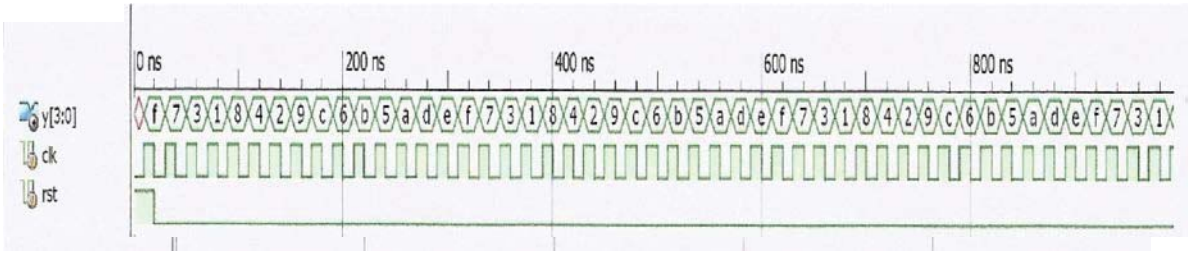


Figure12: Waveforms of Multiple polynomial LFSR

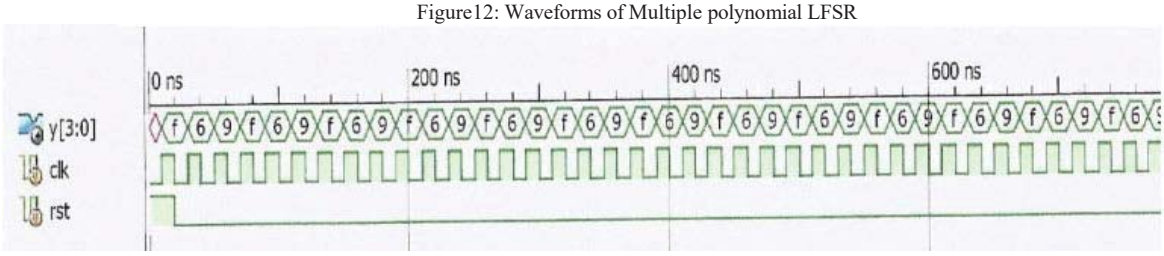


Figure13: Waveforms of Cellular Automation LFSR.

In the entire above four waveforms clock (clk) and reset (rst) are given and y is the output waveform which generates different states.

The waveforms for 32 bit Vedic Multiplier is shown in Fig14.In the figure **a** and **b** are the waveforms for inputs of 16 bit each and **c** is the output of 32bit.

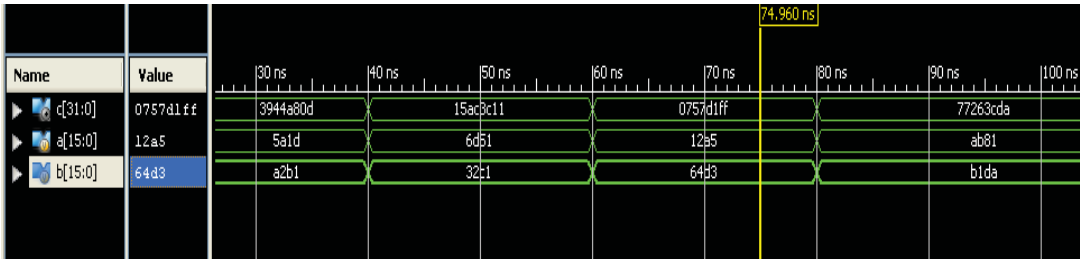


Figure14: Waveforms of Vedic multiplier

The latest method output is shown in Fig.15 where the clock frequency used for this implementation is 34.8MHz.'m'is the waveform for the output bits and 'si' is the signature which indicates whether the chip is pass or fail. If,, 'si' is all zero then chip is fail and if 'si' is 0000000000000101 then the chip is pass. As we can see from waveform chip is pass.

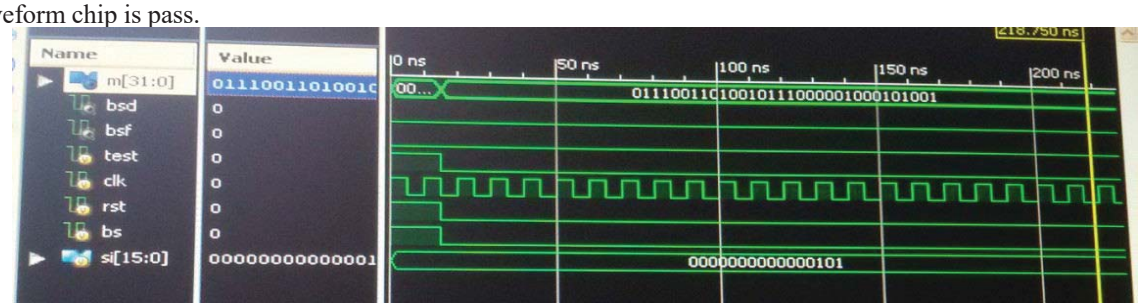


Figure15Waveformsof Latest Method

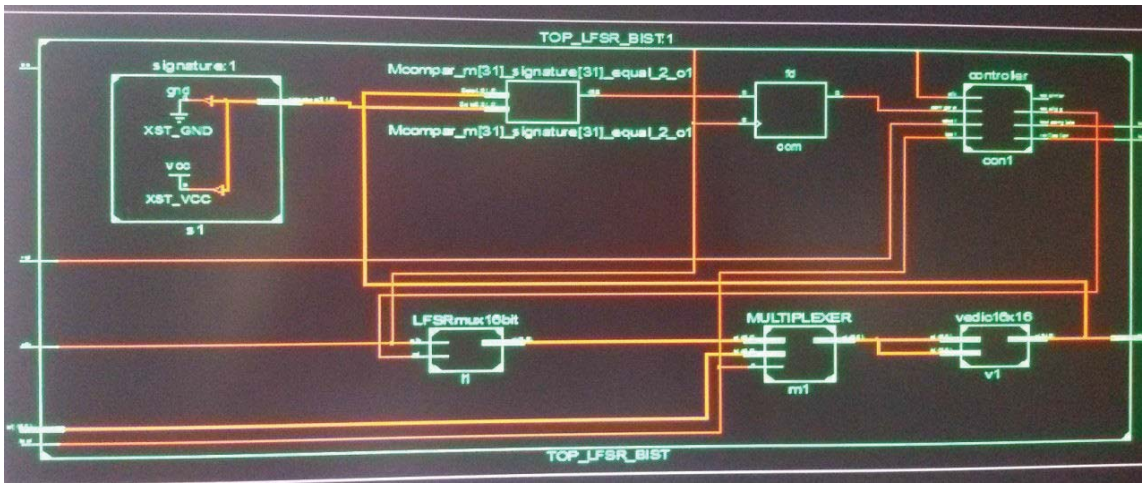


Figure16: RTL Schematic of Latest Method

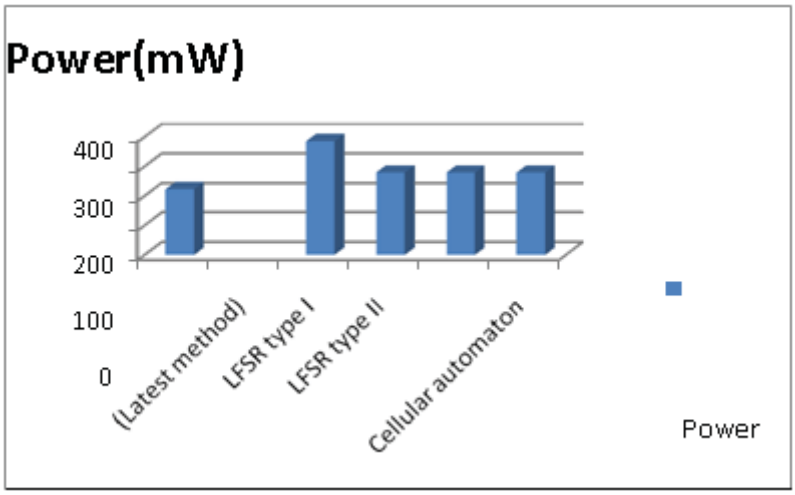
Table 1: Estimationof different values of latest method

PARAMETER	VALUE
Device logic	0.07mw
Clock	2mw
I/O power	1.22mw
Total power	222mw

Table 2: The comparison of LFSR techniques based on the several parameters is given below

	LFSR MUX (Latest method)	LFSR type-I	LFSR type-II	Multiple-polynomial	Cellular automaton
Power	222mW	384mW	278mw	278mw	278mw
Switching activities	17	31	23	34	42
Fault Coverage (for16cycles)	16*P	P	P	P	P

Where P is the probable faults that can be found in 16 cycles with same starting seed. The comparison of LFSR techniques based on the power consumption is given above. From the table1 LFSR mux is the latest method. It can be observed that the power consumption for the latest method is 222mW. When compared with the other existing methods, the power consumption for the latest method is much reduced. Because of less switching activity the fault coverage is also higher than other methods.



VI CONCLUSIONS

In this paper a low power Test Pattern Generator has been incorporated in BIST developed for Vedic multiplier. The switching activities are reduced in the test pattern generation. Fault coverage is increased by the maximum number of clock cycles of the binary counter. The power consumption of different test pattern generation techniques has been found out and compared with the latest method. BIST is implemented for low power test pattern generator i.e., Vedic multiplier in the latest method. It is observed that the power consumption is reduced along with increased fault coverage when compared to other implementations.

REFERENCES

[1] Chakrabarty, et.al, "Deterministic Built-in Test Pattern Generation for High Performance Circuits Using Twisted-Ring Counters,"IEEETrans.ofVLSIsystem, Vol.8, No.5, pp.633-636, Oct 2000.
[2] HakmiA.W, "Programmable deterministic built-in self test", IEEE international Test Conference OAI (ITC), Nov2007.
[3] KattiR.S, RuanX.Y, andKhattri.H, "Multiple Output Low-Power Linear Feedback Shift Register Design," IEEE Transaction on. Circuits & Systems.I, Vol.53, No.7, pp-1487-1495, July2006.
[4] MechrdadNourani, "Low-transitiontestpatterngenerationforBIST-basedapplications", IEEE transactions on Computers, Vol57, No.3, 2008.
[5] Miron Abramovici, "Design for testability", revised edition, Nov1997.
[6] PoornimaM, "Implementation of multiplier using Vedic algorithm", International Journal of Innovative Technology & Exploring Engineering (IJITEE) Vol-2, issue-6, May2013.
[7] RothC.H "Digital System Design Using VHDL"PWS publishing Company, 1998.
[8] SamirPalnithkar, "Aguidetodigital design&synthesis", 2ndedition.
[9] Sybille Hellebrand, "Pattern Generation for a Deterministic BIST Scheme", ACM IEEE on CAD95 (ICCAD-95), SanJose, Nov1995.
[10] Voyiatzis.I, Paschalis. A, Nikolas', and Halatsis.C, "An efficient built-in-self test method for robust path delay fault testing, Journal of electronics testing: Theory and applicationsVol.8, No.2pp.219-222, 1996.
[11] Wang.S& GuptaS.K, "DS BIST TPG for Low Switching Activity", IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, Vol.21, No.7, pp.842-851, 2002.
[12] Zorian.Y, "ADistributedBISTcontrolSchemeforcomplexVLSIdevices,"Proc.VLSItestSymp, pp.4-s9, 1993.