International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 05 | May 2020 www.irjet.net p-ISSN: 2395-0072

Design and Implementation of Low Power Test Pattern Generator

Sumanth Sajjanar[1], Suraj Ullur[2], Mr. P Narashimaraja[3]

[1, 2]*Department of Electronics and Communication Engineering, RV College of Engineering, Bengaluru, Karnataka, India*

[3]*Assistant Professor, Department of Electronics and Communication Engineering, RV College of Engineering, Bengaluru, Karnataka, India*

---***---

Abstract - *This paper discusses about low power test pattern generator used for testing circuits. The testing of VLSI circuits entitles many challenges in terms of area overhead, power and latency. The Low power test pattern generation is a very crucial technique for testing of a complex architecture of VLSI design. As technology progresses, the growing demands of long life batteries in battery operated devices have set ways for new ideas that reduce the power consumed in these devices. As we know that during testing when the device's normal functioning mode is off, the dissipation of power is approximately 200% more than that of normal functioning mode. So a method is proposed to minimize the concerned power at testing mode itself in the very beginning. It is a very important step needed while testing a circuit. Over here, LFSR is used for generating test patterns. And to this LFSR, two algorithms will be implemented. One is by dividing a circuit into two parts and the other way is to using a clock divider circuit. After implementing an efficient LFSR along with the two algorithms, we are able to reduce the average power of LFSR from 262.3nW to 247.7 nW (Algorithm 1) and 235.9 nW (Algorithm 2 .*

Key Words: Test Pattern Generator, LFSR, seed, switching activity, D Flip Flop, Multiplexers

1. INTRODUCTION

With the increase in the availability and the use of portable device, the demand of the number of batteries as well as the long life of battery has reached its pinnacle. Apart from this, with the shrinking size of transistor, the complexity of the chips has also increased. The rise in the complexity has given way to the rise in power consumption. The basic principle behind testing methodology is as follows: These testing are being done keeping in mind the aim to minimize the stuck at faults. A circuit is most likely to be effected by stuck at faults. These test are being designed so as to check every possible stuck at fault that can be present in the circuit.

Each of the early approaches that had been adopted to achieve, these two objectives i.e. to minimize power consumption as well as to generate all set of patterns to check every possible fault, had not succeeded in achieving both The logic proposed in this paper is able to generate all sets of test patterns utilizing minimum possible gates. Together with this the idea of low power consumption has also been obtained to a great extent. The generated test patterns have hamming distance of one.Since all set of patterns are obtained, all possible stuck at faults will be covered by them. Depending upon the initial seed, a pattern is being fed so as to give way to the other following patterns. The design can be reset at any moment by assigning '1' to the reset pin.

As it is known that structural test approach is normally being adopted so as to ensure credibility of testing, not all test patterns are required to test any single gate. So by this argument it can be concluded that it is useless to generate all set of patterns when the number of patterns required is limited. But, since every system is comprised of a number of gates and that too of multiple types, and all patterns that are able to test a single gate do not guaranteed to test all other gates, therefore it is required to generate all set of patterns so as to ensure proper and on-time structural testing of all types of gates present in the circuit. The proposed design is said to be an amalgamation of two designs i.e. standard LFSR and modified clock scheme module but actually it's not so. The standard The code generator ensures minimum switching activity whereas modified clock ensures minimization of use of clock.

The test vectors are generated by the random sequence test pattern which also known as LFSR. The LFSR is the sequential logic circuits used to create pseudorandom binary sequences (PRBSs). An LFSR circuit consists of a set of M registers and feedback taps that establish the sequence of states that the LFSR transitions through. The feedback taps are described by a modulo-2 polynomial. A primitive polynomial generates a maximal length of m-sequence, where the LFSR transitions through the 2m-1 state before repeat sequences, there is a single unused LFSR state. The PRBS is the binary output of the LFSR. The random binary sequence is described as pseudorandom, as the sequence is continuous in nature and that results from the correlation properties of a random sequence.

2. BASIC ARCHITECTURE OF BIST

VLSI testing, only from the context where the circuit needs to be put to a "test mode" for validating that it is free of faults. Circuits tested OK are shipped to the customers with the assumption that they would not fail within their expected life time; this is called off-line testing. However, this assumption does not hold for modern day ICs, based on deep sub- micron technology, because they may develop failures even during operation within expected life time. To cater to this problem sometimes redundant circuitry are kept on-chip which replace the faulty parts. Testing a circuit every time before they startup, is called Built-In- Self-Test (BIST). It's architecture is as shown in Fig 1.

Hardware Test Pattern Generator:

This module generates the test patterns required to sensitize the faults and propagate the effect to the outputs. As the test pattern generator is a circuit (not equipment) its area is limited. So storing and then generating test patterns obtained by ATPG algorithms on the CUT using the hardware test pattern generator is not feasible. Instead, the test pattern generator is basically a type of register which generates random patterns which act as test patterns. The main emphasis of the register design is to have low area yet generate as many different patterns (from 0 to 2n-1, if there are n flip-flops in the register) as possible.

Input Mux:

This multiplexer is to allow normal inputs to the circuit when it is operational and test inputs from the pattern generator when BIST is executed. The control input of the multiplexer is fed by a central test controller.

Output response compactor:

Output response compacter performs lossy compression of the outputs of the CUT. The output of the CUT is to be compared with the expected response (called golden signature). Similar to the situation for test pattern generator, expected output responses cannot be stored explicitly in a memory and compared with the responses of the CUT. So CUT response needs to be compacted such that comparisons with expected responses (golden signatures) become simpler in terms of area of the memory that stores the golden signatures.

ROM:

Stores golden signature that needs to be compared with the compacted CUT response.

Comparator:

Hardware to compare compacted CUT response and golden signature (from ROM).

Test Controller:

Circuit to control the BIST. Whenever an IC is powered up (signal start BIST is made active) the test controller starts the BIST procedure. Once the test is over, the status line is made high if fault is found. Following that, the controller connects normal inputs to the CUT via the multiplexer, thus making it ready for operation

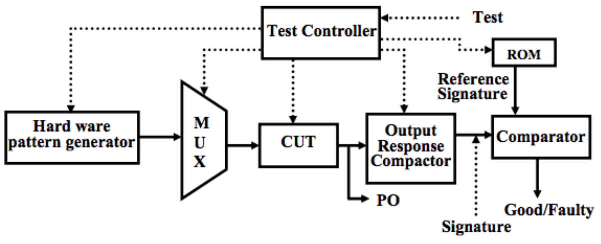

Fig -1: Block Diagram of BIST

Hardware pattern generator (LFSR)

There are two main targets for the hardware pattern generator:
(i) Low area.
(ii) Pseudo-exhaustive pattern generation (i.e., generate as many different patterns from 0 to 2 1 n as possible, if there are n flip-flops in the register).

Linear feedback shift register (LFSR) pattern generator is most commonly used for test pattern generation in BIST because it satisfies the above two conditions. LFSR is as shown in in Fig 2. There are basically two types of LFSRs,
(i) Standard LFSR.
(ii) Modular LFSR.

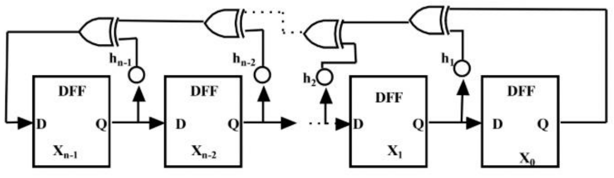

Fig -2: LFSR

3. THE ALGORITHMS

Here, we focus on low power LFSR based test pattern generator that can be used for testing of both combinatorial and sequential circuits. The proposed architecture step-up the correlation between the two tests vectors which reduces the number of transition i.e. switching activities between two test vectors. Minimizing the switching activity between test vectors will result in reducing the power consumption. The conventional LFSR architecture is to be customized in such a way that it routinely injects intermediate patterns between its unique pair of patterns. This can be done by using two scheme i.e. Bipartite and random injection, which is further discussed in this section and with a minimal number of switching activity between two test vectors. We propose a Low Power Test Pattern Generator that introduces two techniques for test vectors generation called Random Injection (RI) and Bipartite LFSR. In brief, the RI technique injects a new pattern between two successive test patterns by using a random-bit injection (R) whether it can be either ‘0’ or ‘1’, in the consequent bit of an intermediate pattern where there is transition occurs in the corresponding bit of pattern pairs. The other algorithm we used is modified clock scheme as shown in Fig 5.

Bipartite LFSR Technique

The realization of an LFSR can be altered to develop some design performance, such as dissipated power, while testing. Nevertheless, such an alteration may change the sequence of patterns or insertion of new patterns will have an effect on the overall randomness. This technique inserts an intermediate test pattern between two consecutive random patterns such that the number of transitions between them is reduced. In this scheme, an LFSR is separated into two equal half by applying two non-overlapping Clk (clock) signals. In other sense, when one half of LFSR is working, the other half is in inactive mode. An LFSR consists of number of flip-flops connected in series with clock signal. The dummy flip-flop adds to the Bipartite LFSR architecture to store the n/2th bit of LFSR when clk1clk2 = 10 and send this value into the (n/2+1) flip-flop when the second half becomes working (clk1clk2= 01).

LFSR is successfully divided into two equal half, and the sheltered flip- flop is a merger between these divisions. An n-bit LFSR is alienated into two n/2-bit LFSRs, which collectively diminish the testing and clock tree power utilization. The limitation of this technique which cuts down the randomness functionality of the LFSR due to separation of it into two LFSR and it also wants to generate and distribute two non-overlapping clocks signals (with half frequency), which raise the area overhead. The LP-LFSR structure is as shown in Fig 3.

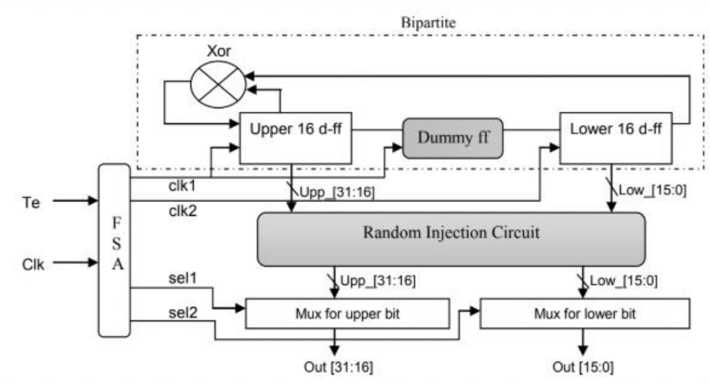

Fig -3: LP-LFSR structure

1. During 1st clock cycle, clk1clk2=11. The upper half of bipartite LFSR is in active whereas the lower half is in inactive condition respectively. With sel1sel2 =11, both half’s of bipartite LFSR are sent to the outputs (Out). In this case, say Ti is generated.

2. During 2nd clock cycle, clk1clk2 =00 and sel1sel2 =10. Both half’s of bipartite LFSR are in inactive mode. The Upper half of bipartite LFSR is sent to the outputs and the RI injector circuit outputs are sent to the outputs. In this step, T i1 is generated.

3. During 3rd clock cycle, clk1clk2 = 01 and sel1sel2 = 11. The lower half of bipartite LFSR is in working condition whereas upper half of bipartite LFSR is in inactive mode. The Bipartite LFSR output is latched to the output. In this, Ti2 is generated.

4. During 4th clock cycle, clk1clk2 =00 and sel1sel2 =01. Both half’s of bipartite LFSR are in inactive mode. With sel1sel2 =01, the injector outputs are sent to the outputs and the lower half of bipartite LFSR to the outputs (Out [15:0]). This generates Ti3.

5. During 5th clock cycle process continues by going through step 1 to generate T i+1. The dummy flip-flop is used to store the states of last bit generated by the upper flip-flop in the clock cycle. The XORing operation is used to generate the different pattern. To generate the non-overlapping enable signal and

select line of Mux required 4-cycle of clock. Select signal 1 is used to select the Mux for upper bit and select signal 2 for lower-bit Mux respectively.

Modified Clock Scheme

The proposed design shown in Fig 4 comprised of a modified clock. The modified clock will minimize the use of clock. Since the pattern generator used is one of the sequential circuits designed so far, each flip flop is clocked sequentially. A normal clock remains active throughout the process but a modified clock will be active only for that flip flop of the design where any event is taking place i.e. either logic 0 is changed to logic 1 or vice versa

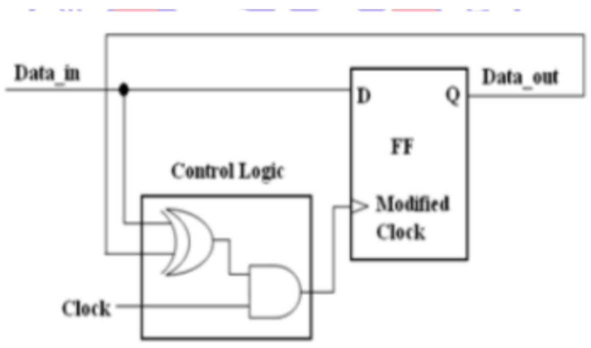

Fig -4: Modified Clock Scheme

4. IMPLEMENTATION AND RESULTS

The tool we are using to analyse our results is Cadence Virtuoso.

The Cadence® Virtuoso® Schematic Editor provides numerous capabilities to facilitate fast and easy design entry, including design assistants that speed common tasks by as much as 5X. Well-defined component libraries allow faster design at both the gate and transistor levels. Sophisticated wire-routing capabilities further assist in connecting devices. For larger and more complex designs, the Virtuoso Schematic Editor not only supports multi-sheet designs but also provides the ability to design hierarchically, with no limit on the number of levels used. The Hierarchy Editor makes hierarchical designs easy to traverse, and automatically ensures all connections are maintained accurately throughout the design.

We have chosen the most efficient flip-flop by analysing the different flip-flops available in terms of area and power consumption i.e D flip flop. We have implemented LFSR, observed the outputs and have plotted the power along with the output obtained.

We have implemented two algorithms to reduce power consumption and have observed the outputs. We have also plotted the power along with the outputs obtained. Given below are the basic blocks we have used in the design. Fig 5 shows the D flip flop used. Fig 6 shows the 2:1 multiplexer used. Fig 7 shows the MUX combinational logic used. Fig 8 shows the control logic used for the modified clock scheme.

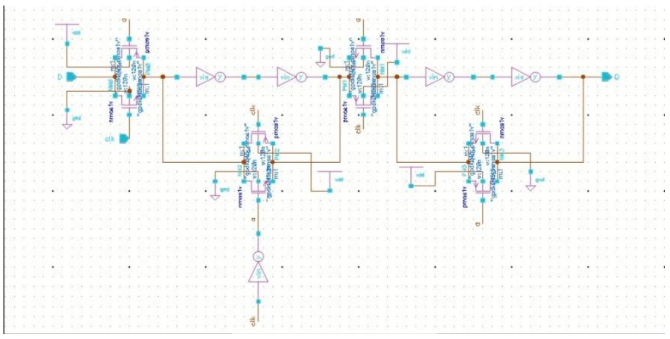

Fig -5: D Flip Flop

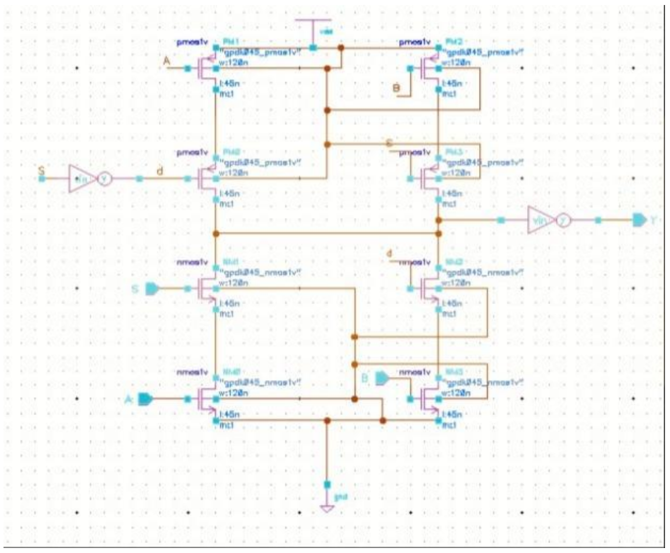

Fig -6: 2:1 MUX

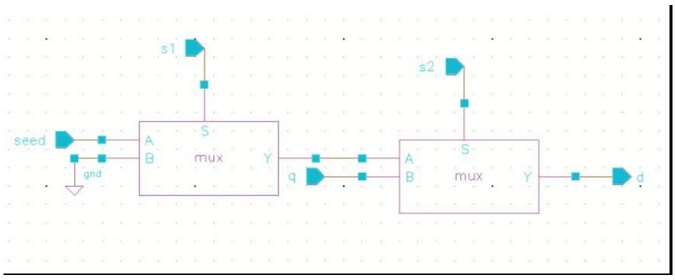

Fig -7: MUX Combinational Logic

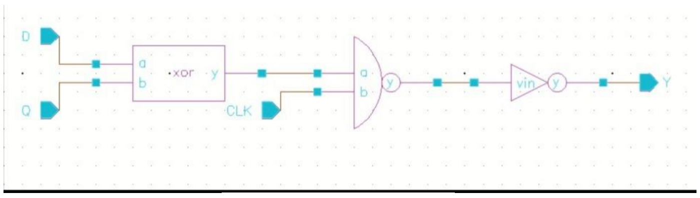

Fig -8: Control Logic

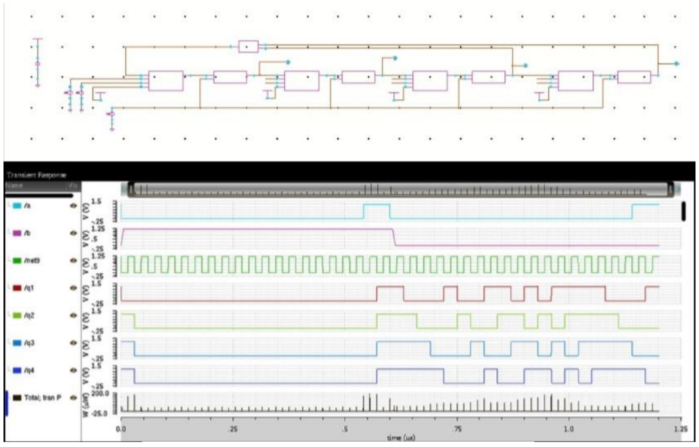

Fig -9: Output waveforms of LFSR

From Fig 9 circuit, we obtained the waveforms. We were able to generate 15 different patterns. The instantaneous power graph is right at the bottom and average power is calculated for it.

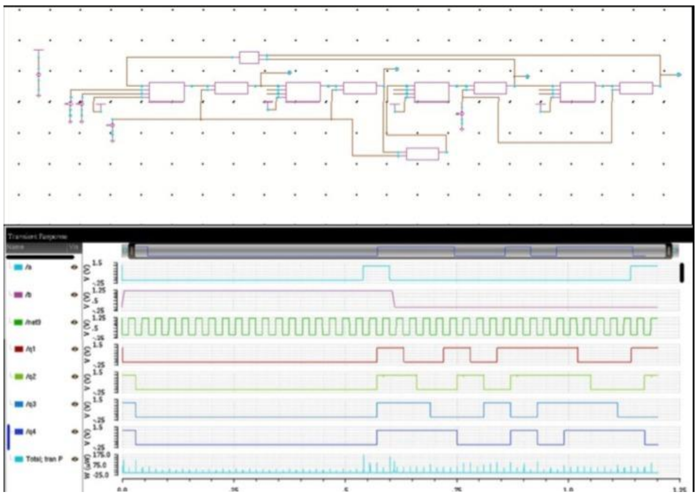

Fig -10: Output waveforms of Algorithm 1

From Fig 10 circuit, we obtained the waveforms. We were able to generate 15 different patterns. We implemented algorithm 1 for this. The instantaneous power graph is right at the bottom and average power is calculated for it. Power obtained here is less than what was obtained for the LFSR as a whole.

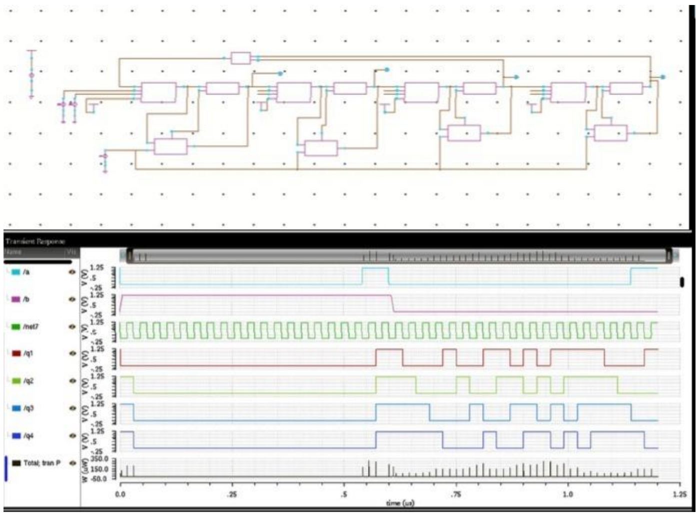

Fig -11: Output waveforms of Algorithm 2

From Fig 11 circuit, we obtained the waveforms. We implemented algorithm 2 for this. We were able to generate patterns but less in number when compared to the others. The instantaneous power graph is right at the bottom and average power is calculated for it. Power obtained is way less than the others.

Table -1: Quantified results for the LFSR, Algorithm1 and Algorithm 2

	LFSR	Algorithm 1	Algorithm 2
Average Power	262.3nW	247.7nW	235.9nW
Peak Power	190.7uW	177.2uW	168.3 uW

CONCLUSION

After implementing an efficient LFSR along with the two algorithms, we are able to reduce the average power of LFSR from 262.3nW to 247.7 nW (Algorithm 1) and 235.9 nW (Algorithm 2). The results can be seen as shown in Table 1.

REFERENCES

[1] Oscar Acevedo, Dimitri Kagaris, "On The Computation of LFSR Characteristic Polynomials for Built-in Deterministic Test Pattern Generation" IEEE TRANSACTIONS ON COMPUTERS, 2016

[2] Irith Pomeranz, "Extra Clocking of LFSR Seeds for Improved Path Delay Fault Coverage" IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, 2019

[3] D.Sunitha, Naga Raju Ravada, G.Leenendra Chowdary, "Design of Low Power Test Pattern Generator" International Journal of Scientific Engineering and Technology Research Volume.03, IssueNo.08, May-2014, Pages: 1304-1308.

[4] R.Ramalakshmi, S.Bibiana Vincy, "DESIGN AND ANALYSIS OF LOW POWER TEST PATTERN GENERATOR USING D FLIP-FLOP" International Conference on Engineering Trends and Science & Humanities (ICETSH-2015) ISSN: 2348 – 8549.

[5] BHARTI MORYANI, D.K.MISHRA,"LOW POWER TEST PATTERN GENERATOR WITH MODIFIED CLOCK FOR BIST" Proceeding International conference on Recent Innovations is Signal Processing and Embedded Systems (RISE -2017) 27-29 October,2017.

[6] Tejas Thubrikar, Sandeep Kakde, Shweta Gaidhani, Shailesh Kamble and Nikit Shah,"Design and Implementation of Low Power Test Pattern Generator Using Low Transitions LFSR" International Conference on Communication and Signal Processing, April 6-8, 2017, India.