ISSN: 2321-2152

IJMECE

International Journal of modern electronics and communication engineering

E-Mail
editor.ijmece@gmail.com
editor@ijmece.com

www.ijmece.com

ISSN2321-2152
www.ijmece .com
Vol 12, Issue.1 March 2024

BIT-SWAPPING LFSR AND SCAN-CHAIN ORDERING: A NOVEL TECHNIQUE FOR PEAK- AND AVERAGE-POWER REDUCTION IN SCAN-BASED BIST

MUTHUKURI AKHILA*1, Mr. M. MADHU BABU*2, KATHI LEKHANA*3, SOMINENI RAMYA*4, RAYALLA BALA NARASIMHULU*5

ABSTRACT

The design for low power has become one of the greatest challenges in high-performance very large scale integration (VLSI) design. It has been found that the power consumed during test mode operation is often much higher than during normal mode operation. This is because most of the consumed power results from the switching activity in the nodes of the circuit under test (CUT), which is much higher during test mode than during normal mode operation. BIST is a design technique that allows a circuit to test itself. In this, the test performance achieved with the implementation of BIST is proven to be adequate to offset the disincentive of the hardware overhead produced by the additional BIST circuit. The technique can provide shorter test time compared to an externally applied test and allows the use of low-cost test equipment during all stages of production. BIST technique uses linear feedback shift register (LFSR) for generating test pattern. The proposed design, called bit-swapping LFSR (BS-LFSR), is composed of an LFSR and a 2 × 1 multiplexer. When used to generate test patterns for scan-based built-in self-tests, it reduces the number of transitions that occur at the scan-chain input during scan shift operation by 50% when compared to those patterns produced by a conventional LFSR. Hence, it reduces the overall switching activity in the circuit under test during test applications. The BS-LFSR is combined with a scan-chain-ordering algorithm that orders the cells in a way that reduces the average and peak power (scan and capture) in the test cycle or while scanning out a response to a signature analyzer. These techniques have a substantial effect on average- and peak-power reductions with negligible effect on fault coverage or test application time.

1. INTRODUCTION

Technology provides smaller, faster and lower energy devices which allow more powerful and compact circuitry, however, these benefits come with cost-the nano scale devices may be less reliable, thermal-and shot- noise estimations alone suggest that the fault rate of an individual nanoscale device may be orders of magnitude higher than today's devices. As a result, we can expect combinational logic be susceptible to faults. So in order to test any circuit or device we require separate testing technique which should be done automatically, for that purpose we are going to BIST. Running the test at a slower frequency than in normal mode. This technique of reducing power consumption, while easy to implement, significantly increases the test application time. Reduce the power consumption in scan-based built-in self-tests (BISTs) is by using scan chain-ordering techniques. These techniques aim to reduce the average-power consumption when scanning in test vectors and scanning out captured responses. Although these algorithms aim to reduce average-power consumption, they can reduce the peak power that may occur in the CUT during the scanning cycles, but not the capture

* 1,3,4,5 B. Tech Students, *2 Assistant Professor,
Dept. of Electronics and Communication Engineering,
RISE Krishna Sai Prakasam Group of Institutions

power that may result during the test cycle (i.e., between launch and capture). Modifying the test vectors generated by the LFSR to get test vectors with a low number of transitions. The main drawback of these techniques is, it results in lower fault coverage and higher test application time.

Modifying the test vectors generated by the LFSR to get test vectors with a low number of transitions. The main drawback of these techniques is, it results in lower fault coverage and higher test application time. Technology provides smaller, faster and lower energy devices which allow more powerful and compact circuitry, however, these benefits come with cost-the nano scale devices may be less reliable, thermal-and shot- noise estimations alone suggest that the fault rate of an individual nanoscale device may be orders of magnitude higher than today's devices. As a result, we can expect combinational logic be susceptible to faults. So in order to test any circuit or device we require separate testing technique which should be done automatically, for that purpose we are going to BIST. Over the last decade, network-on-chip (NoC) has emerged as a better communication infrastructure compared with bus-based communication network for complex chip designs overcoming the difficulties related to bandwidth, signal integrity, and power dissipation [1]. However, like all other systems-on-a-chip (SoCs), NoC-based SoCs must also be tested for defects. Testing the elements of the NoC infrastructure involves testing routers and inter router links. Significant amount of area of the NoC data transport medium is occupied by routers, which is predominantly occupied by FIFO buffers and routing logic. Accordingly, the probabilities of run-time faults or defects occurring in buffers and logic are significantly higher compared with the other components of the NoC. Thus, test process for the NoC infrastructure must begin with test of buffers and routing logic of the routers. In addition, the test must be performed periodically to ensure that no fault gets accumulated. The occasional run-time functional faults have been one of the major concerns during testing of deeply scaled CMOS-based memories. These faults are a result of physical effects, such as environmental susceptibility, aging, and low supply voltage and hence are intermittent (nonpermanent indicating device damage or malfunction) in nature [2]. However, these intermittent faults usually exhibit a relatively high occurrence rate and eventually tend to become permanent [2]. Moreover, wear-out of memories also cause intermittent faults to become frequent enough to be classified as permanent. Thus, there is a need for online test technique that can detect the run-time faults, which are intermittent in nature but gradually become permanent over time. Chip integration has reached a stage where a complete system can be placed in a single chip. When we say complete system, we mean all the required ingredients that make up a specialized kind of application on a single silico n substrate. This integration has been made possible because of the rapid developments in the field of VLSI designs. This is primarily used in embedded systems. Thus, in simple terms a SoCcan be defined as "an IC, designed by stitching together multiple s Tand -alone VLSI designs to provide full functionality for an application." A NoC is perceived as a collection of computational, storage and I/O resources on -chip that are connected with each other via a network of routers or switches instead of being connected with point to point wires. These resources communicate with each other using data packets that are routed through the network in the same manner as is done in traditional networks. It is clear from the definition that we need to employee highly sophisticated and researched methodologies from traditional computer networks and implement the mon chip. we have to explore the motivating factors that are compelling the researchers and designers to move toward the adoption of NoC architectures for future SoCs. The area of NoC is still in its infancy, which is one of the reasons why there are various names for the

same thing; some call it on chip networks, some networks on silicon, but the majority agrees upon “Networks on Chips” (NoCs). However, we will be using these terminologies interchangeably throughout our tutorial. NOC is Integrating various processors and on chip memories into a single chip Faults occur in NOC

- Permanent faults
- Transient fault

Nowadays, the area occupied by embedded memories in System-on-Chip (SoC) is over 90%, and expected to rise up to 94% by 2015. As those memories are very tightly integrated with large number of transistors causes 90% of overall faults in system on chips Thus, they concentrate the large majority of defects. In addition, with aggressive nanometer scaling, defect types are becoming more complex and diverse and may escape detection during fabrication test. If they are not treated adequately, the above trends will increase defect level, affect circuit quality dramatically and impact reliability, as undetected fabrication faults will be manifested as field failures. To cope with, the ability to guaranty a high quality test should be integrated in memory BIST, which is the mainstream test technology for embedded memories. Memory BIST generators can integrate a limited set of test algorithms (see for instance [1][2][3]). Thus, only the test algorithms selected during the design phase can be used after fabrication. However, fixing the memory test algorithms during the design phase is not a good strategy as unexpected failures may be discovered after production. Also, integrating pre-emotively a large number of test algorithms in the BIST generator will result in large area cost. Thus, programmable memory BIST enabling selecting the memory, y test stimuli in silicon and testing the memory for a wide variety of faults is becoming mandatory. This flexibility has to be achieved at low area cost, to make the approach attractive for real products. Also, the flexibility offered by programmable BIST is highly important for thorough screening inspection, failure analysis of customer returns, debug of a new fabrication process or a new memory design, and production ramp-up, since the most challenging issue in these processes is to detect and/or diagnose unexpected failures. There are three memory test stimuli components: the test algorithm determining the operations performed in each memory cell and the instances they performed; the data used in these operations; and the sequence in which the memory addresses are visited by the test algorithm. Previous work comprises programmable BIST enabling test algorithm programmability [4-8] and data programmability [7][9], but no previous work exist concerning address sequence programmability. In the present paper we extend programmable BIST to incorporated address sequence programmability in addition to test algorithm programmability and test data programmability. Thus, all the components of memory test stimuli could be programmed in silicon, enabling testing unexpected failures during fabrication go/no go test, as well as comprehensive testing and diagnosis during failure analysis of customer returns; debug of new fabrication process or new memory design; and production ramp-up. The main challenge when implementing complete programmability of the address sequence used concerns the large amount of data that have to be programmed (here the complete memory address space) and the associated high hardware cost. We resolve this problem by adapting the transparent BIST scheme [10-16] in a way enabling storing the address sequence in the memory under test and using it for testing the memory. This paper talks about walking, marching and galloping pattern tests for RAM. Random access memory circuits are among some of the highly dense VLSI circuits that are being fabricated today. Since the transistor lines are very close to each other, RAM circuits suffer from a very high average number of physical defects per unit chip area compared with other circuits. This fact has motivated researchers to develop efficient RAM test sequences that provide good fault

coverage. For testing today's high density memories traditional algorithms take too much test time. For instance GALPAT and WALKING I/O [5][6] require test times of order n^2 and $n^{3/2}$(where n is the number of bits in the chip). At that rate, assuming a cycle time of 100 ns, testing a 16Mbit chip would require 500 hours for an n^2 test and 860 seconds for an order $n^{3/2}$ test. Other older tests, such as Zero-One and Checkerboard, are of order n, but they have poor fault coverage. Due to the rapid progress in the very large scale integrated (VLSI) technology, an increasing number of transistors can be fabricated onto a single silicon die. For example, a state-of-the-art 130 nm complementary metal-oxide semiconductor (CMOS) process technology can have up to eight metal layers, poly gate lengths as small as 80 nm and silicon densities of 200K-300K gates/mm2 [37]. However, although milliongates integrated circuits (ICs) can be manufactured, the increased chip complexity requires robust and sophisticated test methods. Hence, manufacturing test is becoming an enabling technology that can improve the declining manufacturing yield, as well as control the production cost, which is on the rise due to the escalating volume of test data and testing times. Therefore reducing the cost of manufacturing test, while improving the test quality required to achieve higher product reliability and manufacturing yield, has already been established as a key task in VLSI design [8]. Embedded Memories are growing rapidly to a large amount in terms of its size and density. As Embedded memories are using complex design structures the chances of occurring manufacturing defects is more compared to any other embedded core on SOC. Hence testing of embedded memory is a real challenge for design architect. For SOC the inability to have direct access to a core is one of the major problems in testing and diagnosis. Further the available bandwidth between the primary inputs of the system chip and the embedded core is usually limited. Hence the external access for test purpose is often infeasible. This has prompted a very strong interest in self-test of embedded arrays. In particular, functional March tests have found wide acceptance, mostly because they provide defined detection properties for classical memory array faults such as stuck at faults and transition faults. Memory tests are used to confirm that each location in a memory device is working. This involves writing a set of data to each memory address and verifying this data by reading it back. If all the values read back are the same as those that were written, then the memory device is said to pass the test, otherwise device fails. Different test methodologies have been evolved from the years to identify the memory defects, one such test is memory built in self test which involves built in self test circuitry for each memory array.

2. LITERATURE SURVEY:

In a digital instrument designed for troubleshooting by signature analysis, this method can find the components responsible for well over 99% of all failures, even intermittent Built-in Self Test, or BIST, is the technique of designing additional hardware and software features into integrated circuits to allow them to perform self-testing, i.e., testing of their own operation (functionally, parametrically, or both) using their own circuits, thereby reducing dependence on an external automated test equipment (ATE). BIST is a Design-for-Testability (DFT) technique, because it makes the electrical testing of a chip easier, faster, more efficient, and less costly. The concept of BIST is applicable to just about any kind of circuit, so its implementation can vary as widely as the product diversity that it caters to. The basic idea of BIST, in its most simple form, is to design a circuit so that the circuit can test itself and determine whether it fault-free or faulty. This typically requires that additional circuitry and functionality be incorporated into the design of the circuit to facilitate the self-testing feature. This additional functionality must be capable of generating test patterns as

ISSN2321-2152
www.ijmece .com
Vol 12, Issue.1 March 2024

well as providing a mechanism to determine if the output responses of the circuit under test (CUT) to the test patterns correspond to that of a fault-free circuit. Basic approach of testing is shown in the below figure.

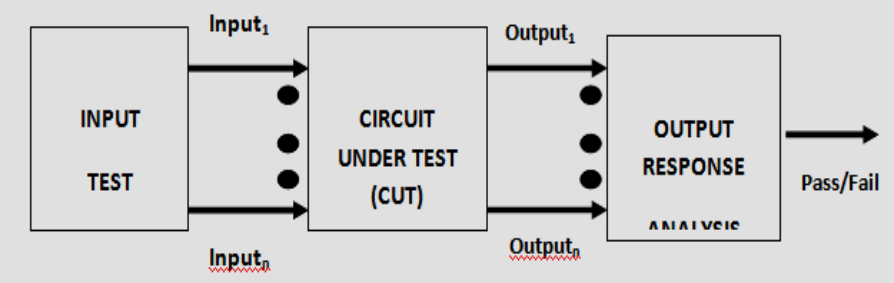


Figure: 1. Basic approach of Testing and boards

There are two kinds of memory test methods: electrical (technology-dependent) and functional (technology-independent). Electrical memory testing consists of parametric testing, which includes testing DC and AC parameters, IDDQ and dynamic testing for recovery, retention and imbalance faults [39]. DC and AC parametric tests are used to verify that the device meets its specifications with regard to its electrical characteristics, such as voltage, current, and setup and hold time requirements of chip's pins. Since embedded memories in SOCs usually do not have their I/O ports directly connected to chip's pins, parametric testing for embedded memories is not a necessity. IDDQ and dynamic testing [5] need a detailed description of the specific process technology. Additional information on electrical testing can be found in [8]. This thesis focuses on technology-independent functional memory testing, whose purpose is to verify the logical behavior of a memory core. Because functional memory testing allows for the development of cost-effective short test algorithms (without requiring too much internal knowledge of the memory under test), it is widely accepted by industry as a low-cost/high-quality solution. Support multiple test algorithms: The conventional MBIST approaches usually implement a single March test algorithm. However, deep submicron process technologies and design rules introduce physical defects that are not screened when using the memory test algorithms developed for previous process generations. There fore MBIST architectures should be programmable to support multiple memory test algorithms to increase the fault coverage and to find the most suitable algorithms for the manufacturing process at hand. 2. Diagnosis and repair support: Diagnosis support in an MBIST architecture is mandatory for manufacturing yield enhancement for new process technology and a rapid transition from the yield ramp phase to the volume production phase [19].

Built in Self-Test (BIST):

The trend to include more test logic on an ASIC has already been mentioned. Built-in self-test (BIST) is a set of structured-test techniques for combinational and sequential logic, memories, multipliers, and other embedded logic blocks. In each case the principle is to generate test vectors, apply them to the circuit under test (CUT) or device under test (DUT), and then check the response. BIST is a viable approach to test today's digital systems. With the ever increasing need for system integration, the trend today is to include in the same VLSI device a large number of functional blocks, and to package such devices, often, in Multi-Chip Modules (MCMs) that comprise complex systems. This leads to difficult testing problems in the manufacturing process and in the field. An

attractive approach to solve these problems is to use a multi-level integrated Built-In Self-Test (BIST) strategy**.** This strategy assumes that BIST is used at each level of manufacturing test, and it is reused at all consecutive levels, i.e. device, MCM, board, system. Boundary-Scan standard to realize self-testing at different levels. This strategy can only be realized.

LFSR:

The only linear function of single bits is xor, thus it is a shift register whose input bit is driven by the exclusive-or (xor) of some bits of the overall shift register value.

The initial value of the LFSR is called the seed, and because the operation of the register is deterministic, the stream of values produced by the register is completely determined by its current (or previous) state. Likewise, because the register has a finite number of possible states, it must eventually enter a repeating cycle. However, an LFSR with a well-chosen feedback function can produce a sequence of bits which appears random and which has a very long cycle.

Applications of LFSRs include generating pseudo-random numbers, pseudo-noise sequences, fast digital counters, and whitening sequences. Both hardware and software implementations of LFSRs are common.

Linear feedback shift registers (LFSRs) are commonly used as test pattern generators (TPGs) in low overhead built-in self-test (BIST) schemes. This is due to the fact that an LFSR can be built with little area overhead. Attainment of high fault coverage with sequences of practical lengths has traditionally been the main objective of BIST techniques. Even though this still remains the main objective, we believe, reducing heat dissipated during test application is becoming another important objective. The correlation between consecutive random patterns generated by an LFSR is low; this is a well-known property of LFSR-generated patterns. On the other hand, a significant correlation exists between consecutive vectors applied to the inputs of a circuit during its normal operation. Hence, switching activity in a circuit can be significantly higher during self-test than that during its normal operation. The hardware used in this paper for generating the primary input sequence consists of a linear-feedback shift-register (LFSR) as a random source [17], and of a small number of gates (at most six gates are needed for every one of the benchmark circuits considered). The gates are used for modifying the random sequence in order to avoid cases where the sequence takes the circuit into the same or similar reachable states repeatedly. This is referred to as repeated synchronization [18]. In addition, the on-chip test generation hardware consists of a single gate that is used for determining which tests based on will be applied to the circuit.

ISSN2321-2152
www.ijmece .com
Vol 12, Issue.1 March 2024

3. EXISTING METHOD:

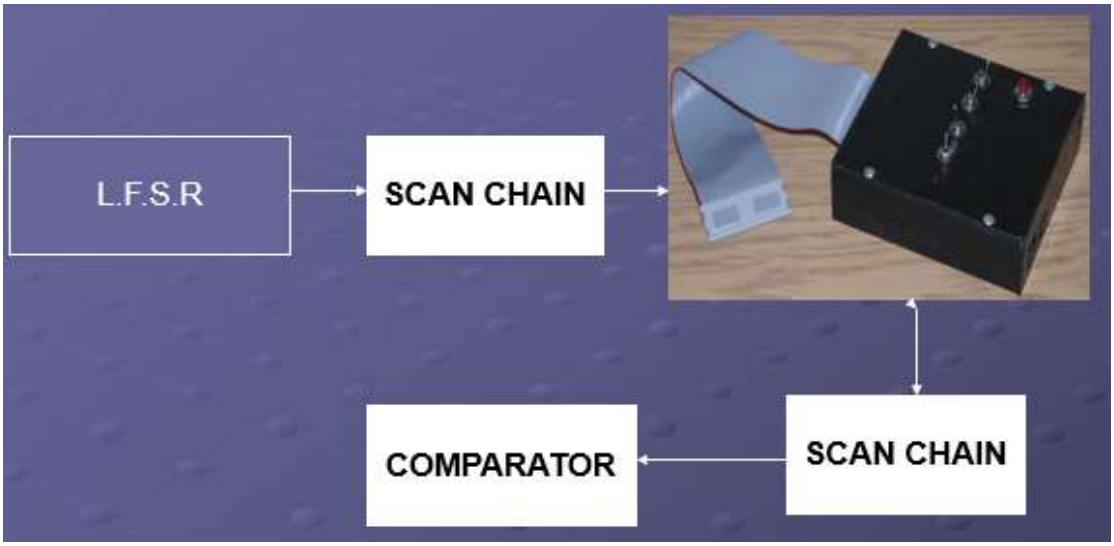


Fig:2. Existing block diagram

While designing a system (implementing a given algorithm or a function like multiplication, division etc.) a student can select needed microoperations for each unit of data path from the whole set of possible predesigned microoperations. Different architectures can be chosen for implementation of a given function. The control path is a microprogrammed controller which implements a finite state machine (FSM). The microprogram is developed by the student to realize a given algorithm based on the available (selected in prior) resources of the data path. The user fills in the microprogram table represented as a subpanel of the applet (Fig.2). Each microoperation has a gate-level implementation, and the number of gates determines the cost of the microoperation. By selecting a set of microoperations for the whole data path the student will get also the cost of the data path in the number of gates. Students can compare different solutions and find the tradeoffs. For every chosen architecture, the system calculates the cost of hardware. The speed (the number of clock cycles the microprogram needs) can be measured by simulation.

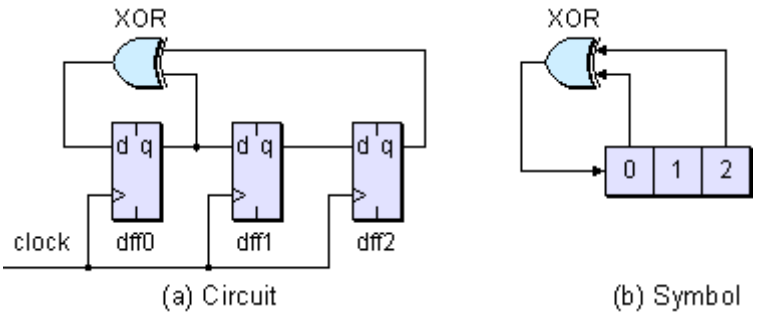
Simulation is carried out at the higher level by using Java subroutines (corresponding to functional units) which are activated by the control signals in the order given in the microprogram table. Fault simulation is carried out at the gate level by using SSBDDs [5]. Faults for the given block are inserted into the BDDs. The simulation process is controlled by the data in microprogram table. The target of the fault simulation (a block in the data path) is selected by a student and the results will be recorded.

Different possibilities of testing can be exercised: testing by working microprograms (called “functional testing”), generating special test microprograms for selected blocks, different BIST architectures like (BILBO, circular self test path, “store and generate” a.o.) [6] can be emulated and their quality can be measured in terms of fault coverage. New emerging technique called “functional BIST” [7] can be also investigated by combining different configurations of blocks with the goal to estimate their suitability for functional BIST.

LINEAR FEED BACK SHIFT REGISTER:

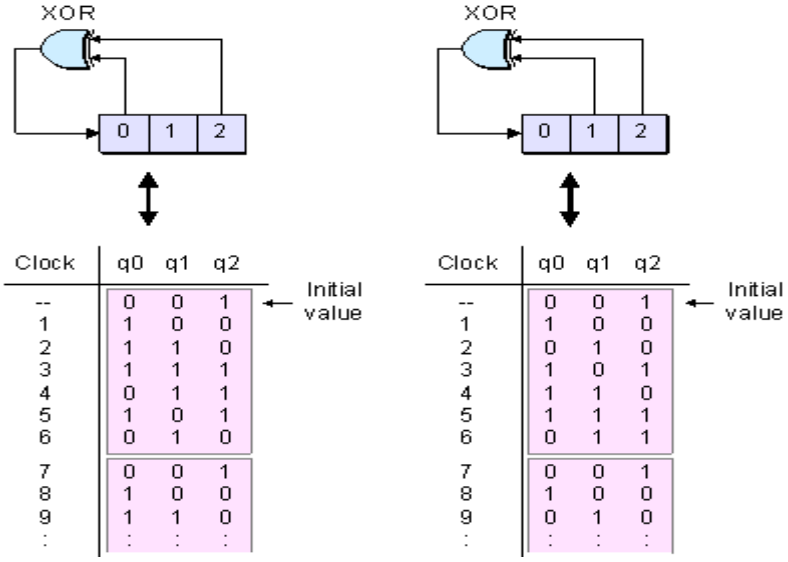
LFSRs are simple to construct and are useful for a wide variety of applications, but are often sadly neglected by designers. One of the more common forms of LFSR is formed from a simple shift register with feedback from two or more points, or taps , in the register chain (Fig 1).

ISSN2321-2152
www.ijmece .com
Vol 12, Issue.1 March 2024



1. LFSR with XOR feedback path.

The taps in this example are at bit 0 and bit 2, and can be referenced as [0,2]. All of the register elements share a common clock input, which is omitted from the symbol for reasons of clarity. The data input to the LFSR is generated by XOR-ing or XNOR-ing the tap bits; the remaining bits function as a standard shift register. The sequence of values generated by an LFSR is determined by its feedback function (XOR versus XNOR) and tap selection. For example, consider two 3-bit XOR based LFSRs with different tap selections (Fig 2).



2. Comparison of alternative tap selections.

Both LFSRs start with the same initial value but, due to the different taps, their sequences rapidly diverge as clock pulses are applied. In some cases an LFSR will end up cycling round a loop comprising a limited number of values. However, both of the LFSRs shown in Fig 2 are said to be of maximal length because they sequence through every possible value (excluding all of the bits being 0) before returning to their initial values.

4. PROPOSED METHOD:

BIST USING BIT-SWAPPING LFSR

The bit-swapping LFSR (BS-LFSR), is composed of an LFSR and a 2 × 1multiplexer. When used to generate test patterns for scan-based built-in self-tests, it reduces the number of transitions**.**

- The proposed BS-LFSR generates the same number of 1s and 0s at the output of multiplexers after swapping of two adjacent cells; hence, the probabilities of having a 0 or 1 at a certain cell of the scan chain before applying the test vectors are equal. Hence, the proposed design retains an important feature of any random TPG.
- In the BS-LFSR, consider the case that c1 will be swapped with c2 and c3 with c4, . . . , cn−2 with cn−1 according to the value of cn which is connected to the selection line of the

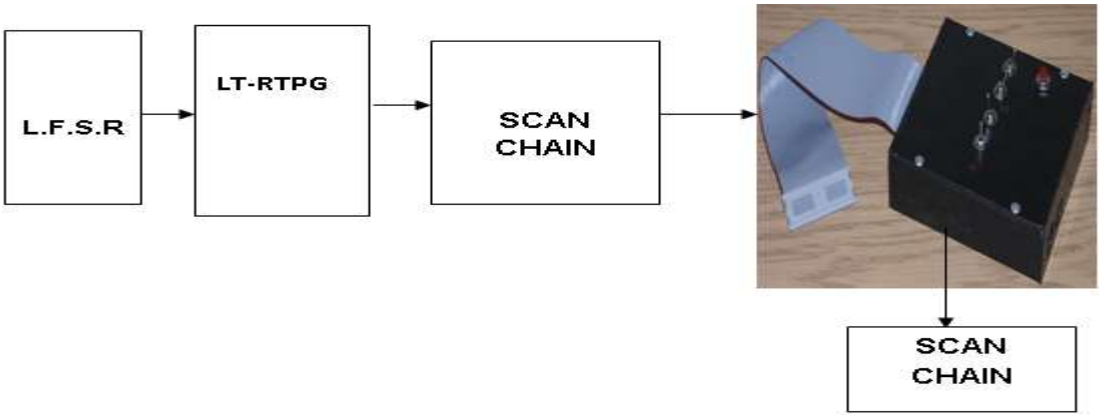
ISSN2321-2152
www.ijmece .com
Vol 12, Issue.1 March 2024

multiplexers. In this case, we have the same exhaustive set of test vectors as would be generated by the conventional LFSR, but their order will be different and the overall transitions in the primary inputs of the CUT will be reduced.

BIST is a viable approach to test today's digital systems. With the ever increasing need for system integration, the trend today is to include in the same VLSI device a large number of functional blocks, and to package such devices, often, in Multi-Chip Modules (MCMs) that comprise complex systems. This leads to difficult testing problems in the manufacturing process and in the field**.** An attractive approach to solve these problems is to use a multi-level integrated Built-In Self-Test (BIST) strategy**.**

This strategy assumes that BIST is used at each level of manufacturing test, and it is reused at all consecutive levels, i.e. device, MCM, board, system. Boundary-Scan standard to realize self-testing at different levels. This strategy can only be realized.

Linear feedback shift registers (LFSRs) are commonly used as test pattern generators (TPGs) in low overhead built-in self-test (BIST) schemes. This is due to the fact that an LFSR can be built with little area overhead. Attainment of high fault coverage with sequences of practical lengths has traditionally been the main objective of BIST techniques. Even though this still remains the main objective, we believe, reducing heat dissipated during test application is becoming another important objective.



BIT-SWAPPING L.F.S.R

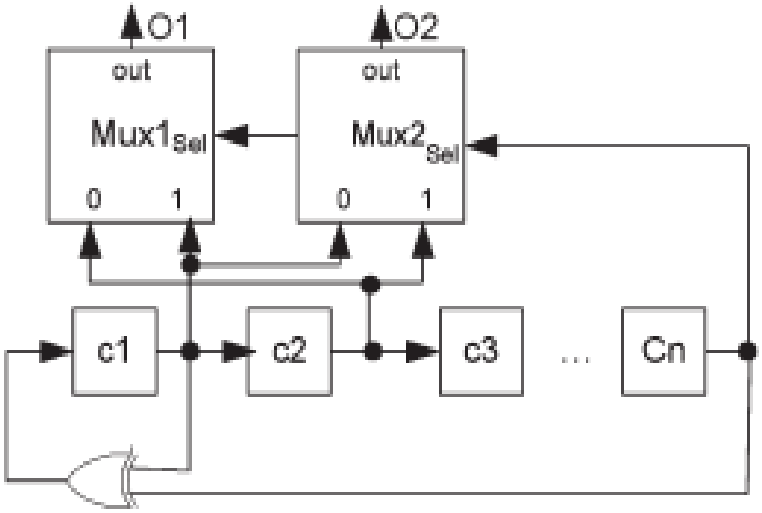


FIG:3. external L.F.S.R with bit-swapping

ISSN2321-2152
www.ijmece .com
Vol 12, Issue.1 March 2024

The above figure shows the external LFSR by using bit swapping technique. Bit swapping is a swap of a bit from 'm' position to tone 'n' position. It is used to reduce the transitions occurred at the input side of applying test vectors to circuit. The proposed BS-LFSR reduces the average and instantaneous weighted switching activity (WSA) during test operation by reducing the number of transitions in the scan input of the CUT. So from the above circuitry we can reduce the peak power while applying the test vectors to the circuitry. By reducing the number of transitions in the L.F.S.R we are going to reduce the peak power.

From this BIT SWAPPING technology we are going to reduce the peak power. By connecting multiplexers on the LFSR register as shown in above arrangement the number of transitions are decreased for that cell which are under bit swapping.

The below table shows the number of transitions in each register in LFSR without applying BIT SWAPPING technology, after applying bit swapping technology.

LFSR outputs of m, m+1									Multiplexers outputs O_1, O_2						
States			Next states			transition			states		Next States		transition		
c_1	c_2	c_n	c_1	c_2	c_n	c_1	c_2	$\sum$	O_1	O_2	O_1	O_2	O_1	O_2	$\sum$
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
			0	0	1	0	0	0			0	0	0	0	0
0	0	1	1	0	0	1	0	1	0	0	0	1	0	1	1
			1	0	1	1	0	1			1	0	1	0	1
0	1	0	0	0	0	0	1	1	1	0	0	0	1	0	1
			0	0	1	0	1	1			0	0	1	0	1
0	1	1	1	0	0	1	1	2	0	1	0	1	0	0	0
			1	0	1	1	1	2			1	0	1	1	2
1	0	0	1	1	0	0	1	1	0	1	1	1	1	0	1
			1	1	1	0	1	1			1	1	1	0	1
1	0	1	0	1	0	1	1	2	1	0	1	0	0	0	0
			0	1	1	1	1	2			0	1	1	1	2
1	1	0	1	1	0	0	0	0	1	1	1	1	0	0	0
			1	1	1	0	0	0			1	1	0	0	0
1	1	1	0	1	0	1	0	1	1	1	1	0	0	1	1
			0	1	1	1	0	1			0	1	1	0	1
∑Transitions						**8**	**8**	**16**					**8**	**4**	**12**

REDUCING THE AVERAGE POWER WHILE CAPTURING RESPONSES

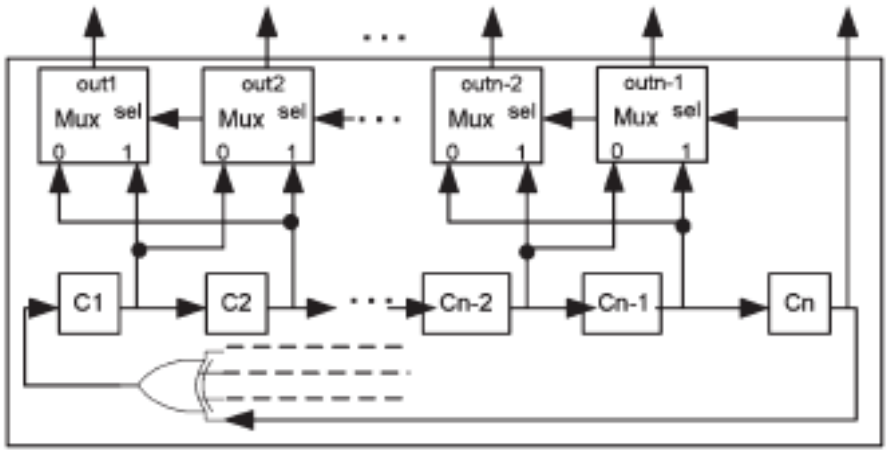
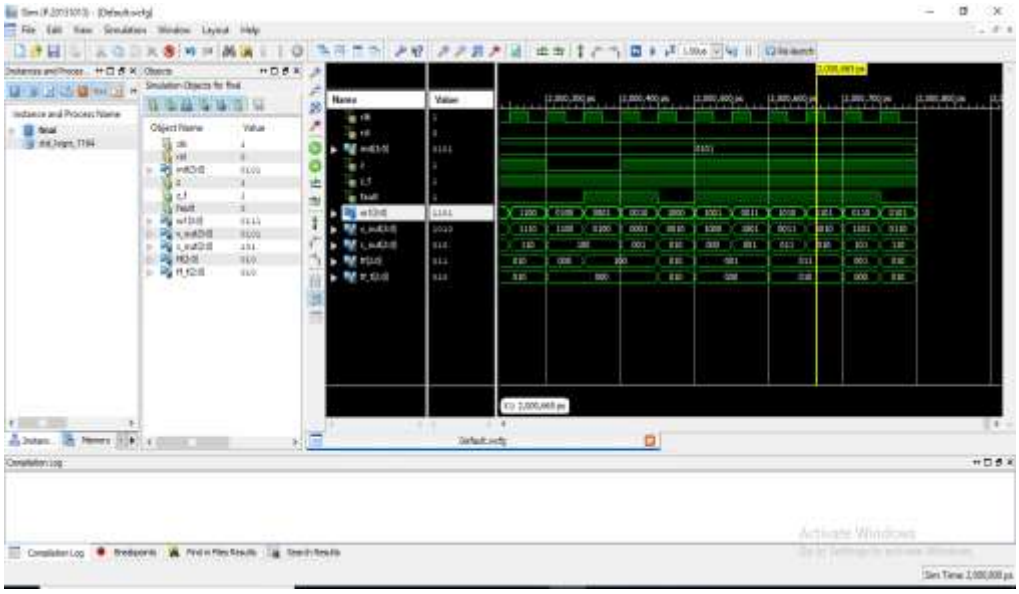
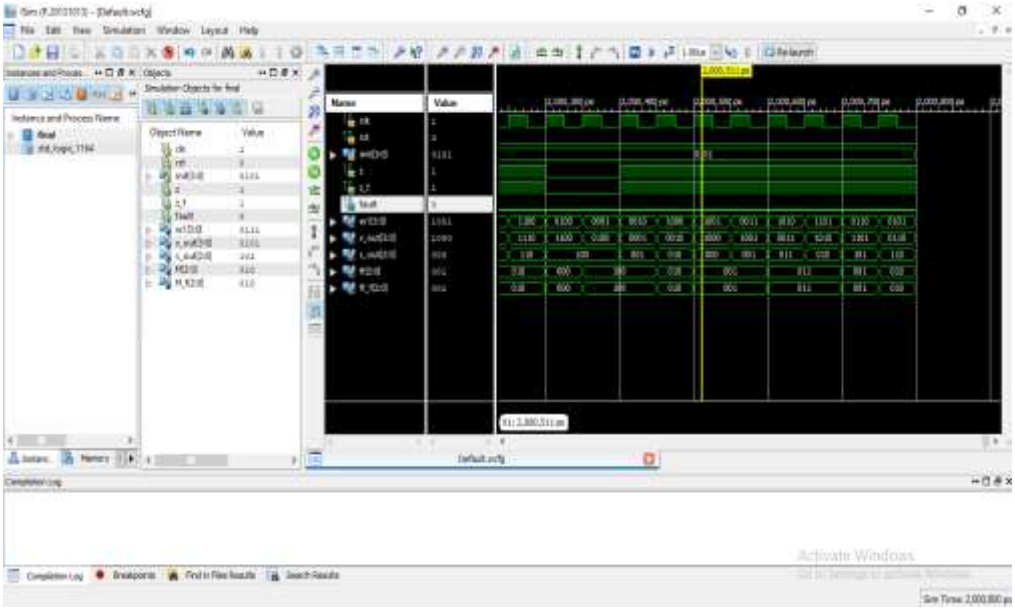


FIG:4.B.S-L.F.S.R can be used to generate exhaustive patterns

5. RESULTS:

ISSN2321-2152
www.ijmece .com
Vol 12, Issue.1 March 2024





ADVANTAGES and APPLICATIONS

1. 50% power is reduced.
2. Low hardware over area head.
3. High fault coverage

Mainly used in system booting process…..when we switch on the computer booting process starts.i.e., the system checks hard disk drives etc.,.whether they are working properly or not. So during booting process we can use bit swapping technique.

1. Cryptography. In this we show, that the statistical properties of cryptographic algorithms are the reason for the excellent pseudorandom testability of cryptographic processor cores. The work is especially concerned with modern symmetric block encryption algorithms and their VLSI implementations. For the examination typical basic operations of these cryptographic algorithms are categorized in classes and analyzed regarding their pseudorandom properties. Based on the results the pseudorandom properties of symmetric block

ciphers can be determined by means of data flow graphs (DFG) and so-called predecessor operation lists.

This is demonstrated with a paradigm algorithm, the symmetric block cipher 3WAY. The results of the theoretical analysis lead to a so-called global BIST concept for cryptographic processor cores. This self-test approach is characterized by central pseudorandom pattern generators and signature registers at the primary inputs and outputs of the cores. The global BIST is exemplarily applied to an implementation of the 3WAY algorithm. Finally, the quality of the developed test approach is determined by fault simulations.

2. Novel Built-In Self-Test (BIST) approaches for integrated optoelectronic systems are presented. The methods are compatible with scan chain design and allow testing the internal functionality of the device, the interconnection between modules, the analog characteristics of the transmitters and receivers and the Bit Error Rate (BER) of the channels.

The proposed approaches enable system evaluation under realistic operating conditions (including crosstalk degradation) through BER testing. They can speedup both circuit-level and system-level testing by providing means for accessing different parts of the circuit easily and allowing time-consuming performance tests to be run in parallel with boundary-scan tests.

6. CONCLUSION

A low-transition TPG that is based on some observations about transition counts at the output sequence of LFSRs has been presented. The proposed TPG is used to generate test vectors for test-per scan BISTs in order to reduce the switching activity while scanning test vectors into the scan chain. Furthermore, a novel algorithm for scan-chain ordering has been presented.

When the BS-LFSR is used together with the proposed scan-chain-ordering algorithm, the average and peak powers are substantially reduced. The effect of the proposed design in the fault coverage, test-application time, and hardware area overhead is negligible. Comparisons between the proposed design and other previously published methods show that the proposed design can achieve better results for most tested benchmark circuits.

7. FUTURE ENHANCEMENT

Fault Tolerant system

The future enhancement of the FPGA implementation of Bit-Swapping LFSR along with Scan-Chain Ordering is to add a block which identifies the component with fault. It not only identifies the fault in the system but also shows where the fault has occurred.

REFERENCES:

[1] P. H. Bardell, W. H. McAnney, and J. Savir, Built-In Test for VLSI: Pseudorandom Techniques. New York: Wiley, 1987.

[2] S. Hellebrand, J. Rajski, S. Tarnick, S. Venkataraman, and B. Courtois, “Built-In test for circuits with scan based on reseeding of multiple- polynomial linear feedback shift registers,” IEEE Trans. Comput., vol. 44, no. 2, pp. 223–233, Feb. 1995.

[3] N. Zacharia, J. Rajski, and J. Tyszer, “Decompression of test data using variable-length seed LFSRs,” in Proc. IEEE 13th VLSI Test Symp., 1995, pp. 426–433.

[4] S. Hellebrand, S. Tarnick, and J. Rajski, “Generation of vector patterns through reseeding of multiple-polynomial linear feedback shift regis-ters,” in Proc. IEEE Int. Test Conf., 1992, pp. 120–129

[5] N. A. Touba and E. J. McCluskey, “Altering a pseudo-random bit se- quence for scan-based BIST,” in Proc. IEEE Int. Test Conf., 1996, pp. 167–175.

[6] M. Chatterjee and D. K. Pradhan, “A new pattern biasing technique for BIST,” in Proc. VLSITS, 1995, pp. 417–425.

[7] N. Tamarapalli and J. Rajski, “Constructive multi-phase test point in sertion for scan-based BIST,” in Proc. IEEE Int. Test Conf., 1996, pp. 649–658.

[8] Y. Savaria, B. Lague, and B. Kaminska, “A pragmatic approach to the design of self-testing circuits,” in Proc. IEEE Int. Test Conf., 1989, pp. 745–754.

[9] J. Hartmann and G. Kemnitz, “How to do weighted random testing for BIST,” in Proc. IEEE Int. Conf. Comput.-Aided Design, 1993, pp. 568–571.

[10] J. Waicukauski, E. Lindbloom, E. Eichelberger, and O. Forlenza, “A method for generating weighted random test patterns,” IEEE Trans. Comput., vol. 33, no. 2, pp. 149–161, Mar. 1989.

[11] H.-C. Tsai, K.-T. Cheng, C.-J. Lin, and S. Bhawmik, “Efficient test- point selection for scan-based BIST,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 6, no. 4, pp. 667–676, Dec. 1998.

[12] W. Li, C. Yu, S. M. Reddy, and I. Pomeranz, “A scan BIST generation method using a markov source and partial BIST bit-fixing,” in Proc. IEEE-ACM Design Autom. Conf., 2003, pp. 554–559.

[13] N. Z. Basturkmen, S. M. Reddy, and I. Pomeranz, “Pseudo random patterns using markov sources for scan BIST,” in Proc. IEEE Int. Test Conf., 2002, pp.