I J C T A, 9(34) 2016, pp. 367-380
© International Science Press

VLSI Architecture for an Efficient Memory Built in Self Test for Configurable Embedded SRAM Memory

Nisha O. S.[1] and Dr. K. Siva Sankar[2]

ABSTRACT

Memories are the most dominating blocks present on a chip. All types of chips contain embedded memories such as a Read Only Memory (ROM), Static Random Access Memory (SRAM), Dynamic Random Access Memory (DRAM), and flash memory. Testing of these memories is a very tedious and challenging job as area over head, testing time and cost of the test play an important role. In this work an efficient VLSI architecture for MBIST (Memory Built in Self Test) which incorporates a modified March Y- algorithm using concurrent technique and a modified Linear Feedback Shift Register (LFSR) based address generator is proposed. Because of concurrency in testing the sequences the test results were observed in less time than the traditional March tests. The synthesis and simulation of the design is done using Xilinx ISE software. The design is coded in Verilog and the experimental result when compared with similar existing works shows a reduction in complexity and delay time.

Keywords: MBIST, March algorithm, SRAM, Address generator, Xilinx.

1. INTRODUCTION

Today's semiconductor technology is making it feasible to integrate millions to billions of circuit elements e.g., diodes, transistors and other components such as resistors and capacitors, together with interconnections, within a very small Silicon area [1]. The shrinking of technology has created various mechanical and physical defects such as opens, bridges and shorts causing more parasitic capacitances, leakage power and their severe effects on the overall performance of the System-on-Chips (SoCs). Earlier, the SoCs were influenced by its functional cores, but because of the increased demand of high data storage, SoC area is currently being influenced by the on-chip Volatile or Non-volatile memory blocks [2], [3]. As a result of the survey made by Semiconductor Industry Association (SIA), the semiconductor memories are expected to occupy 94% of SoC area in 2014 as compared to 71% in 2005.The user data will be generally stored in volatile SRAM, whereas the nonvolatile ROMs store the system programs as well as the test program and test vectors by which the system will be tested for any manufacturing defects and functional errors during test mode [4]. The exact functionality of the memory chips is becoming vital in SoCs, because the control data/signals used for controlling and functioning of almost all the blocks in SoCs, their scheduling information in addition to the user data are being stored on-chip to reduce the latency as compared to that of storage of these data in an off-chip storage device.

The most commonly occurring functional faults [5], [6] in regular 2D memory arrays include Address decoder Faults (AFs), Stuck-At Faults (SAFs), Neighborhood Pattern Sensitive Faults (NPSFs) and Coupling Faults (CFs). As the memories in SoCs are mostly occupying the higher chip area, the study and application of fault models that can target these memory faults has become significant for cost effective memory

[1] Assistant Professor, Faculty of Computer Science and Information Technology, Lourdes Matha College of Science and Technology, Trivandrum, Kerala, *E-mail: nishaos0587@gmail.com*

[2] Assistant Professor, Department of Information Technology, Nooral Islam University, Thuckalay, Thamil Nadu

testing. Unlike the logic blocks, the memory arrays have an identical 2-dimensinal structures, the MBIST algorithms may be commonly applied for testing of all memory arrays available in SoCs. The memory can be effectively tested using march algorithms [7], [8].

Power dissipation is becoming a challenging problem for the VLSI design engineers and testing engineers because the power consumed by the system in testing mode is 200% more than in its normal mode [9]. There are two types of power dissipation 1. Static power 2. Dynamic power and in this case we are going to consider the dynamic power dissipation. The dynamic power dissipation is calculated from an equation.

$$P_{avg} = \alpha T.C_{load}V^{2}{}_{dd}.f_{Clk} \tag{1}$$

Where,

αT -Switching activity factor of the gate.

C_{load} -Total load capacitance

V_{dd} -Supply voltage

f_{Clk} -Operating frequency

In the above equation the average power is directly proportional to the ‘áT’. Therefore the power dissipation during testing can be reduced by controlling the switching activity .The advancement in submicron manufacturing technology and system-on-a-chip (SOC) design methodology has led to a large number of cores, especially the memory cores, are now integrated into a single chip. It has been predicted that by the year of 2014, memory cores may occupy 94% area of a typical SOC [10]. Memory thus plays an important role in SOC. Since the probability of memory fault is more compared to that of other type of faults in a circuit the need for testing of memory is more important. However due to the availability of a small number of I/O pins in a circuit BIST for Memory (MBIST) is used as a solution to this problem [11]. In conventional MBIST the address bus, data bus, and read/write control signals that are generated by the test pattern generator of the BIST are applied to the memory under test. The address bus indicates the memory location, the Read/write control determines the operation (read/write) to perform in this particular memory location and the data bus include the data to write or to read in the memory location the address bus indicate.

The remainder of this paper will discuss SRAM testing algorithms available in current literature and discuss uses of industry standard testing algorithms for testing configurable memory resources (Section 2). Section 3 will discuss on the research methodology with respect to the proposed scenario. In section 4 the complete description of the implemented MBIST architecture is exhibited with neat sketches. Section 5 will justify the efficient of this methodology with suitable experiments and analysis. And finally, Section 5 will offer a summary and conclusion of our work.

2. RELATED WORKS

K. Murali Krishna and M. Sailaja in [12] implemented an LFSR based address generator with reduction in switching activity for low power MBIST. In that method the address patterns were generated by a combination of LFSR and a 2-bit pattern generator (Modified LFSR) and two separate clock signals. By using the modified architecture switching activity was reduced. Since the switching activity was proportional to the power consumed, reducing the switching activity of the address generator reduces the power consumption of the MBIST. They had designed and stimulated their address generator using Xilinx ISE tools and compared with the switching activities of the conventional LFSR and BS-LFSR. Results showed a reduction in switching activity and a reduction of more than 90% of the total dynamic power when compared to conventional LFSR.

Balwinder Singh *et.al* in [13] implemented all March algorithms used for memory testing in verilog for the testing of 1Kb and 4kbmemories with BIST. After designing and implementation, their performance is

compared on the basis of their length, number of cycles used during writing and reading of memory, and area overhead. All that comparison was presented in the tabular form. From comparison on the basis of area overhead, it is observed that for the memories of smaller in size March X is most efficient and March AB is the least efficient, and Vice versa in the case of March Y and March LA . During the analysis power consumption it was examined that March X and March C consume least power in case of 1 Kb memory and 4 Kb memories respectively. March C, X and Y takes the least number of cycles for memory writing and reading in 1Kbmemory and March Y in case of 4 Kb memory.

Che-Wei Chou *et.al* in [14] presented a low-cost built-in self-diagnosis (BISD) scheme for NAND flash memories, which could support the March-like test algorithms with page-oriented data backgrounds. Two simple test time reduction techniques were also designed to reduce the test time. Experimental results showed that the proposed BISD circuit for a 2M-bit flash memory only needs 1.7K gates. Also, their new test time reduction techniques could effectively reduce the test time. Analysis results showed that they can reduce the test time to 48.628% of the normal test scheme for a 4G-bit flash memory tested by the March-FT test algorithm with solid data backgrounds.

A systematic approach in testing flash memories, including the development of March-like test algorithms, cost effective fault diagnosis methodology, and built-in self-test (BIST)scheme was presented in [15] by Jen-Chieh Yeh *et.al* . The improved March-like tests algorithms can detect disturb faults derived from the IEEE STD 1005 and conventional faults. As the memory array architecture and/or cell structure varies, the targeted fault set may change. They have developed a flash-memory fault simulator called RAMSES-FT, with which they could easily analyze and verify the coverage of targeted faults under any given test algorithm. In addition, the RAM test algorithm generator test algorithm generator by simulation has been enhanced based on RAMSES-FT, so that one can easily generate tests for flash memories, whether they are bit- or word-oriented. Their newly designed fault diagnosis methodology helps improve the production yield. They also developed a built-in self-diagnosis (BISD) scheme a BIST design with diagnosis support. The BISD circuit collects useful test information for off-chip diagnostic analysis. It had unique test mode control that reduces test time and diagnostic data shift-out cycles by a parallel shift-out mechanism.

In [16] Masnita et.al, had designed a data and read/write controller as a finite state machine (FSM) BIST that would generate test patterns based on the march-based diagnostic algorithm developed to distinguish between stuck-at and transition faults. A description related to SAFs and TFs were presented with the intention of covering the aspect of distinguishing both of these faults. Related design based on the selected algorithm was also presented with simulation results to show the functionality of the design. The design of the controller can be used to build a complete MBIST engine to test the effectiveness of the proposed algorithm in distinguishing SAFs from TFs.

Manikandan.B and Praveen kumar.J implemented a FSM-based programmable memory built in self test (MBIST) Controller used for testing the memory devices in [17]. The MBIST controller was designed to implement a new test algorithm known as March based test algorithm. The controller and test algorithm are studied and designed using verilog HDL and implemented in SPARTAN-3E FPGA. The simulation portrays that the tested data and the expected data are able to be compared in the architecture. The implemented controller has the ability to detect faulty or good memory ICs. Synthesis result shows that the FSM -based HP-MBIST controller employed only 75 instances with clock frequency 246.15 MHz with a less usage of Logic Elements (LE) with High speed testing of memories. It was also justified that the FSM-based HP-MBIST controller consumes less area overhead and high speed while the other compared designs consumed more area overhead and less speed. The experimental results also showed that the presented BIST can be implemented with low area overhead.

K Padma Priya in [18] presented a high speed FSM-based controller for programmable memory built-in self test for testing memory devices. Her technique was popular because of its flexibility of new test

algorithms. The architecture of controller was designed to implement a new test algorithm has less number of operations and the designed algorithm emphasis testing of high density memory ICs either faulty or good .The components of controller was analyzed and designed using Verilog HDL. The analysis of the timing, logic area usage and speed are also presented.

In [19] M. Jahnavi *et.al*. Presented the implementation of online test scheme for RFID memories based on Memory Built in Self Test (MBIST) architecture. In that work they presented the, Symmetric transparent version of March SS algorithm, implementation of Memory BIST. The comparison between the different march algorithms and the advantage of the March SS algorithm over all other is also presented. The whole design was implemented using Verilog HDL and was, in turn, verified on Xilinx ISE 13.2 simulator, and synthesized.

3. RESEARCH METHODOLOGY

The test algorithm and address generation are to be optimized to cover all the faults, reduce testing time and consume less power. This can be achieved by employing evolutionary algorithms in selecting the patterns such that the inputs of design switch minimally. Test algorithm is designed using these evolutionary algorithms so that the test vectors selected can be used for reducing the switching activity in the circuit and also by maintain the fault coverage. Considering the above facts, in this work the objective is to implement a highly efficient MBIST architecture. Since the main source of power and area consumption of the whole system is dependent on the address generator and the test algorithm used in the MBIST architecture, the address generator in the conventional MBIST can be replaced with a LFSR based address generator with low switching activity [11] with modification to its operations based on the adapted testing algorithm. After analyzing the Area Overhead and Power Analysis of March Algorithms for Memory BIST [12] it is obvious that the MARCH-Y algorithm requires low area and power with less clock cycle for its operation, So in this work for better result, an modification to the conventional March Y algorithm can be done for efficient contribution. The MBIST controller can be designed using a FSM with better state transition. The complete architecture can be coded in verilog-HDL, synthesized and simulated using Xilinx 14.1 software in windows platform. The simulated results for area, power and speed can be computed and compared with other existing MBIST architectures reported in recent works.

4. PROPOSED METHOD

March Y test is an extension of March X. This test is of complexity 8n and can detect all faults detectable by MARCH X. The conventional March Y test algorithm is given as,

$$\Updownarrow \boldsymbol{W}0; \Uparrow \boldsymbol{R}0, \boldsymbol{W}1, \boldsymbol{R}1; \Downarrow \boldsymbol{R}1, \boldsymbol{W}0, \boldsymbol{R}0; \Updownarrow \boldsymbol{R}0 \quad (2)$$

When implementing this algorithm in hardware the working process include the following steps.

Step 1: Write the data ‘0’ to all the address locations without considering the order of address.

Step 2: With an increasing address order Read the data ‘0’, Write the data ‘1’ and Read the data ‘1’ within a single location.

Step 3: With a decreasing address order Read the data ‘1’, Write the data ‘0’ and Read the data ‘0’ within a single location.

Step 4: Finally Read the data ‘0’ to all the address locations without considering the order of address.

The three operations mentioned in steps 2 and 3 is performed adjacently with increasing or decreasing address order. The main drawbacks in the hardware realization of the conventional March Y algorithm make the MBIST processor to perform its operations with high power consumption and with low speed. A clear analysis of the hardware implementation of March Y algorithm reveals that the drawbacks are due to

the address generator and the delay in process. So in order to make the system to perform efficiently with high speed with low power supply, the address generator and the process delay have to be modified. For reducing the processing time the March Y algorithm can be modified as given below,

$$\Updownarrow \boldsymbol{W}0;\begin{Bmatrix}\Uparrow \boldsymbol{R}0,\boldsymbol{W}1,\boldsymbol{R}1\\ \Downarrow \boldsymbol{R}1,\boldsymbol{W}0,\boldsymbol{R}0\end{Bmatrix};\Updownarrow \boldsymbol{R}0 \tag{3}$$

Hence after the modification as given in equation (3) the working process include the following steps.

Step 1: Write the data ‘0’ to all the address locations without considering the order of address.

Step 2: With an increasing address order Read the data ‘0’, Write the data ‘1’ and Read the data ‘1’ within a single location at the same time with a decreasing address order Read the data ‘1’, Write the data ‘0’ and Read the data ‘0’ within a single location.

Step 3: Finally Read the data ‘0’ to all the address locations without considering the order of address.

The block schematic for our modified MBIST engine is shown in figure 1.

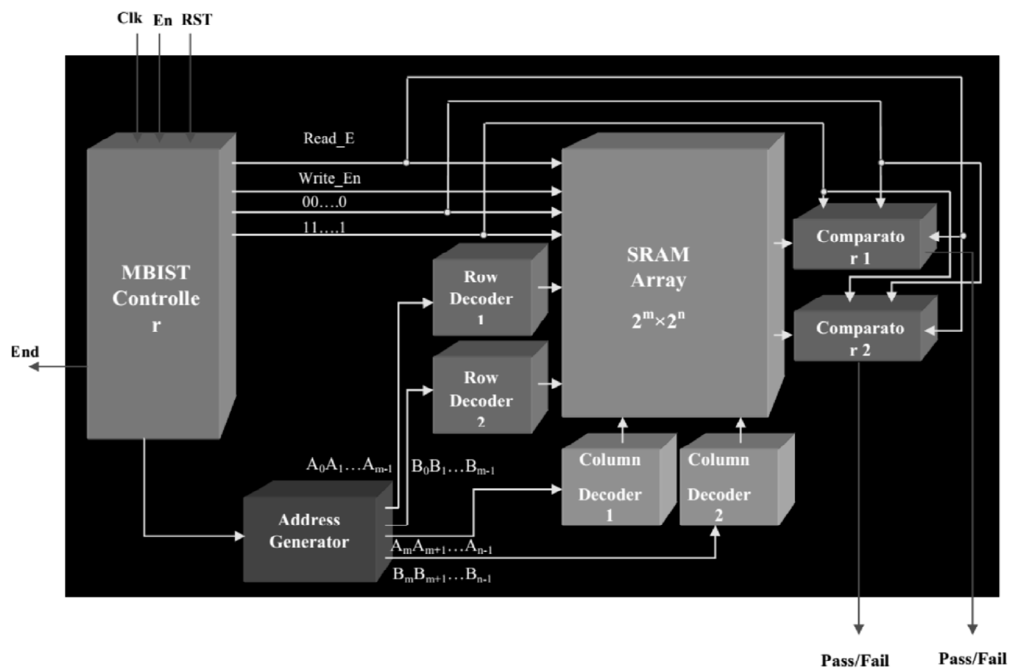

Figure 1: Proposed MBIST block schematic

The MBIST engine implemented in this work consists of a Modified SRAM structure, Address Generator, Comparators and MBIST Controller. The MBIST Controller generates the control signals depending on the clock cycle with respect to our modified March algorithm. The address generator generates the address locations where the Read or Write operations is to be performed with the order mentioned in the algorithm depending on the control signal from the MBIST controller. The Modified SRAM structure consists of an additional Row and Column decoder which is enabled only when the step 2 operations are performed. Since this modification are disabled during the step 1 and 3 process, this will not affect the power consumption, whereas a minor increase in the overall hardware of the system occurs which has no effect on the MBIST engine. Two comparators are used for comparing the stored memory data in the specified location and the read out data. If both the data are same then pass signal is high if not then fail signal goes high and the testing process is aborted.

Modified SRAM Structure

The SRAM architecture is shown in figure.2 is described in detail. With the support of circuitry to decode addresses, and to execute the required read and write operations, in which the basic architecture of a static RAM includes one or more rectangular arrays of memory cells. Extra support circuitry are used to implement special features, for instance burst operation, may also be present on the chip. SRAM memory arrays are set in rows and columns of memory cells called wordlines and bitlines, correspondingly. Defined by the intersection of a row and column, each memory cell has a unique location or address. In particular data input/output pin every address is allied. The speed at which the memory must operate layout and testing requirements, the total size of the memory and the number of data I /Os on the chip is used to determine the number of arrays on a memory chip. A bi-stable flip-flop made up of four to six transistors, is an SRAM memory cell. The flip-flop possibly is both of the two states that can be interpreted by the support circuitry to be a 1 or a 0. The memory chip's support circuitry that allows the user to read and the write the data stored in the memory cells. This circuitry generally includes:

- To select rows and columns address logic is used.
- In a memory cell, translation logic "reads" the data and sends that data to the data I/O.
- At the input, the write logic that takes the user data applied and stores it in a memory cell.
- Unless specifically preferred, output enables logic to avoid data from appearing at the outputs.
- Burst address sequences, pipelined data, and other control functions on the chip are tracked by internal counters and register.
- To control the timing of the read and write operations and each of their variations, clock circuitry is used.

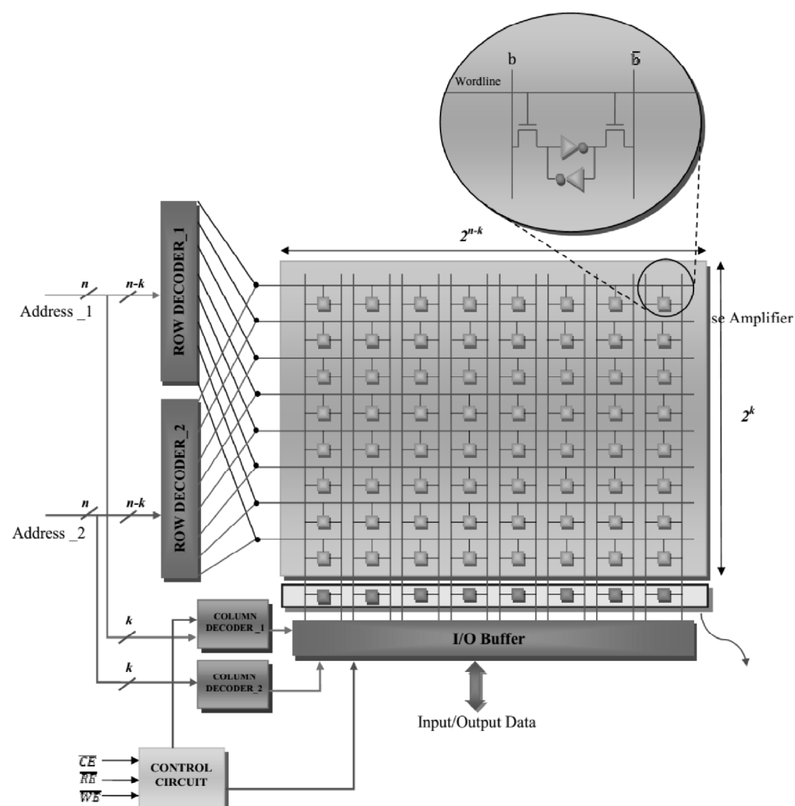

Figure 2: Modified SRAM architecture

For performing the simultaneous process in step 2 of our modified March algorithm, we have added an extra row and column decoder blocks as shown in figure 2.The extra row and column decoder is enabled if only there is a output signal from address input 2 and hence this may not affect the power consumption of our MBIST engine.

Address Generator

According to our modified March Y Algorithm in eqn. (3), we need two address generators for performing the step 2 operations simultaneously. But using two address generators increase the complexity of the complete architecture with high area and power consumption. So in order to implement efficient hardware architecture in this work, a low power address generator using LFSR implemented in [12] is adapted. The address generator is modified in such a way that it can perform the address generations as per the demand for our modified March Y algorithm. The address generator adapted for the MBIST architecture is shown in figure 3.

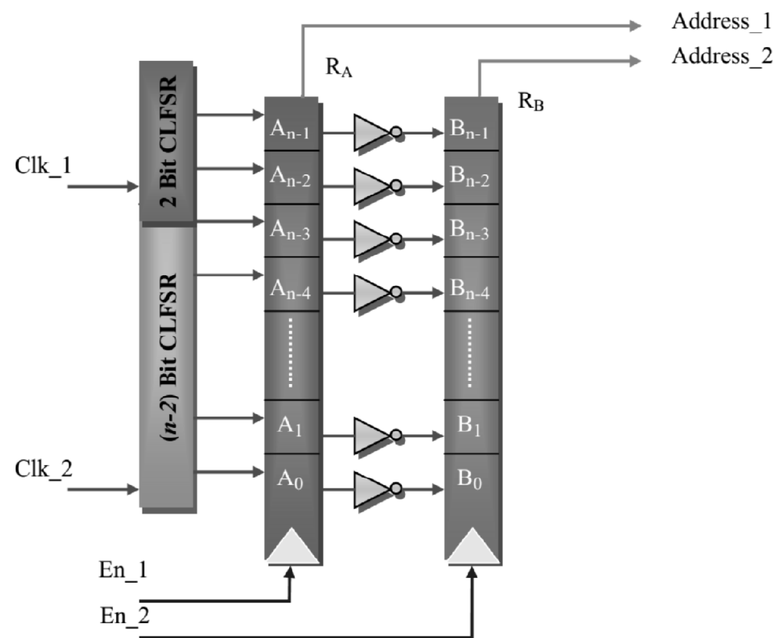

Figure 3: Modified Address generator

The output from the address generator (R_A) and its complement (R_B) are enabled by En_1 and En_2 respectively. The R_A values are fed to the Address_ 1 and R_B values to the Address _ 1.The enable signals are generated by the MBIST controller and the values with respect to the steps 1, 2 and 3 discussed previously are tabulated in table 1.

Table 1
Address generator control

Process	*En_1*	*En_2*
Step 1	1	0
Step 2	1	1
Step 3	1	0

During the process of writing ‘0’ independent to the order of address generator in step 1 En_1 is made ‘1’ and En_2 as ‘0’,hence the address generator outputs only the ‘R_A’ as address location thereby enabling

row decoder_1 and column decoder_1 whereas row decoder_2 and column decoder_2 remains idle. For the step 2 processes which include two simultaneous parallel operations two address locations are needed one with an increasing order and the other with a decreasing order. The LFSR based address generator generates an output which when considered as the increasing order then its complement is the decreasing order. This property of the address generator we have adapted is made use for implementing our design with a better performance. Also the probability of same address generation in address_1 and address_2 for our modified address generator is zero. For this the En_1 and En_2 values are made '1', thereby generating two address locations simultaneously as per the algorithmic requirement. For the process in step 3 the address generator performs a similar operation to that of step 1.

Comparator

The comparator logic compares the read out data that is stored in a particular bit cell with the original value that is written in the bit cell for faults. This comparator is enabled only if the read enable signal goes high. The same En_1 and En_2 signal that controls the enable function of the address generator is connected to the comparator_1 and comparator _2. When the output value from a particular cell which is previously stored with '1' exhibit '0' then a fault is detected and the comparator return a 'fail' output and the complete testing process is aborted, if the read out value is '1' then the comparator exhibits a 'pass' output and the next process takes place.

MBIST Controller

The MBIST controller controls the functioning of Read_En, Write_En of the SRAM and En_1, En_2 of the address generator. The functioning of MBIST controller with respect to the clock cycle for step 1, 2 and 3 process is tabulated in table 2, 3 and 4 below.

Table 2
Control signals for step 1 process of our modifier March Y algorithm

Clk	*Read_En*	*Write_En*	*AG*		*Data*	*Comparator*	
			En_1	*En_2*		*En_1*	*En_2*
0 to 2^n-1	0	1	1	0	000….00	1	0

Table 3
Control signals for step 2 process of our modifier March Y algorithm.

Clk	*Read_En*	*Write_En*	*AG*		*Data*	*Comparator*	
			En_1	*En_2*		*En_1*	*En_2*
0 to 2^n-1	1	0	1	1	000….00111….11	1	1
	0	1	1	1	111….11000….00	1	1
	1	0	1	1	111….11111….11	1	1

Table 4
Control signals for step 3 process of our modifier March Y algorithm.

Clk	*Read_En*	*Write_En*	*AG*		*Data*	*Comparator*	
			En_1	*En_2*		*En_1*	*En_2*
0 to 2^n-1	1	0	1	0	000….00	1	0

5. RESULTS AND DISCUSSION

In this section, we implement the newly designed architecture for MBIST which we have discussed in the previous sections to analyze its area, speed and power requirements. We have selected the Xilinx® Virtex™-

5 xc5vlx20t-2-ff323 device as the target FPGA. The proposed architecture is modeled in verilog-HDL and synthesized for different address length (5-bits and 10-bits) using XST™ of Xilinx® ISE™ version 14.1 design software. And all the experiments were performed on 3.10GHz Intel(R) i5, 4.00GB RAM, and 32-bit operating system with windows7 professional.

The synthesized design can be viewed as a schematic in the register transfer level (RTL) viewer. The RTL Schematic shows a representation of the pre-optimized design in terms of generic symbols such as AND gates and OR gates adders, multipliers, counter, that are independently of the targeted Xilinx device. Figure 4 and 5 exhibits RTL schematic of our MBIST architecture with 5 and 10 bits address generator respectively.

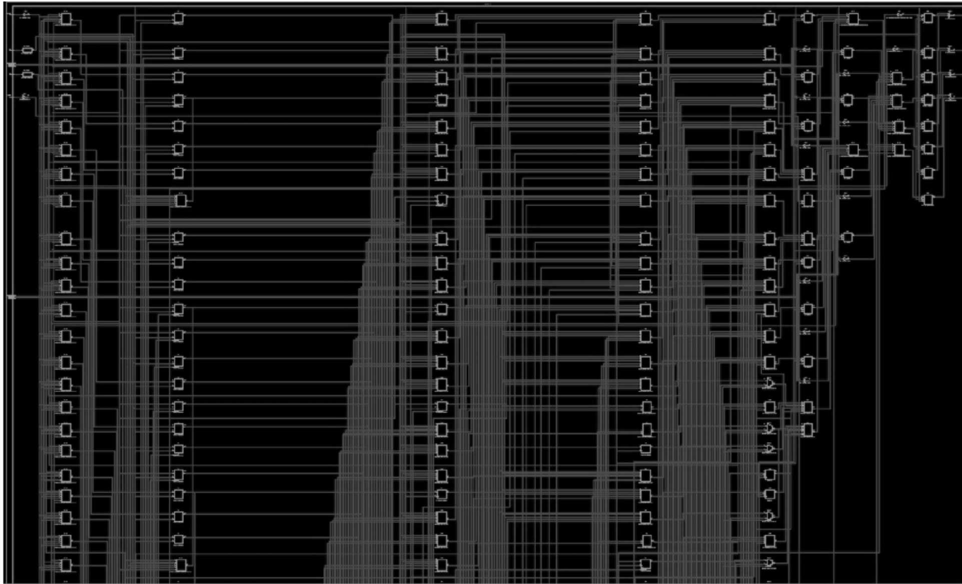

Figure 4: RTL Schematic of MBIST with 5-bit Address generator

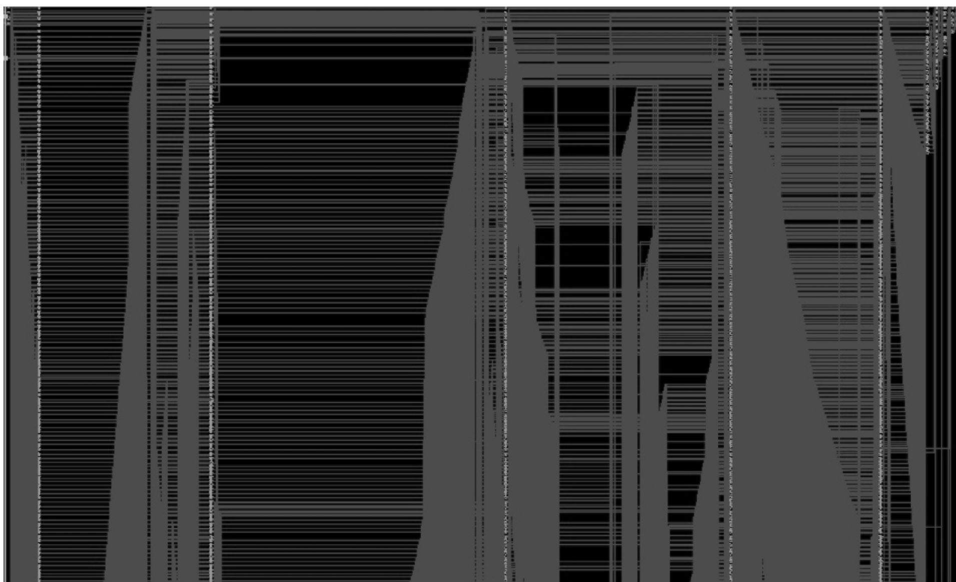

Figure 5: RTL Schematic of MBIST with 10-bit Address generator

The implemented design is simulated using ISE Simulator tool in the Xilinx software. Simulations are performed separately for the implemented MBIST module with and without fault injection with 5-bit and 10-bit address generator. The simulation results are shown in figures 6-9 below.

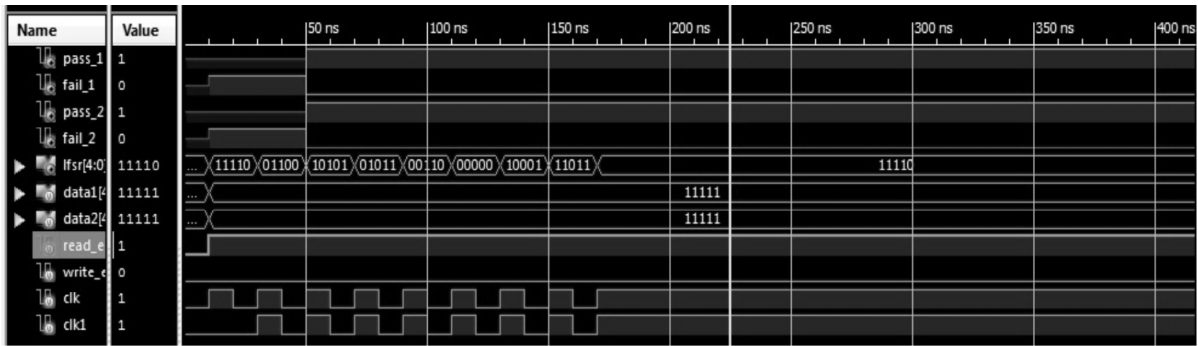

Figure 6: Simulation without fault of MBIST with 5-bit Address generator

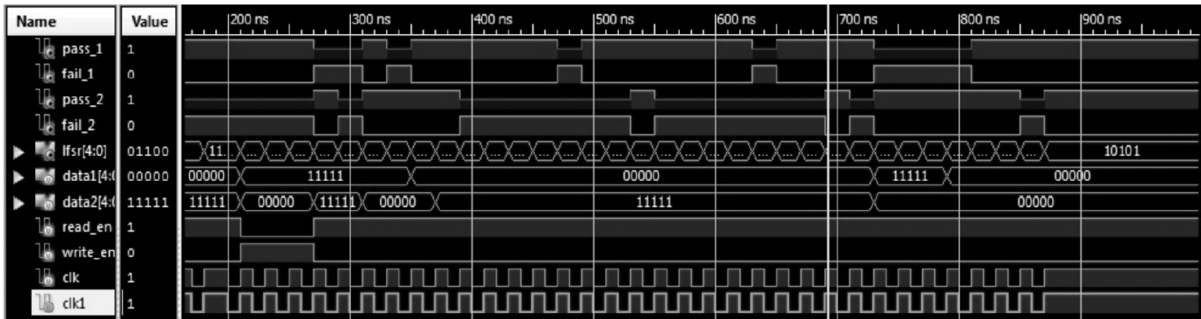

Figure 7: Simulation with fault of MBIST with 5-bit Address generator

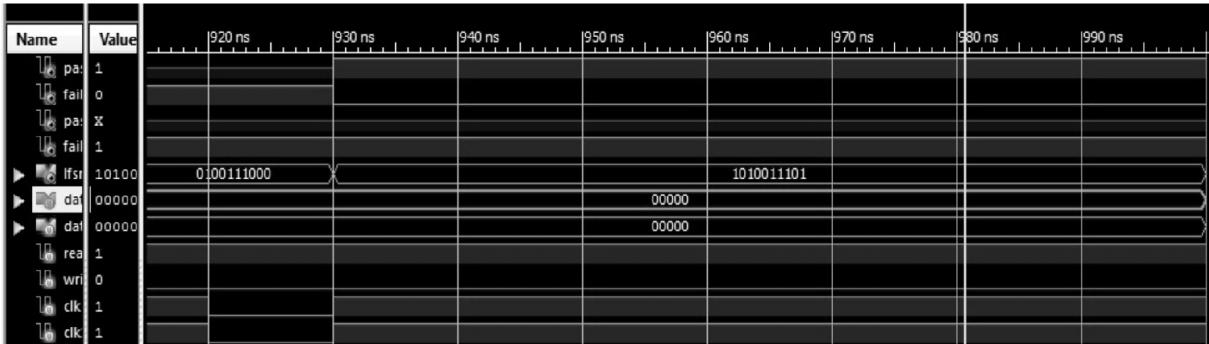

Figure 8: Simulation without fault of MBIST with 10-bit Address generator

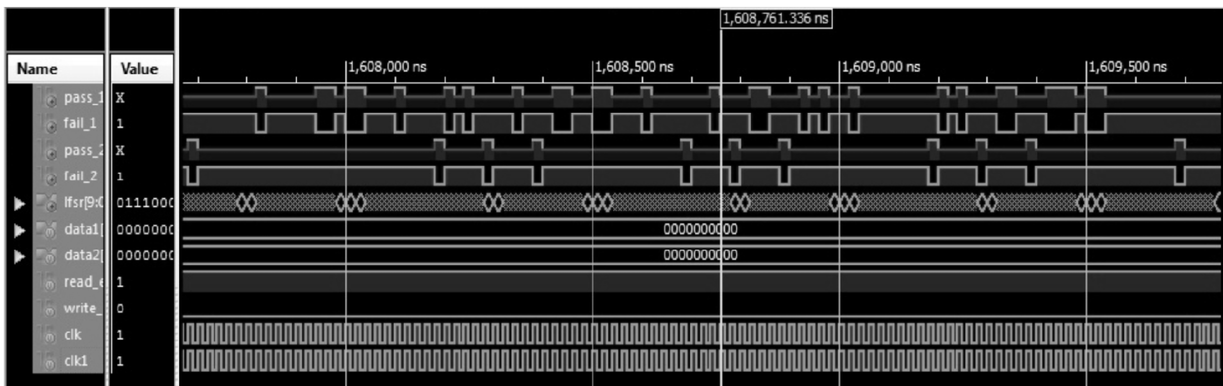

Figure 9: Simulation with fault of MBIST with 10-bit Address generator

Area

Our target device includes 3,120 Slices (12,480 Slice LUTs and 12,480 Sliced Registers) and 172 bonded IOBs. Each slice contains 2 flip-flops (FFs) and 2 look-up tables (LUTs).

Table 5
Area report of MBIST with 5-bit Address generator

Device Utilization Summary			
Slice Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	192	12,480	1%
Number used as Flip Flops	192		
Number of Slice LUTs	340	12,480	2%
Number used as logic	340	12,480	2%
Number using O6 output only	337		
Number using O5 and O6	3		
Number of occupied Slices	108	3,120	3%
Number of LUT Flip Flop pairs used	347		
Number with an unused Flip Flop	155	347	44%
Number with an unused LUT	7	347	2%
Number of fully used LUT-FF pairs	185	347	53%
Number of unique control sets	7		
Number of slice register sites lost to control set restrictions	12	12,480	1%
Number of bonded IOBs	23	172	13%
IOB Flip Flops	4		
Number of BUFG/BUFGCTRLs	2	32	6%
Number used as BUFGs	2		
Average Fanout of Non-Clock Nets	5.88		

Design synthesis of MBIST with 5-bit Address generator reports tabulated in table 5 shows that total of 108 slices, 192 slice fliflops, 340 LUTs and 23 bonded IOBs have been used in the synthesis. In the case of MBIST with 10-bit Address generator, the device utilization summary generated reported in table 6 shows that total of 896 slices, 1320 slice fliflops, 1629 LUTs and 28 bonded IOBs have been used in the synthesis.

Table 6
Area report of MBIST with 10-bit Address generator

Device Utilization Summary			
Slice Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	1,320	12,480	10%
Number used as Flip Flops	1,320		
Number of Slice LUTs	1,629	12,480	13%
Number used as logic	1,629	12,480	13%
Number using O6 output only	1,629		
Number of occupied Slices	876	3,120	28%
Number of LUT Flip Flop pairs used	2,249		
Number with an unused Flip Flop	929	2,249	41%
Number with an unused LUT	620	2,249	27%
Number of fully used LUT-FF pairs	700	2,249	31%
Number of unique control sets	261		
Number of slice register sites lost to control set restrictions	776	12,480	6%
Number of bonded IOBs	28	172	16%
Number of BUFG/BUFGCTRLs	2	32	6%
Number used as BUFGs	2		
Average Fanout of Non-Clock Nets	3.95		

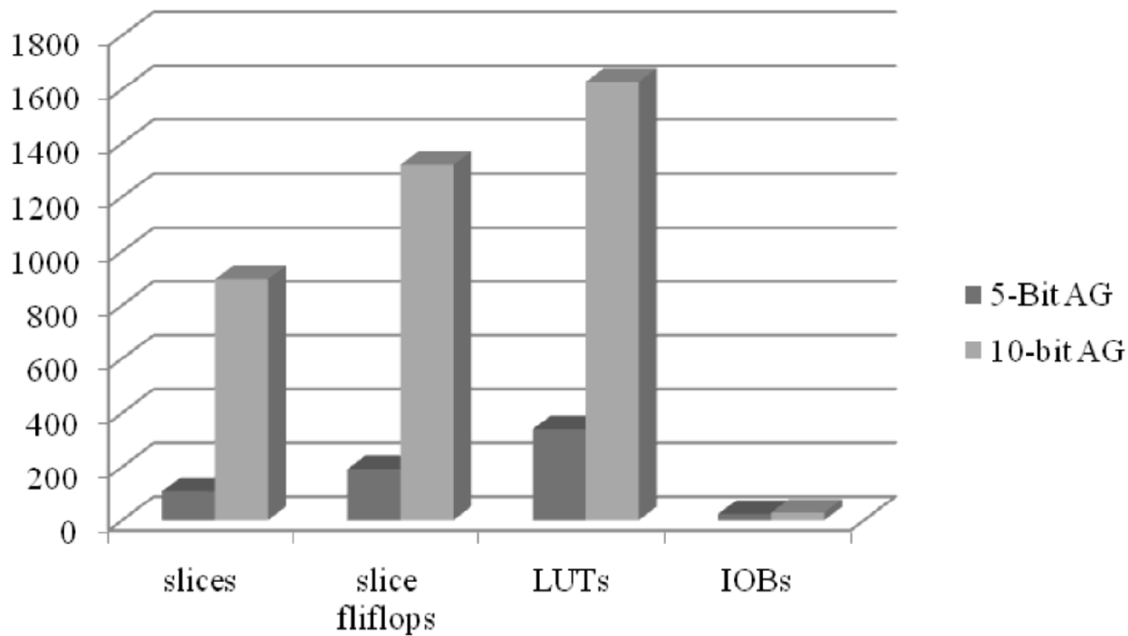

Figure 10: Area comparison of MBIST with 5-bit and 10 bit Address generator

In Figure 10 the bar-chart shows that with the increase in address bit length slice requirement increases, whereas the bonded IOBs remains almost the same.

Power and Performance

The power consumed by the implemented architecture is estimated by the Xpower analyzer tool available in the Xilinx software. The graph plot for the Dynamic power, static power and the total supply power with respect to the increase in frequency for the MBIST architecture with 5-bit and 10 bit address generator is shown in figure 11 and 12.

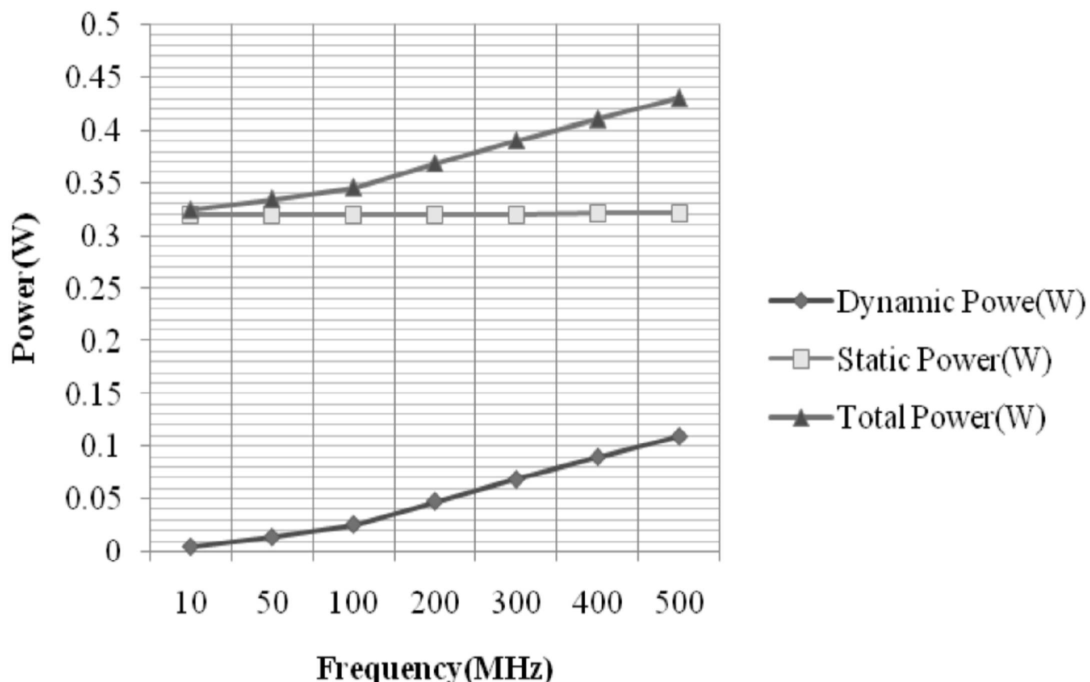

Figure 11: Power Vs Frequency graph of MBIST with 5-bit Address generator

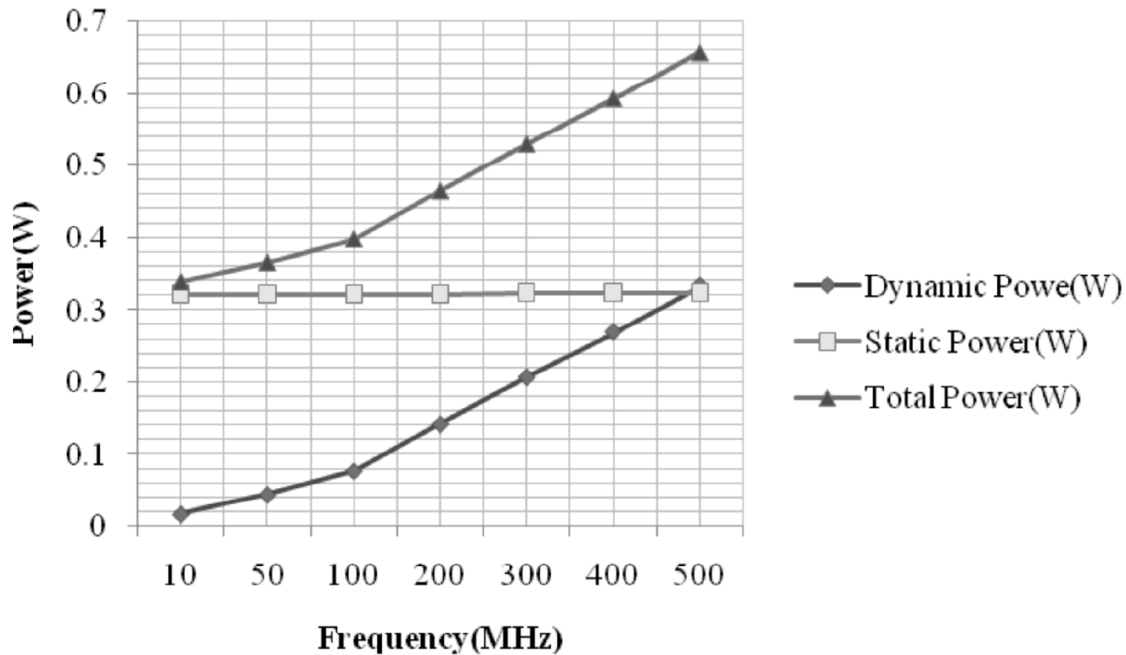

Figure 12: Power Vs Frequency graph of MBIST with 10-bit Address generator

It can be noted that with the increase in frequency the dynamic power increases slightly from 10 MHz to 100 MHz after which there is a steady increase in power. The static power remains the same with the increase in frequency and hence the total power depends directly on the dynamic power consumption. Table 7 highlights the timing summary of our implemented MBIST with 5-bit and 10 bit address generator.

Table 7
Timing summary

Parameter	*5-bit AG*	*10-bit AG*
Minimum period	3.344ns	4.527ns
Maximum Frequency	299.011 MHz	220.895MHz
Minimum input arrival time before clock	2.287ns	3.049ns
Maximum output required time after clock	2.844ns	2.849ns
Maximum combinational path delay	-	-

Table 8
Complexity and delay comparison

Algorithm	*Complexity*	*Delay*
Traditional C-[20]	10 N	13.782ns
Modified C-[20]	8 N	11.783ns
Proposed	5N	3.344/4.527ns

Our complexity and delay reports are compared with the Traditional-C and a Modified-C results reported in [20] .From table 8 comparison we can see that the complexity when compared with Traditional-C shows a 50% improvement and 37.5% when comparing with modified-C. The delay time comparison shows a reduction of approximately 71.44% and 66.6% respectively with the same.

6. CONCLUSION

In this work, VLSI Architecture for an Efficient Memory Built in Self Test for Configurable Embedded SRAM Memory was designed and implemented using Xilinx ISE software. The implemented MBIST architecture includes a modified SRAM structure, a modified address generator and comparators which operate based on the MBIST control signal with respect to a modified March Y algorithm. Area and power reports were analyzed with respect to different parameters. Experimental results confirmed that the MBIST architecture implemented in this work reported a reduction in complexity by about 50% and reduction in delay by about 70% when compared with existing similar works.

REFERENCE

[1] Kiran Gupta and T R.Gopalakrishnan.Nair , “A Study of VLSI Technology, Wafers and Impact on Nanotechnology”, IEEE National Conference on Advanced VLSI/DSP, Bangalore, 2008.

[2] David Choi, Kyu Choi, and John D. Villasenor, “New Non-Volatile Memory Structures for

[3] FPGA Architectures”, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 16, No. 7, July 2008.

[4] Technical report on “Non-Volatile Memory: Emerging Technologies And their Impacts on Memory Systems “Porto Alegre, Sep 2010.

[5] Venkatapathi Naidu Rayapati, “VLSI semiconductor random access memory functional testing”, Microelectronics Reliability, Vol.30, No 5, pp: 877–889, 1990.

[6] R. Dekker, F. Beenker, A. Thijssen, “Fault modeling and test algorithm development for static random access memories,” International Test Conference 1988.

[7] D. Niggemeyer, M. Redeker, J. Otterstedt, “Integration of nonclassical faults in standard march tests,” International Test Conference 1998, pp. 91-6.

[8] Luca Benini, Member, Alessandro Bogliolo, and Giovanni De Micheli, “A Survey of Design Techniques for System-Level Dynamic Power Management,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 8, No. 3, June 2000.

[9] E. J. Marinissen, B. Prince, D. Keitel-Schulz, Y. Zorian, “Challenges in Embedded Memory Design and Test”, Proc. of Design, Automation and Test in Europe, 2005, pp. 722 – 727.

[10] NurQ.Noor, YusrinaYusof, and AzilahSparon, “Low Area FSM-Based Memory BIST for Synchronous SRAM”, proceeding of the international colloquium of Signal Processing and Its application, 2009, pp. 409-412.

[11] K. Murali Krishna and M. Sailaja, “Low Power Memory Built in Self Test Address Generator Using Clock Controlled Linear Feedback Shift Registers”, Journal of Electronic Testing, Theory and Applications (JETTA), Vol.30, No.1, pp: 77-85, Feb.2014.

[12] Balwinder Singh, Arun Khosla and Sukhleen Bindra Narang, “Area Overhead and Power Analysis of March Algorithms for Memory BIST”, International Conference on Communication Technology and System Design, pp: 930 – 936, 2012.

[13] Che-Wei Chou ,Chih-Sheng Hou and Jin-Fu Li “Built-in self-diagnosis and test time reduction techniques for NAND flash memories” International Symposium on VLSI Design, Automation and Test (VLSI-DAT),April 2011.

[14] Jen-Chieh Yeh, Kuo-Liang Cheng, Cheng-Wen Wu and Yung-Fa Chou, “Flash Memory Testing and Built-In Self-Diagnosis with March-Like Test Algorithms”, IEEE Transactions on Computer-Aided Design Of Integrated Circuits and Systems, Vol. 26, No. 6, pp: 1101-1113, JUNE 2007.

[15] Masnita, Wan Zuha W.H, R.M. Sidek and A. Halin, “The data and read/write controller for March-based SRAM diagnostic algorithm MBIST”, proceedings of 2009 IEEE conference on Research and Development, Nov.2009.

[16] Manikandan.B and Praveen kumar.J, “A VLSI Implementation of High Speed FSM-based programmable Memory BIST Controller”, Journal of Electronics and Communication Engineering Research, Vol.1, No.2, pp: 01-06, 2013.

[17] K.Padma Priya, “High Speed FSM-based programmable Memory Built-In Self-Test (MBIST) Controller”, International Journal of Computer Science and Mobile Computing, Vol. 2, No. 2, pp:.46-52, February 2013.

[18] M. Jahnavi, P. S. Indrani, M. J. C. Prasad, “Implementation of Concurrent Online MBIST for RFID Memories using March SS Algorithm”, International Journal of Innovative Technology and Exploring Engineering (IJITEE), Vol.3, No.6, November 2013.

[19] Nitin Sharma, Dr. Pradeep Kumar, Mr. Sunil Kumar, “Optimizing March Algorithm For Testing Embedded Memory: A Case Study,” International Journal of Advance Research In Science And Engineering, IJARSE, Vol.2, No.6, June, 2013.