Indian Journal of Science and Technology, Vol 12(36), DOI: 10.17485/ijst/2019/v12i36/147752, September 2019
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Performance Analysis of BIST Algorithms

Nongthombam Imocha Singh* **and Prashant V. Joshi**

School of ECE, REVA University, Yelahanka, Bangalore – 560064, Karnataka, India; imochamm@yahoo.com, prashanthvjoshi@reva.edu.in

Abstract

Objective: Performance analyses of March LR and March C- are presented to achieve high fault coverage, low power dissipation, less area utilization and minimum testing time. **Methods**: Testing of memory consist of Built-In-Self-Test (BIST) controller and circuit-under-test. BIST algorithms are resided inside the BIST controller. BIST algorithms such as March LR and March C- are coded in term of finite state machine. Memory is modeled in verilog and simulated in ModelSims for testing memory faults and it is synthesized by using Xilinx Vivado 2012.2 EDA tool for power, area and timing analysis. **Findings**: Memory tests are conducted to verify the correctness of each memory location. It involves writing a particular set of data to each memory location and checking that data by reading it. After reading back, if the values are same as those of writing values, then the test is past. If not, the test is fail. Various test methodologies have been developed to detect the memory defects. One such test is BIST technique. Before implementing BIST algorithm, it is necessary to study the various functional faults models for memory defects. The commonly occurs faults are Stuck-At Fault (SAF), Stuck Open Fault (SOF), Transition Fault (TF), Coupling Fault (CF) etc. These functional faults are model and write the code in verilog. Faults are inserted and detected using ModelSims. Using Xilinx Vivado 2012.2 EDA tool, power, area and timing are analyzed. Comparison is made between March C- and march LR. Even though March LR has more length (14N), it has high fault coverage and lesser power dissipation. Thus, March LR is more efficient than March C-. **Applications/Improvements**: The fault coverage is improved by using March LR. More improvement in March LR can detect more faults which result in the efficiency for online BIST

Keywords: BIST Algorithm, Built-In-Self-Test (BIST), March C- and March LR, Memory Testing

1. Introduction

Testing of memory is required to verify the correction of its hardware. The rapid growth in the field of semiconductor leads to increase the complexity of the memory structure exponentially[1,2]. The testing done by using Automatic Test Equipment (ATE) takes longer time and is expensive. To solve these challenges, technique known as Built-In-Self-Test (BIST) is introduced. In a study by[3], "BIST is a design-for-test technique in which testing is accomplished through built-in-hardware features". In BIST method, extra components are implemented on the chip for testing itself. Test Pattern Generator (TPG), Circuit-Under-Test (CUT) and Output Response Analyzer (ORA) are the main components of basic BIST module. Figure 1 shows the simple BIST block diagram. TPG produces test pattern for locating particular address on CUT. The output data will be collected and verify by ORA from CUT which produces the result whether it is pass or fail. The whole operation is controlled by BIST Controller when the memory in testing mode.

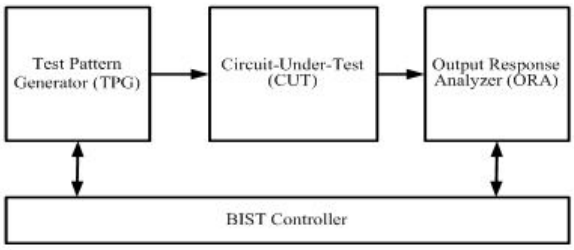

Figure 1. Basic architecture of BIST.

*Author for correspondence

2. BIST Algorithm for Memory Testing

Memory tests are conducted to verify the correctness of each memory location. It involves writing a particular set of data to each memory location and checking that data by reading it. After reading back, if the values are same as those of writing values, then the test is past. If not, the test is fail. Various test methodologies have been developed to detect the memory defects. One such test is BIST technique.

Before implementing BIST algorithm, it is necessary to study the various functional faults models for memory defects. The commonly occurs faults are as follow:

1. Stuck-At Fault (SAF): Logical value a cell is always at either 'high (1)' or 'low (0)'.
2. Stuck Open Fault (SOF): SOF occurs when a particular cell may not be accessed because of some physical defect.
3. Transition Fault (TF): A particular cell fails to change the logical value of that cell from high to low or low to high.
4. Coupling fault (CF): Coupling fault occurs when there is change in logical value of a particular cell due to write operation of another cell. It can be inversion Coupling Fault (CFin), idempotent Coupling Fault (CFid), and state Coupling Fault (CFsts).

There are so many memory faults are left to mention but for our performance analysis we are considering the above faults only.

Many BIST algorithms have been developed so far. Among them, March algorithms are commonly popular. There are different types of March algorithms such as March C, March C-, March X, March LR etc. March algorithm has March element and march primitives[4,5]. March element may be operated in increasing, decreasing or either increasing or decreasing order depending of the algorithm. The operation may be writing operation or reading operation. Table 1 shows the various March Algorithms.

Where (↑) denotes increasing order of address, (↓) denotes decreasing order of address, (↕) denotes either increasing or decreasing order of address, (wb0 or wb1) indicates either writing a zero or one to a cell respectively, (rb0 or rb1) indicates either reading a zero or one from a cell respectively, N indicates no. of cells in the memory, AF indicates address fault, SAF indicates stuck-at fault, TF indicates transition fault, CF indicates coupling fault, CFid indicates idempotent coupling fault.

March based algorithm test gives high coverage faults and the testing time is usually linear with the size of the memory. In this paper, March C- and March LR will be analyzed.

3. Performance Analysis of BIST Algorithms

The March C- and March LR algorithms are considered for their performance analyses. For that 8 × 8 bit of memory is modeled in verilog and then simulated using ModelSim. Figure 2 & 3 shows the fault finding simulation for March LR algorithm. Fault finding simulation for March C- is also done in similar way.

For power, area and timing analysis, 1024x8 bit of memory is modeled using verilog and synthesized by using Xilinx Vivado 2012.2 EDA tool. The target device used for implementation is Virtex 7, XC7VX485TFFG1157-1. The power, area and maximum frequency reports are shown in Table 2.

Table 1. March algorithms

Sl. No.	Algorithm	Length	March elements	Faults coverage
1.	MATS++	5N	{↕(w0); ↑ (r0, w1); ↓(r1, w0)}	AF, SAF
2.	March X	6N	{↕(w0); ↑ (r0, w1); ↓ (r1, w0); ↕(r0)}	AF, SAF, TF, some CFs
3.	March C-	10N	{↕(w0); ↑(r0, w1); ↑(r1, w0); ↓(r0, w1); ↓(r1, w0); ↕(r0)}	AF, SAF, TF, CF, some CFs
4.	March B	17N	{↕(w0); ↑(r0, w1, r1, w0, r0, w1); ↑(r1, w0, w1); ↓(r1, w0, w1, w0); ↓(r0, w1, w0)}	AF, SAF, TF, CF, linked CFid
5.	March LR	14N	{↕(w0); ↓(r0, w1); ↑(r1, w0, r0, w1); ↑(r1, w0); ↑(r0, w1, r1, w0); ↑(r0)}	AF, SAF, TF, linked CF

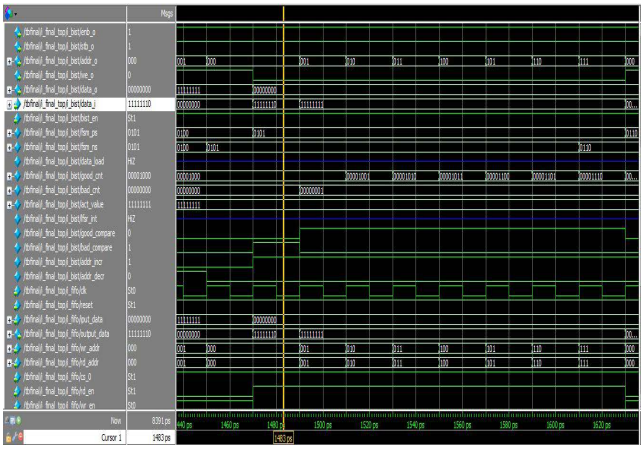

Figure 2. SAF0 in the memory.

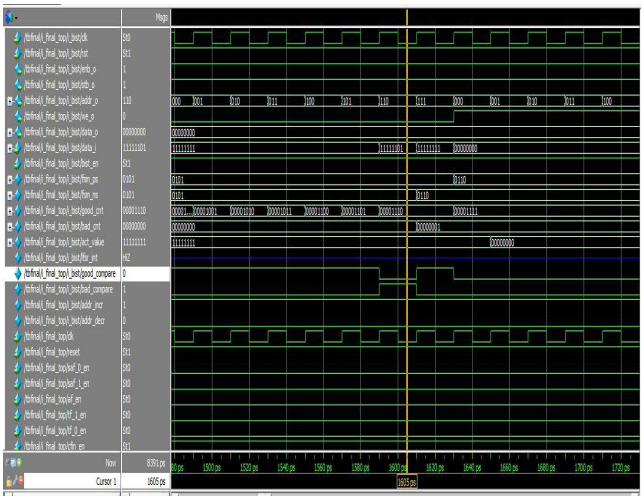

Figure 3. CF in the memory.

Table 2. Synthesis report

	March C-	March LR
Power (mw @ 25.3°C)	305	211
Area (no. of LUT)	266	260
Fmax (MHz)	194	180

March LR is much more efficient than the March C- as March LR used lesser power and lesser area utilization compared to that of March C-. But March C- has complexity of 10N where as March LR has complexity of 14N. So, March C- has lesser complexity.

4. Conclusion

This study is focused on the performance analysis of March algorithm such as March C- and March LR. It is concluded that March LR is more efficient than March C-. By modifying March LR algorithm, more BIST algorithm may be design for testing the memory in future.

5. References

1. Ying JC, Tseng WD, Tsai WJ. Asymmetry dual-LFSR reseeding for low power BIST, Integration - The VLSI Journal, Elsevier. 2018; 60:272–76. https://doi.org/10.1016/j.vlsi.2017.10.012.
2. The International Technology Roadmap for Semiconductors. Date accessed: 2011. http://www.maltiel-consulting.com/ITRS-2011-Executive-Summary.pdf.
3. Agrawal V, Kime C, Saluja K. A tutorial on built-in self test. In: IEEE Design and Test of Computers. IEEE Computer Soc. Pressand Silver Spring, MD; 1993. p. 73–80. https://doi.org/10.1109/54.199807.
4. Singh B, Narang SB, Khosla A. Modeling and Simulation of Efficient March Algorithm for Memory Testing. In International Conference on Contemporary Computing. Springer Berlin Heidelberg; 2010. p. 96–107. https://doi.org/10.1007/978-3-642-14825-5_9.
5. Parvathi M, Vasantha N, Satya Parasad K. Modified March C- Algorithm for embedded memory testing, International Journal of Electrical and Computer Engineering. 2016; 2:571–76. https://doi.org/10.11591/ijece.v2i5.1587.