ISSN 2350 - 0328

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 3, Issue 5, May 2016

A NOVEL APPROACH IN MEMORY BIST USING MODIFIED MARCH C ALGORITHM IN SRAM BASED FPGA

R.Karthick, Dr.V.Saminadan

Research Scholar, Department of ECE, Pondicherry Engineering College, Puducherry.

Professor, Department of ECE, Pondicherry Engineering College, Puducherry

ABSTRACT: The final DFT approach is BIST, which incorporates test pattern generation and output response analysis capabilities in the chip (or on the PCB) itself. Other than initiating the BIST sequence, and reading the BIST results at the end of the sequence, there is no need for external manipulation or monitoring of the device during the testing sequence, as is the case in all of the other DFT techniques. While this external intervention for the other DFT approaches prevents testing at the system operating frequency in many cases, BIST can easily facilitate testing at system speed (at-speed testing) if implemented properly. BIST can also be used at all levels of testing, from wafer-level to system-level (vertical testability). These are the main advantages BIST has over the other DFT techniques. March algorithms are known for memory testing because March-based tests are all simple and possess good fault coverage hence they are the dominant test algorithms implemented in most modern memory BIST. The proposed march algorithm is modified march c- algorithm which uses concurrent technique. Using this modified march c- algorithm the complexity is reduced to 8n as well as the test time is reduced greatly. Because of concurrency in testing the sequences the test results were observed in less time than the traditional March tests. This technique is applied for a memory of size 256x8 and can be extended to any memory size.

KEYWORDS: Memory BIST, March Algorithm, BIST, Concurrent technique.

I. INTRODUCTION

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 [1-4]. 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.The advantage of March tests lay in the fact that high fault coverage can be obtained and the test time were usually linear with the size of the memory which makes it acceptable from industrial point of view.March based algorithms were capable of locating and identifying the fault types which can help to catch design and manufacturing errors. Especially SAF dominate the majority of defects that occur in embedded RAMS.

The method proposed in this paper is Modified March C- algorithm with concurrent technique. This algorithm retains the high fault coverage of March C but at reduced time the tests can be done. The paper further describes the functional fault models in the memory, classical and March based tests in section II. The proposed Modified March c- algorithm was discussed in section III. Results and comparisons were discussed in section IV. Conclusions were given in section V.

II.HISTORY OF FUNCTIONAL FAULT MODELS

For testing purpose the functional fault models are modeled after faults in memories so that functional tests to detect these faults can be used. This modeling helps to clarify, simplify and generalize the testing approach of a memory. The quality of tests is strongly dependent on the fault model in terms of its fault coverage, its test length as well as the test time required.

There are various fault models to test the functional faults such as stuck at faults; coupling faults are considered when it deals with SRAM. Address decoder faults and bridging faults will be considered when it deals with DRAM. Hence the most possible faults which occur in general are stuck at faults .*Stuck at fault (SAF)* : The stuck-at fault (SAF) considers that the logic value of a cell or line is always 0 (stuck-at 0 or SA0) or always 1 (stuck-at 1 or SA1). To detect and locate all stuck-at faults, a test must satisfy the following requirement: from each cell, a 0 and a 1 must be read.*Transition Faults(TF)*: The transition fault (TF) is a special case of the SAF. A cell or line that fails to undergo a 0 → 1 transition after a write operation is said to contain an up transition fault. Similarly, a down transition fault indicates the failure of making 1 → 0 transitions. According to van de Goor [8, 9], a test to detect and locate all the transition faults should satisfy the following requirement: each cell must undergo an ↑ transition (cell goes from 0 to 1) and a ↓ transition (cell goes from 1 to 0) and be read after each transition before undergoing any further transitions.The fault detection for both SAFs and TFs will be done by considering MATS++ algorithm and March C- algorithm. Although different in test length, these tests are capable of detecting both faults while being capable of detecting other faults as well. The detection process can be understood by examining the Mach C- algorithm as indicated in expression below.*March Test Notation*: A March test consists of a finite sequence of March elements [10-12]. A March element is a finite sequence of operations or primitives applied to every memory cell before proceeding to next cell. For example, ↓(r1, w0) is a March element and r0 is a March primitive. The address order in a March element can be increasing (↑), decreasing (↓), or either increasing or decreasing (↕). An operation can be either writing a 0 or 1 into a cell (w0 or w1), or reading a 0 or 1 from a cell (r0 or r1).

Accordingly notation of March C- test is described as follows:
{↕(w0) ; ↑ (r0, w1) ; ↑ (r1, w0) ; ↓ (r0, w1) ; ↓ (r1, w0) ;↕(r0)}

•   _ _ _ _ _ _
M$_{I}$ M$_{II}$ M$_{III}$ M$_{IV}$ M$_{V}$ M$_{VI}$

March C- algorithm has 6 elements as shown with a complexity of 10n.

III. MODIFIED MARCH C- ALGORITHM

The proposed Modified March C- algorithm almost similar to March C- but it follows concurrency in testing the sequences. The steps for following the concurrency are as follows:•Group entire memory into subgroups. •For each subgroup, generate all test vectors for the first fault in the group.Simulate all faults in the subgroup to select one vector that detects most faults in that subgroup. If more vectors than one detect the same number of faults within the group, then select the vector that detects most faults outside the group as well. •Apply the final test vectors to all subgroups concurrently

In the proposed method the memory is divided in to two subgroups such as M1 and M2. Then applied the algorithm for concurrency. The following are the elements in Modified March C- algorithm.

M1: { ↑ (w0) ; ↑ (r0, w1) ; ↑ (r1) ; ↓ (w0) ; ↓ (r0, w1) ; ↓ (r1) ;
M2: { ↑ (w1) ; ↑ (r1, w0) ; ↑ (r0) ; ↓ (w1) ; ↓ (r1, w0) ; ↓ (r0) ;

The number of March elements is same as traditional March c- and is 6 but because of concurrency the complexity is reduced to 8n.

The pseudo code for modified march c- is as follows:

```
//for writing 0s in block 1 and writing 1s in block 2, let n and m are rows and columns
for(i=0;i<(n-1)/2;i=i+1)
begin
for(j=0;j<(m-1);j=j+1)
mem[i][j]=0;//write 0 in m1
end
for(i=(n-1)/2;i<(n-1);i=i+1)
begin
for(j=0;j<(m-1);j=j+1)
mem[i][j]=1;//write 1 in m2
end
//for reading background and for writing alternate
for(i=0;i<(n-1)/2;i=i+1)
begin
for(j=0;j<(m-1);j=j+1)
begin
if(mem[i][j]==0)
mem[i][j]=1;
else return;
end
end
for(i=(n-1)/2;i<(n-1);i=i+1)
begin
for(j=0;j<(m-1);j=j+1)
begin
if(mem[i][j]==1)
mem[i][j]=0;
else return;
end
end
```

According to Modified March C- elements, when 0s are written in one memory group, 1s will be written in another group concurrently. So the test sequence can be taken through an inverter hence true form will goes to one block of memory and complement form will goes to another block of memory. Hence the test sequence generator requires additionally one inverter in order to perform test concurrently. The method directly reduces the time required to write and read the bit concurrently. This reduces the test time and test costs also. Finally, there may be additional design cost in terms of inverter only which need to generate complement test sequence to other part of the memory block.

IV. RESULTS AND COMPARISIONS

Table 1 indicates delay performance for each element present in traditional March C- algorithm given for fault free ondition and faulty condition. Under faulty condition using SA fault models the overall delay observed as 13.782ns.
Table 2 shows the delay performance using Modified March C- algorithm. In this also delay performance were alculated separately for fault free as well as faulty conditions. Under faulty condition the overall delay was observed as 11.784ns. Hence it is proved that using Modified March C- algorithm using concurrency the overall delay is greatly reducing. It is giving at speed test performance than any other traditional algorithm. The result tables also provide the information on minimum input arrival time before clock and maximum output time after the clock. Simulation was

carried using Xilinx ISE 10.1i tool for the device XC3S4004tq144 and tested on Spartan 3 kit. Fig 1 and 2 shows the simulation results respectively for modified march elements I and II when fault is imposed.

Table 1. Results for Traditional March C- Algorithm

MARCH ELEMENT	MINIMUM PERIOD IN NANO SEC		MINIMUM INPUT ARRIVAL TIME BEFORE CLOCK IN NANO SEC		MAXIMUM OUTPUT REQUIRED TIME AFTER CLOCK IN NANO SEC	
	WITH NO FAULT	WITH FAULT	WITH NO FAULT	WIYH FAULT	WITH NO FAULT	WIYH FAULT
M_{I} : ↕(w0)	1.483	2.075	3.439	4.033	6..314	6..28
M_{II}:↑(R0,w1)	1.585	2.085	3..504	3..529	6..318	6..314
M_{III}: ↑(R1,w0)	1.585	2.085	3.504	3.529	6.318	6.314
M_{IV}: ↓(R0,w1)	1.585	2.085	3.504	3.529	6.318	6.314
M_{V}: ↓(R1,w0)	1.585	2.085	3.504	3.529	6.318	6.314
M_{VI}: ↕(R0)	2.196	3.367	3.955	4.170	6..318	6..3

Table 2. Results for Modified March C- Algorithm

MARCH ELEMENT	MINIMUM PERIOD IN NANO SEC		MINIMUM INPUT ARRIVAL TIME BEFORE CLOCK IN NANO SEC		MAXIMUM OUTPUT REQUIRED TIME AFTER CLOCK IN NANO SEC	
	WITH NO FAULT	WITH FAULT	WITH NO FAULT	WIYH FAULT	WITH NO FAULT	WIYH FAULT
M1: ↑(w0) M2:↑(w1)	1.483	2.111	3.439	3.473	6..31	6..28
M1:↑(R0,w1) M2:↑(R1,w0)	2.132	2.196	4.755	3..979	6..28	6..441
M1: ↑(R1) M2:↑(R0)	2.132	1.585	3.96	3.50	6.28	6.318
M1: ↑(w1) M2:↑(w0)	1.483	2.111	3.439	3.473	6.31	6.28
M1:↓(R0,w1) M2:↓(R1,w0)	2.132	2.196	4.755	3..979	6.28	6.441
M1: ↓(R1) M2:↓(R0)	2.132	1.585	3.96	3.50	6.28	6.314

Table 3. Comparison

TYPE OF ALGORITHM USED	COMPLEXITY	DELAY (NANO SEC)
TRADITIONAL MARCH C-	10N	13.782
MODIFIED MARCH C-	8N	11.783

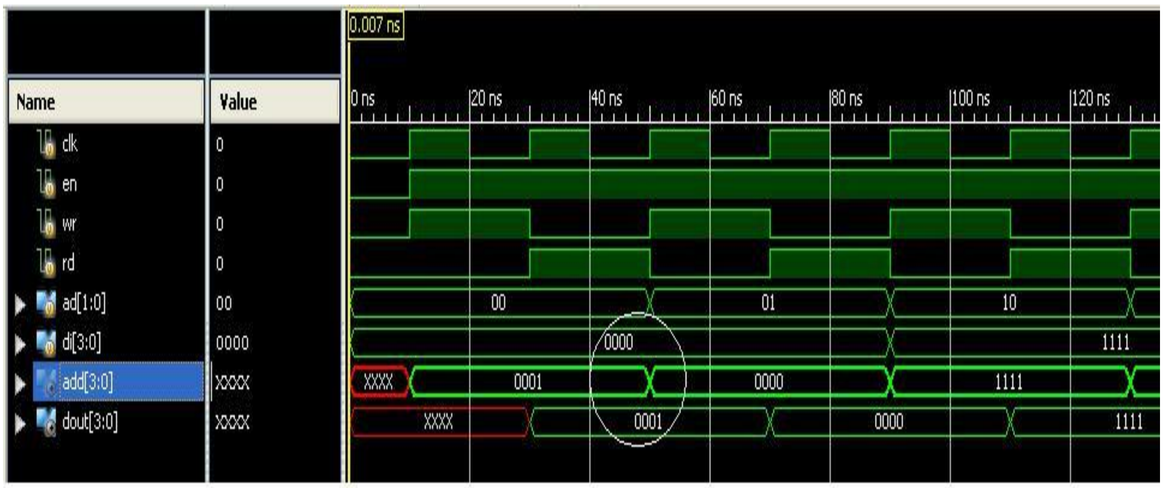

Fig.1 Modified March C Element I

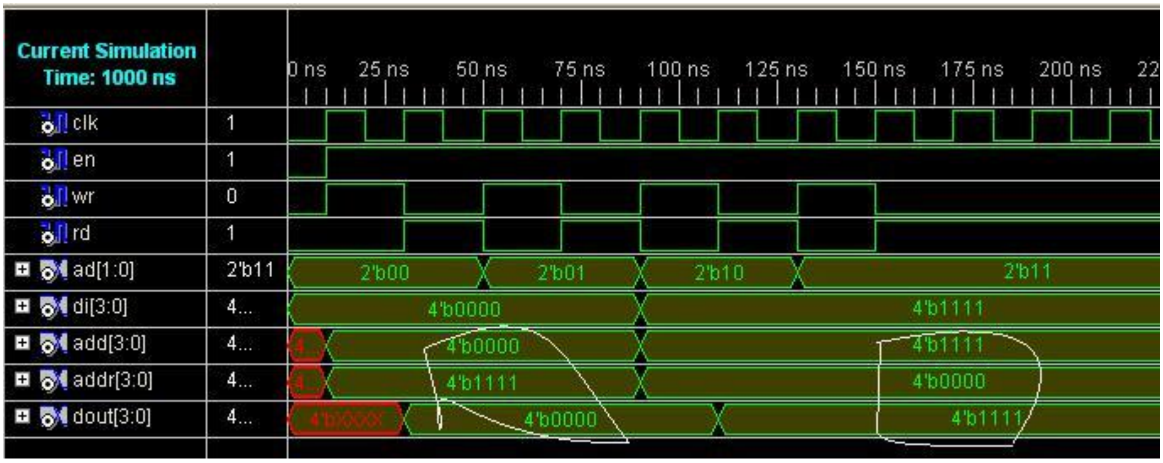

Fig.2 Modified March C Element II

V. CONCLUSION

This paper defines the functional fault model and compared the traditional march c- algorithm with modified march c- algorithm in terms of speed of the test sequence and complexity of the number of test sequences. The crucial part in testing is how well the test can be completed in minimum time with minimal test length. The modified march algorithm has proved that the test length is minimal as well as the time required to test SAF also minimum when compared with traditional march c-. Hence this modified march c is much comparable and could be used for detection of various faults other than SAF as future work in SRAM based FPGA.

REFERENCES

[1] Shibaji Banerjee, Dipanwita Roy Chowdhury and Bhargab B. Bhattacharya, in proceedings of the 2005 IEEE International Workshop on Memory Technology, Desgin and Testing (MTDT'05).
[2] The national Roadmap for Semiconductors, 2000. Semiconductor Industry Association.
[3] E Eric Dupont, Michael Nicolaidis, and Peter Rohr, " Embedded Robustness IPs forTransient Error free ICs", IEEE
Design & Test of Computers, 2002.
[4] Dupont, E. Nicolaidis, M. And Rohr .P. " Embedded Robustness IPs for Transient Error Free ICs" IEEE Design & Test of Computers, Vol 19, No 3, pp 56-70, May-June, 2002.
[5] J.F.Li, K.L.Cheng, C.T.Huang, and C.W.Wu, "March based RAM diagnostic algorithms for stuck-at and coupling faults",Proc,IEEE ITC pp,758-767,2001
[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
[7] R. Dekker, F. Beenaker, L.Thijssen, "A realistic fault model and test algorithm for static random access memories",IEEE Trans. Computer-Aided Design, vol,9,99.567-572, june 1990.
[8] J. van de Goor. Testing Semiconductor Memories: Theory and Practice.A.J.vandeGoor,1998.
[9] Verilog Digital System Desigh, 2nd Edition, Zainalabedin Navabi, Tata McGraw Hill, 2008
[10] A.J.Van de Goor, Testing Semiconductor Memories, Theory and Practice, John Wiley & Sons, Chichester, England,1991.
[11] Van De Goor, A.J. Sehanstra. I., "Address and Data Scrambling: Causes and Impact on Memory Tests", IEEE DELTA Workshop, Christchurch, January 2002.
[12] ER.Manoj Arora, Er.Shipra Tripathi, " Comparative Simulation of MBIST using March Test Algorithms", International Journal of Scientific & Engineering Research, Volume 2, Issue 12, December-2011.