International Journal of Emerging Engineering Research and Technology
Volume. 2, Issue 6, September 2014, PP 81-89
ISSN 2349-4395 (Print) & ISSN 2349-4409 (Online)

Design of Microcode Based Memory Built in Self Test

M Sarika

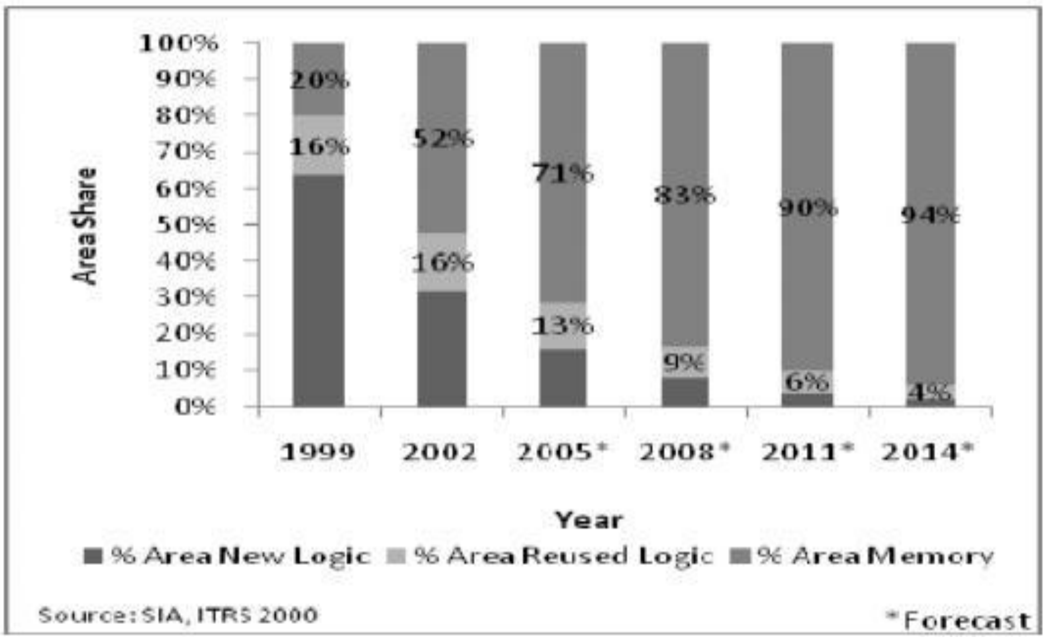
M.Tech-VLSI, GRIET

Abstract: *It becomes highly important to test various kinds of defects rapidly and precisely to reduce the testing cost and to improve the memory quality especially under the SOC design environment. Memory defects can be modelled as struck-at, coupling transition, address decoder and pattern sensitive faults and it is known that the 80% of the failures are due to leakage defects[1]. Memory test patterns can be generated deterministically or randomly[3] through either a test equipment (or) a BIST circuitary. A new microcoded based BIST circuitary for embedded memory components is proposed in this paper. A microcoded BIST architecture which can implement these new March tests having number of operations per element according to the needs of embedded memory testing. This is shown by implementing March SS algorithm and testing for new faults including Write disturb fault[WDF], Transition coupling faults[CFT], which established tests like March C are not capable of detection. Verilog HDL code of this architecture is written and synthesized using Xilinx ISE 8.2i. verification of architecture is done by testing Modelsim PE 10.3a as a simulation tool.*

Keywords: *Built –In self Tes(BIST),embedded memory fault, Memory Bult-in Self Test(MBIST:, Microcoded MBIST; MUT(Memory Under Test)*

1. Introduction

As embedded memory area of System- on-chip (SoC) is increasing in exponentially and memory density also increasing, problem of faults is growing exponentially.Newer test algorithms are formulated for detecting these new faults. These new March algorithms have much more number of procedures than the March algorithms existing earlier. An architecture applying these new algorithms is presented here. This is instanced by implementing the newly defined March SS algorithm According to the 2001 ITRS, today's embedded dominating memory area of System- on- Chips (SoCs) are increasing .The dominating logic (about 64% in 1999) is changing to dominating memory (approaching 90% by 2011 and 94% by 2014) as shown in Fig 1 below.



[6]Fig1. *The future of Embedded Memory*

The new tendencies in memory testing will be driven by the following fault models.

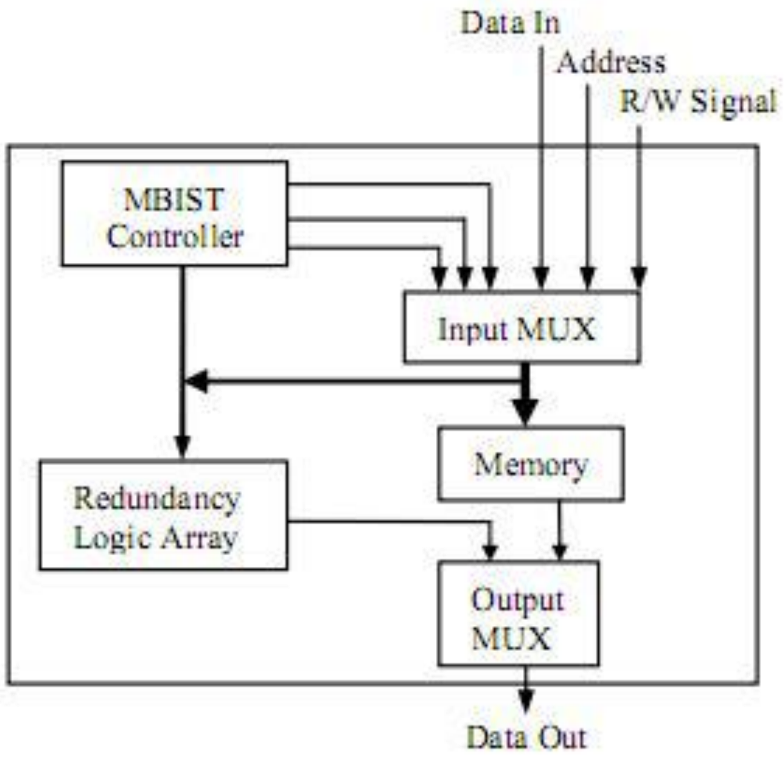
Fault modeling: This fault models should be established in order to deal with the new defects introduced by current and future technologies.

Test algorithms: Optimal test/diagnosis algorithms to guarantee high defect coverage for the new memory technologies and reduce the DPM level.

BIST: It is uses at high speed testing for detect the faults in embedded memory .This only solution that allows at speed testing for embedded memories.

BISR: Combining BIST with efficient and low cost repair schemes in order to improve the yield and system reliability as well.

March SS **[5]** and March RAW **[3]** are examples of two such newly developed test algorithms that deal with detecting some recently developed static and dynamic fault models. A new microcoded BIST architecture is presented here which is capable of employing these new test algorithms. A word-oriented BISR array is used to repair the faulty memory locations as indicated by the BIST controller. The interface of repair array with BIST controller and Memory under test is shown in Fig. 2.



[6] Fig2. *Block Diagram of BISR*

2. ARCHITECTURE OF BISR

The architecture of BISR which have been developed to implement earlier tests like March C- may not be able to easily implement these newer test algorithms. The reason is that most of the newly formulated algorithms have up to six or seven (or even more) number of test operations per test element. For example test components M1 through M4 of March SS algorithm have five test operations per element. This is in distinction with some of the algorithms developed earlier like March B, MATS+, March C which only had up to two operations per March element. Thus some of the recently developed architectures [6] older algorithms can only implement up to two march operations per march element, delivering them incapable of easily implementing the new test algorithms.

March algorithms can be successfully implemented and applied using this architecture. This has been illustrated in the present work by implementing March SS algorithm. The same hardware has also been used to implement other new March algorithms. This requires just changing the Instruction storage unit, or the instruction codes and sequence inside the instruction storage unit. The instruction storage is used to store predetermined test pattern. The architecture of the micro based built in self test and repair is shown in the Fig.3 below,

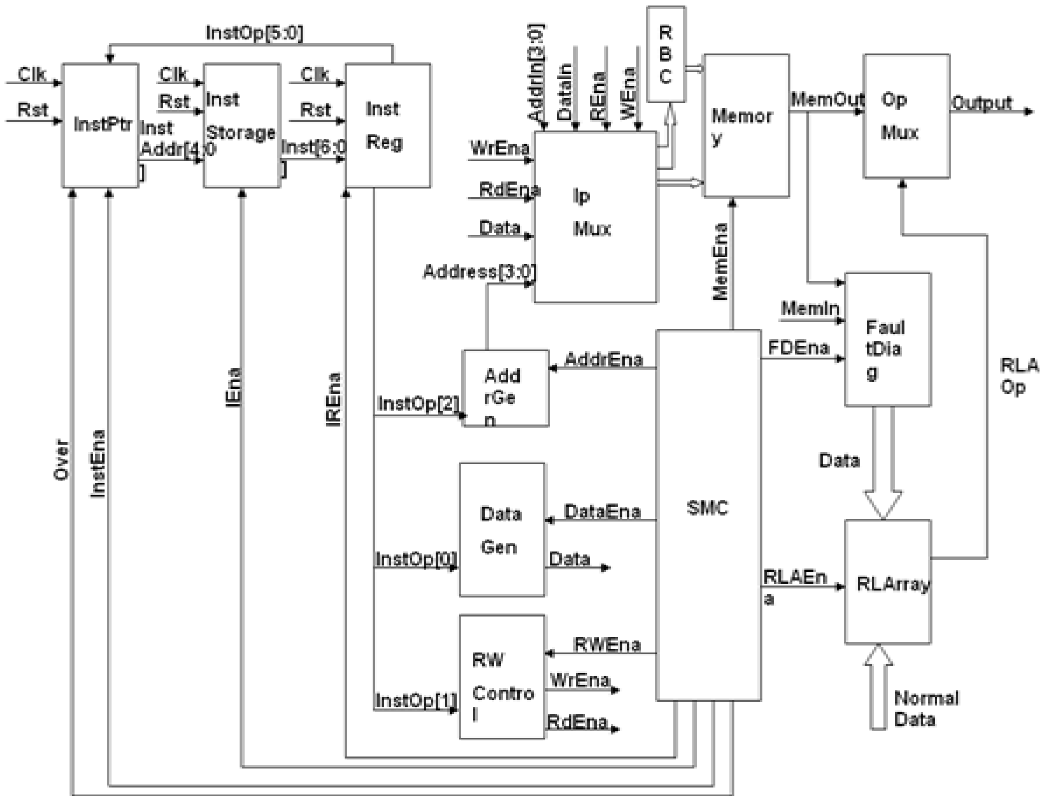


Fig3. *Architecture of Built in Self Test and Repair*

The block diagram of the BIST controller architecture together with fault diagnosis interface through input MUX shown in above Fig.3.

It consists of the Instruction Pointer, Instruction storage, Instruction register, AddrGen, DataGen , DataGen ,RWControl ,Input multiplexer, memory, redundant logic array, output multiplexer, fault diagnosis, SMC controller**.**

Instruction Pointer is used to point the which instruction we have to fetch from the instruction storage. It works for every rising edge of the Clk depending on the enabling signals and Rst values. If Rst is active low InstAddr is reset to zero.

Instruction Storage is used to store the instructions. 22 instructions are used to find the faults in the memory. These instructions are stored in the instruction register. If Rst is active high, for every rising edge of the Clk if InstEna is active high one instruction will be fetched from the memory from the location instruction address pointed by instruction pointer.

Instruction Register is used to store the instruction. Which is coming out from the instruction storage. The instruction is decoded and given as input to the required modules. It is a 7-bit register. If Rst is active low instruction register value is reset to zero .If Rst is active high and for every rising edge of the Clk, if IREna is active high instruction is stored in the instruction register and decoded.

AddrGen is used to generate the address. If Rst is active low, address will be reset to zero, otherwise for every rising edge of the Clk, if AddrEna is active high address will be incremented or decremented according to the input signals.

DataGen is used to generate the data, which is given as input to the memory.. If Rst is active low, data will be reset to zero, otherwise for every rising edge of the Clk, if DataEna is active high data will be according to the input signals.

RWControl is used to generate the RdEna and WrEna signals, which are given as inputs to the memory.. If Rst is active low, then RdEna and WrEna signals will be reset to zero, otherwise for every rising edge of the Clk, if RWEna is active high then RdEna and WrEna signals will be set according to the input signals.

Input Multiplexer is used to select one group of signals depending on whether it is working in Normal/Test mode. Multiplexer output is given as input to the memory.

Memory is used as a unit under test. If MemEna,WrEna both are active high and RdEna is active low, the data is written into the memory location specified on the address signal. If MemEna, RdEna both are active high and WrEna is active low, the data is from the memory location specified on the address signal.

Fault Diagnosis is used to compare the expected data with the original data. If any change is there, it gives that location address and actual data as input to the Redundant Logic Array.

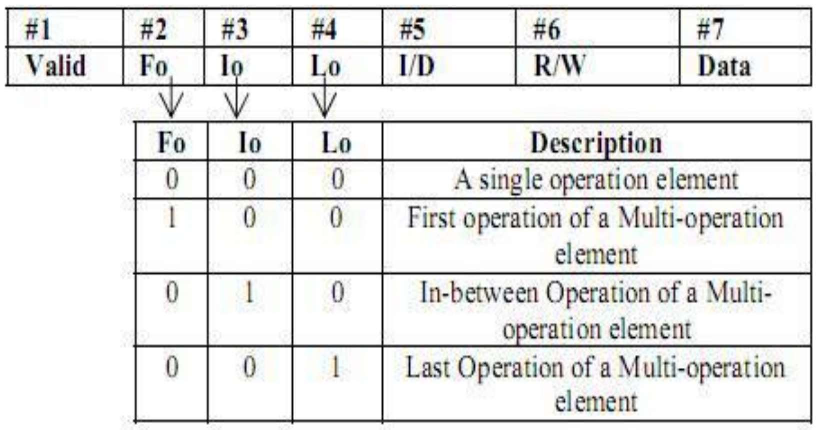
Redundant Logic Array acts as the redundant memory. In this we will store the memory faulty locations address and data. In normal mode it compares normal input address with the existing faulty locations, if it matches it uses redundant logic memory for read and write operations. If it doesn’t match it will use the original memory for read and write operations.

Output multiplexer is used to select one value from the Redundant memory and Memory depending whether it is faulty or not.

3. Specification of Microcode Instruction

The purposed architecture has the ability to execute algorithms with unlimited number of operations per March element. Thus almost all of the recently developed March algorithms can be successfully implemented and applied using this architecture. This has been illustrates in the present work by implementing March SS algorithm. The same hardware has also been used to implement other new March algorithms. This needs just changing the Instruction storage unit, or instruction codes and sequence inside the instruction storage unit. The instruction storage is used to store predetermined test pattern. The microcode is a binary code that consists of a fixed number of bits, each bit specifying a particular data or operation value. As there is no standard in developing a microcode MBIST instruction, the microcode instruction fields can be structured by the designer depending on the test pattern algorithm to be used. The microcode instruction developed in this work is coded to denote one operation in a single micro word. Thus a five operation March element is made up by five micro-code words. The format of 7 bit microcode MBIST instruction word is as shown in Table 1.

Table1. *Format of Microcode Instruction word*



#1	#2	#3	#4	#5	#6	#7
Valid	Fo	Io	Lo	I/D	R/W	Data

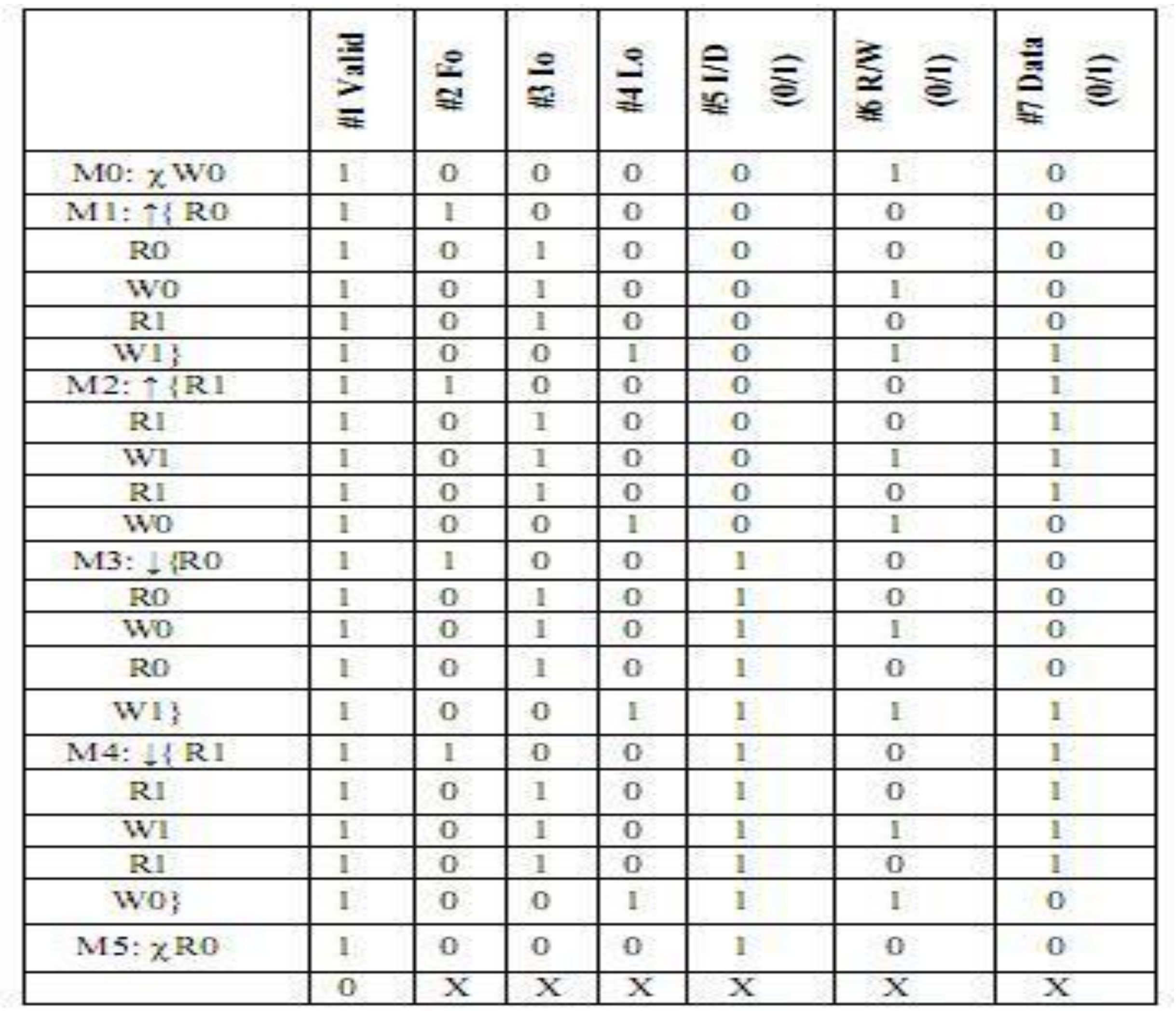
Fo	Io	Lo	Description
0	0	0	A single operation element
1	0	0	First operation of a Multi-operation element
0	1	0	In-between Operation of a Multi-operation element
0	0	1	Last Operation of a Multi-operation element

Its various fields are explained as follows: Bit #1 (=1) indicates a valid microcode instruction, otherwise, it indicates the end of test for BIST Controller. Bits #2, #3and #4 are used to specify first operation, in-between operation and last operation of a multi-operation March element, interpreted as shown in Table 1.

Bit #5 (=1) notifies that the memory under test (MUT) is to be addressed in decreasing order; else it is accessed in increasing order. Bit #6 (=1) indicates that the test pattern data is to be written into the MUT; else, it is retrieved from the memory under test. Bit #7 (=1) signifies that a byte of 1s is to be generated (written to MUT or expected to be read out from the MUT); else byte containing all zeroes are generated.

The instruction word is so designed so that it can accommodate any existing or future March algorithm. The contents of Instruction storage unit for March SS algorithm are shown in Table 2.

Table2. *March SS Algorithm*



	#1 Valid	#2 Fo	#3 Io	#4 Lo	#5 I/D (0/1)	#6 R/W (0/1)	#7 Data (0/1)
M0: χ W0	1	0	0	0	0	1	0
M1: ↑{ R0	1	1	0	0	0	0	0
R0	1	0	1	0	0	0	0
W0	1	0	1	0	0	1	0
R1	1	0	1	0	0	0	0
W1}	1	0	0	1	0	1	1
M2: ↑{R1	1	1	0	0	0	0	1
R1	1	0	1	0	0	0	1
W1	1	0	1	0	0	1	1
R1	1	0	1	0	0	0	1
W0	1	0	0	1	0	1	0
M3: ↓{R0	1	1	0	0	1	0	0
R0	1	0	1	0	1	0	0
W0	1	0	1	0	1	1	0
R0	1	0	1	0	1	0	0
W1}	1	0	0	1	1	1	1
M4: ↓{ R1	1	1	0	0	1	0	1
R1	1	0	1	0	1	0	1
W1	1	0	1	0	1	1	1
R1	1	0	1	0	1	0	1
W0}	1	0	0	1	1	1	0
M5: χ R0	1	0	0	0	1	0	0
	0	X	X	X	X	X	X

The first march element M0 is a single operation element,which writes zero to all memory cells in any order,whereas the second march element M1 is a multioperation element, which consists of five operations: i)R0, ii) R0, iii) W0, iv) R1 and v) W1. MUT is addressedin increasing order as each of these five operations is performed on each memory location before moving on to the next location.

4. Word Redundancy MBISR

The BISR mechanism used here [17] employs an array of redundant words placed in parallel with the memory. These redundant words are used in place of faulty words in memory. For successful interfacing with already existing BIST solutions as shown in Fig. 2, the following interface signals are taken from the MBIST logic:

1) A fault pulse indicating a faulty position address

2) Fault address

3) Required data or correct data that is compared with the results of Memory under test.

The MBISR logic used here can function in two modes.

4.1. Mode1. *Test & Repair Mode*

In this mode the input multiplexer connects test collar input for memory under test as generated by the BIST controller circuitry. As faulty memory locations are detected by the fault diagnosis module of BIST Controller, the redundancy array is programmed. A redundancy word is as shown in Fig 4.

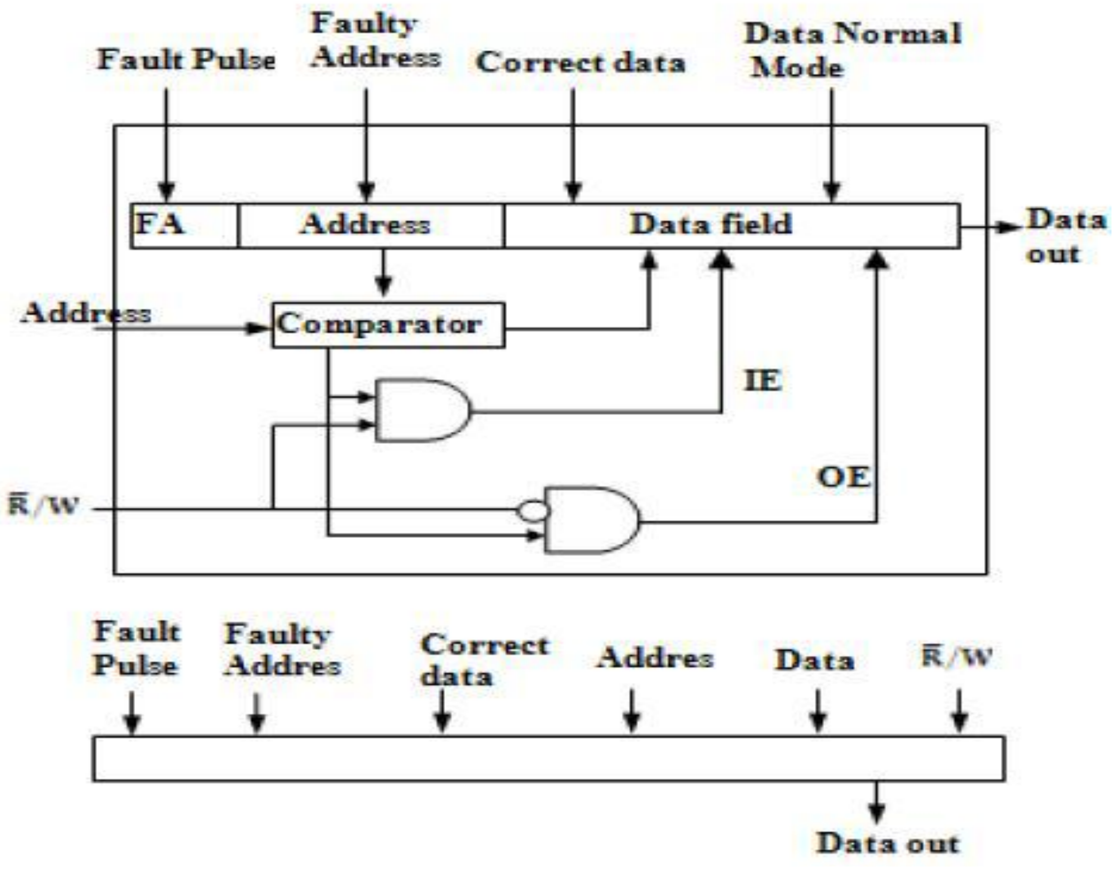


Fig4. *Redundancy Word Line*

The fault pulse acts as an activation signal for programming the array. The redundancy word is divided into three fields. The FA (fault asserted) indicates that a fault has been detected. The address field of a word contains the faulty address, here as the data field is programmed to contain the correct data which is compared with the memory output.

The IE and OE signals respectively act as control signals for writing into and reading from the data field of the redundant word. An overflow signal indicates that memory can no longer be repaired if all the redundancy words have been programmed. The complete logic of programming of memory array is shown in Fig.5 below

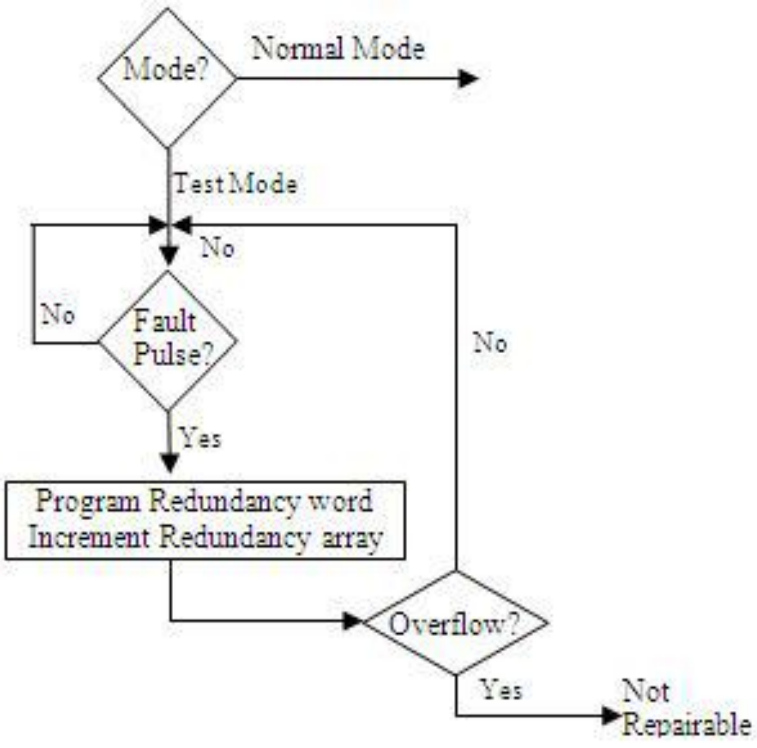


Fig5. *Flowchart of Redundancy Array*

4.2. Mode2. *Normal*

During the normal mode each incoming address is compared with the address field of programmed redundant words. If there is a match, the data field of the redundant word is used along with the faulty memory location for reading and writing data. The output multiplexer of Redundant Array Logic then ensures that in case of a match, the redundant word data field is selected over the data read out (= 0) of the faulty location in case of a read signal. This can be easily understood by the redundancy word.

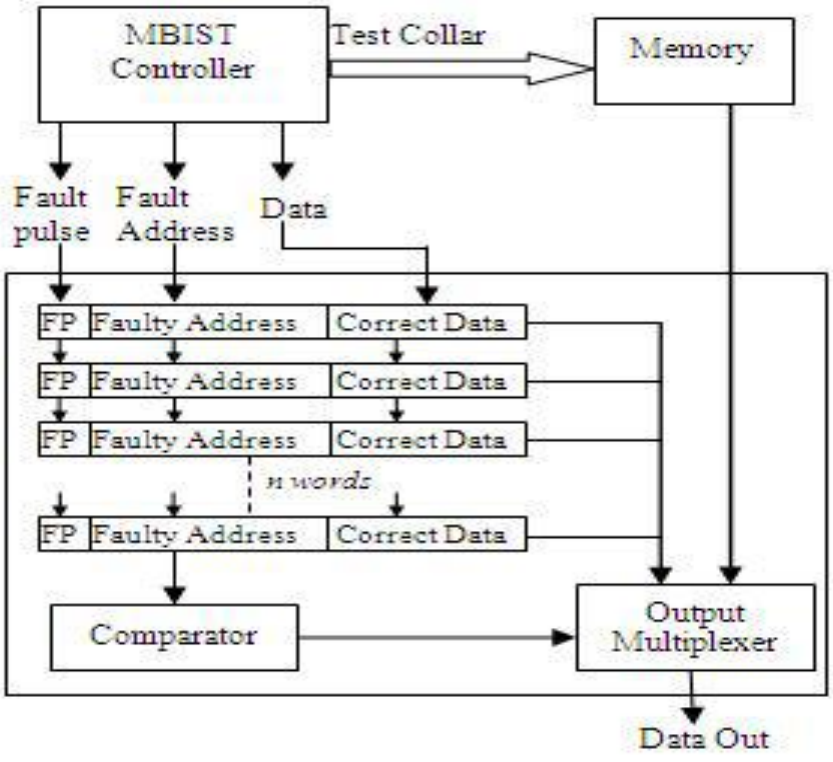


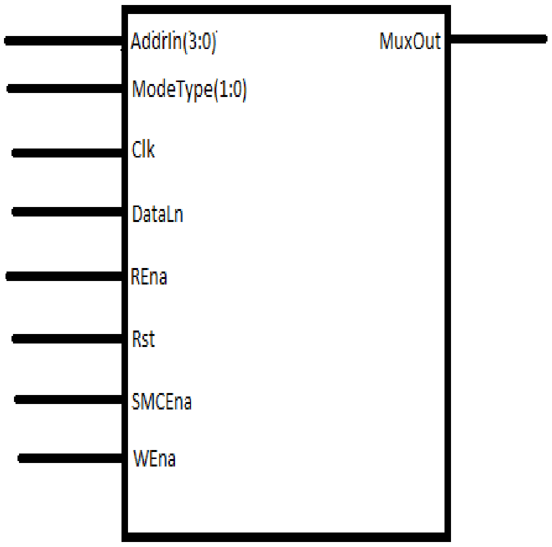
Fig6. *Repair module*

The above Fig .6, shows the repair module including the redundancy array and output multiplexer and its interfacing with the existing BIST module.

5. SIMULATION RESULTS

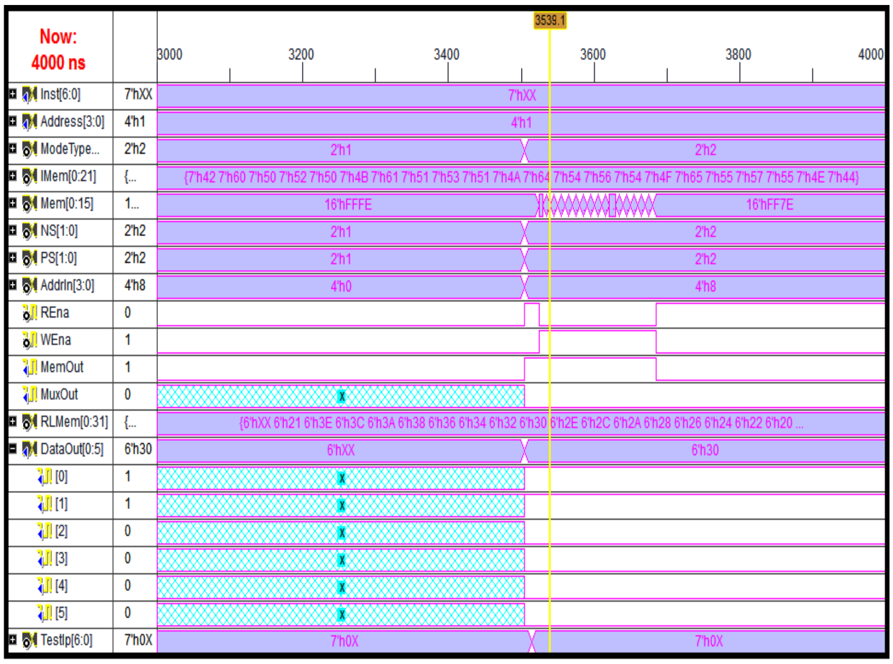
5.1. Top Module

The block diagram of TopModule is shown in Fig 6.29. It consists of Clock, Reset, SMCEnable, Modetype, Address in, Data in, Write Enable, Read Enable as inputs and Muxout as output.



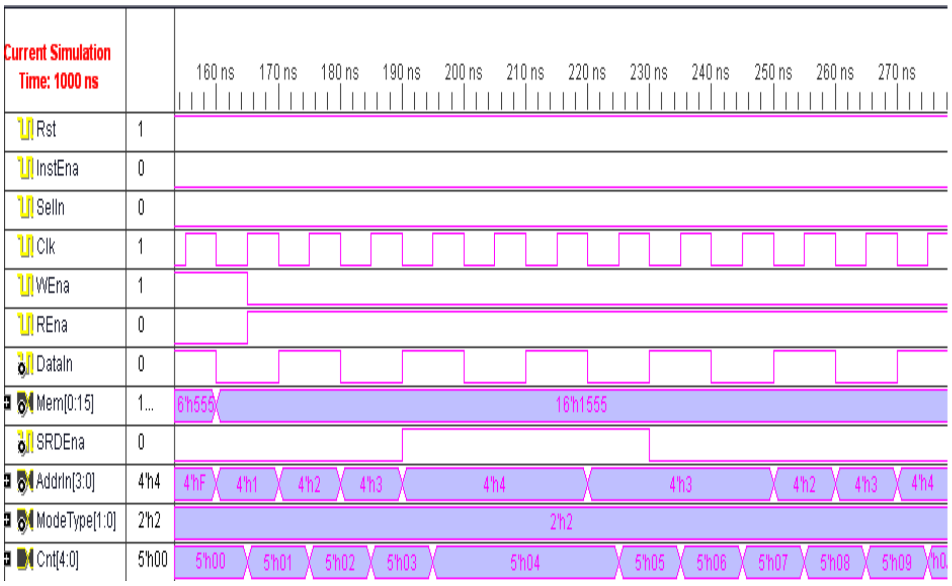
Top Module Block Diagram

The Output waveform of TopModule (Without REC) is shown in Fig 6.30. Here MuxOut is the output and it always depends on the mode type, REna or Wena. the data which has written in all the memory locations is logic ‘0’, but the memout value for the location 4’h8 is found faulty ,so the Muxout delivers the correct output which is coming from the redundancy logic array.


Now:
4000 ns
3539.1
3000
3200
3400
3600
3800
4000
Inst[6:0] 7'hXX 7'hXX
Address[3:0] 4'h1 4'h1
ModeType... 2'h2 2'h1 2'h2
IMem[0:21] {... {7'h42 7'h60 7'h50 7'h52 7'h50 7'h4B 7'h61 7'h51 7'h53 7'h51 7'h4A 7'h64 7'h54 7'h56 7'h54 7'h4F 7'h65 7'h55 7'h57 7'h55 7'h4E 7'h44}
Mem[0:15] 1... 16'hFFFE 16'hFF7E
NS[1:0] 2'h2 2'h1 2'h2
PS[1:0] 2'h2 2'h1 2'h2
AddrIn[3:0] 4'h8 4'h0 4'h8
REna 0
WEna 1
MemOut 1
MuxOut 0 X
RLMem[0:31] {... {6'hXX 6'h21 6'h3E 6'h3C 6'h3A 6'h38 6'h36 6'h34 6'h32 6'h30 6'h2E 6'h2C 6'h2A 6'h28 6'h26 6'h24 6'h22 6'h20 ...
DataOut[0:5] 6'h30 6'hXX 6'h30
[0] 1 X
[1] 1 X
[2] 0 X
[3] 0 X
[4] 0 X
[5] 0 X
TestIp[6:0] 7'h0X 7'h0X 7'h0X

Output waveform of TopModule (Without REC)

The Output waveform of TopModule (With REC) is shown in Fig 6.31.The Reliability is measured in the terms of number of memory accessing. The read operation is performed on the location 4'h4 twice then the count value incremented once when SRDEna is logic '1'.else the count value is incremented irrespective of operation performed on the memory.



Output waveform of TopModule (With REC)

6. CONCLUSION

The simulation results have shown that the micro-coded MBIST architecture described here is an effective testing method to test embedded memories as it provides a flexible approach and better fault coverage. Just as March SS, any new march algorithm can be implemented using the same BIST hardware by changing the instructions in the microcode storage unit, without the need to redesign the entire circuitry.

REFERENCES

[1] International SEMATECH, "International Technology Roadmap for Semiconductors (ITRS): Edition 2001"

[2] S. Hamdioui, G.N. Gaydadjiev, A.J .van de Goor, "State-of-art and Future Trends in Testing Embedded Memories", International Workshop on Memory Technology, Design and Testing (MTDT'04), 2004.

[3] S. Hamdioui, Z. Al-Ars, A.J. van de Goor, "Testing Static and Dynamic Faults in Random Access Memories", In Proc. of IEEE VLSI Test Symposium, pp. 395-400, 2002.

[4] S. Hamdioui, et. al, "Importance of Dynamic Faults for New SRAM Technologies", In IEEE Proc. Of European Test Workshop, pp. 29-34, 2003.

[5] S. Hamdioui, A.J. van de Goor and M. Rodgers, "March SS: A Test for All Static Simple RAM Faults", In Proc. of IEEE International Workshop on Memory Technology, Design, and Testing, pp. 95-100, Bendor, France, 2002.

[6] Dr. R.K. Sharma " Modeling and Simulation of Multi-Operation Microcode-based Built-In Self Test for Memory", Fault Detection and Repair, 2010 IEEE Annual Symposium on VLSI