International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 1 Issue 9, November- 2012

Reliability Enhancement of SRAM and it's Fault Repairing by using March-SS algorithm

N.B.JILANI*1 K.MURALI KRISHNA*2
*1M tech (VLSID), *2Assistant Professor,
Department of ECE, ADITYA ENGINEERING COLLEGE, Surampalem,
East Godavari (dt), A.P.,India

Abstract- As embedded memory on-chip is increasing and Memory Density is growing, problem of faults is growing exponentially. For detecting and repairing certain new faults March SS algorithm was introduced. For implementing this March SS algorithm, a word-oriented memory Built-in self Repair Methodology is employed to repair the faulty locations indicated by the MBIST controller. This paper also presents to prevent a SRAM from executing successive multiple read operations on the same position, such that the hard-to-detect defects can't manifest as functional faults. This can prolong the life time of the SRAM with latent hard-to detect defects. Experimental results show that the proposed reliability-enhancement circuit (REC) can effectively improve the reliability of SRAMs.

Keywords- Built-in Self Test(BIST); Built-in Self Repair(BISR);Memory Built in self Test(MBIST);Memory Built in self Test(MBISR), March Element ,Reliability enhancement circuit(REC).

I.INTRODUCTION

Most SOCs consist of many diversified memory cores due to today's memory hungry applications. With the shrinking transistor size and aggressive design rules, memory cores are easily prone to manufacturing defects and reliability problems. Therefore, efficient reliability-enhancement techniques are imperative for modern random access memories (RAMs). Built-in self-repair (BISR) techniques have been shown to be a good approach for repairing embedded memories. Various BISR approaches for memories have been reported in [1]–[6]. A BISR circuit usually consists of a built-in self-test (BIST) component, and Redundancy Logic array(RLA). The BIST is used to detect the targeted functional faults. The RLA allocates the redundancy (spare element) according to the detected fault patterns. To detect and Repair certain new faults a March-SS algorithm was used. A new Microcoded BIST architecture here which is capable of employing this March-SS algorithm. The interface of BISR with REC is shown in fig 1.

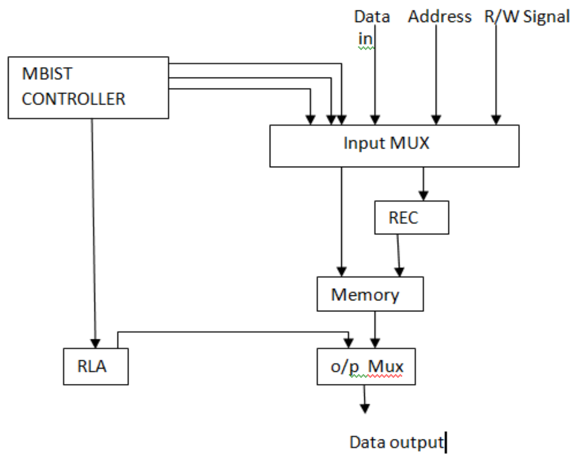


Fig1.Block Diagram of proposed BISR

II.DETAILED ARCHITECTURE OF BISR

The DPM screening of a large number of tests to a large number of applied to a large number of memory chips,

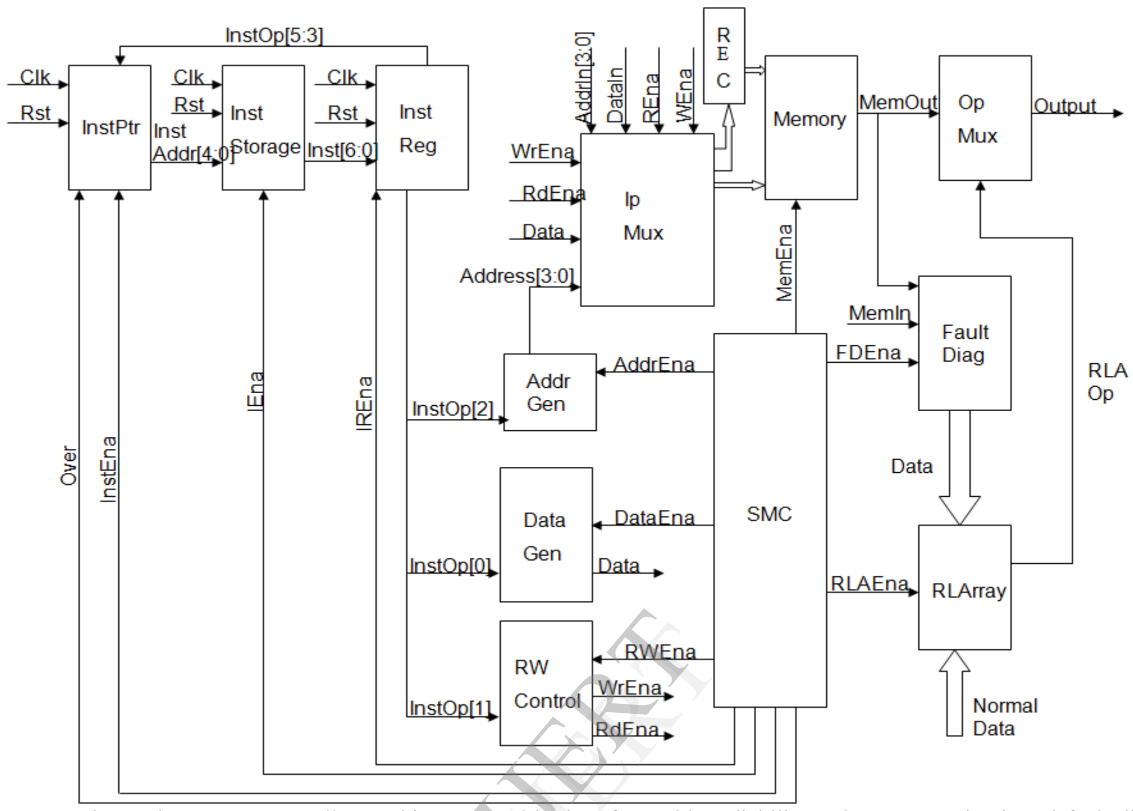


Figure2. Microcode MBIST controller Architecture and its interface with Reliability Enhancement circuit and fault diagnosis through input MUX

Showed that many well known fault models, developed before 1990's failed to explain the occurrence

faults. This implied that new memory technologies involving high density of shrinking devices lead to newer faults. This stimulated the introduction of new fault models, based on defect injection and SPICE simulation. Some such newly defined fault models [2] are Write Disturb Fault (WDF), Deceptive Read Disturb Coupling Fault (Cfdrd). Another class of faults called Dynamic faults which require more than one operation to be performed sequentially in time in order to be sensitized have also been defined. [3-4] These new faults cannot be easily detected by es tablished tests like March C-, rendering it insufficient/ inadequate for today's and the future high speed memories. More appropriate test algorithms like March SS and March RAW have been developed to deal with these new fault models. While March SS covers some of the new fault models like Deceptive Read Destructive fault (DRDF), Write disturb fault (WDF), etc., March RAW covers some of the Dynamic faults. Thus architectures 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 developed algorithms have up to six or seven (or even more) number of test operations per test element. For example test elements M1 through M4 of March SS algorithm have five test operations per element. This is in contrast 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 that had been specifically designed to implement these older algorithms can only implement up to two March operations per march element, rendering them incapable of easily implementing the new test algorithms. The proposed 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 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 unit is used to store predetermined test pattern.

A)Methodology

The block diagram of the BIST controller architecture together with fault diagnosis interface through input MUX shown in Figure .2 The BIST Control Circuitry consists of Clock Generator, Pulse Generator, Instruction Pointer, Microcode Instruction storage unit, Instruction Register. The Test Collar circuitry consists of Address Generator, RW Control and Data Control.

Pulse Generator generates a 'Start Pulse' at positive edge of the 'Start' signal marking the start of test cycle.***Instruction Pointer*** points to the next microword, that is the next march operation to be applied to the memory under test (MUT). Depending on the test algorithm, it is able to i) point at the same address, ii) point to the next address, or iii) jump back to a previous address. ***Instruction Register*** holds the microword (containing the test operation to be applied) pointed at by the Instruction Pointer. The various relevant bits of microword are sent to other blocks from IR. ***Address Generator*** points to the next memory address in MUT, according to the test pattern sequence. It can address the memory in forwards as well as backwards direction. ***RW Control*** generates read or write control signal for MUT, depending on relevant microword bits.***Data Control*** generates data to be written to or expected to be read out from the memory location being pointed at by the Address Generator. The Address Generator, RW Control and Data Control together constitute the Memory *Test Collar*. ***Input Multiplexer*** directs the input to memory by switching between test algorithm input and input given externally during the normal mode. The control signal for this multiplexer is also given externally by the user. If it indicates test mode then internally generated test data by BIST controller is given to the memory as input from the Test Collar. In case of Normal mode the memory responds to the external address, data and read/write signals. ***Fault Diagnosis*** module works during the test mode to give the fault waveform which consists of positive pulses whenever the value being read out of the memory does not match the expected value as given by Test Collar. In addition, it also gives the diagnostic information like the faulty memory location address and the expected/correct data value. This diagnostic information is used for programming the repair redundancy array as explained in the following section.

B) Microcode Instruction specification.

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 [7], 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 microword. 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 Fig. 6. 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, #3 and #4 are used to specify first operation, in-between operation and last operation of a multi-operation March element, interpreted as shown in Figure 6. 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

#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

Fig3. Format of microcode instruction word

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 1. 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 multi- operation element, which consists of five operations: i) R0, ii) R0, iii) W0, iv) R0 and v) W1. MUT is addressed in increasing order as each of these five operations is performed on each memory location before moving on to the next location.

	#1 valid	#2 F0	#3 I0	#4L 0	#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
WO	1	0	1	0	0	1	0
RO	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: χR0	1	0	0	0	1	0	0
	0	X	X	X	X	X	X

Fig.4 Contents of instruction storage unit for March SS Algorithm

III.WORD REDUNDANCY MBISR

The MBISR logic used here can function in two modes.

A) *Mode 1: 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 Figure 7. 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, whereas 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.

B) Mode 2: Normal Mode

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 (R/W=0) of the faulty location in case of a read signal. This can be easily understood by the redundancy word detail shown in Figure 7. Figure 9 shows the Repair Module including the redundancy array and output multiplexer and its interfacing with the existing BIST module.

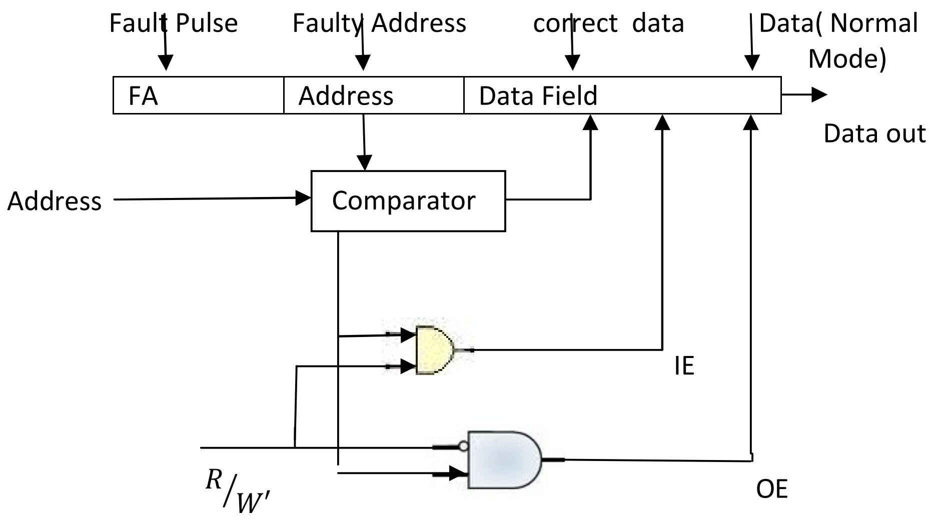


Fig.5 Redundancy word Line.

III.RELIABILITY ENHANCEMENTS :-

A.Hard-to-Detect Resistive-Open Defects

The fault behavior of a dynamic fault caused by a resistive-open defect may vary with the defect size (the resistance value of the defect). For example, if a resistive-open defect existing between the Vdd and the pull-up transistor of an 6 T SRAM cell, then the defect may cause a data retention fault (DRF) [16] or a dRDF (also called dynamic destructive read fault [15]). The testing of DRF in SRAM is difficult. To cope with this problem, some efficient design-for-testability techniques were proposed. If a resistive-open defect in the pull-up transistor of a SRAM cell causes a dRDF, then it is detectable by a Read after Write operation. But if the size of the resistive-open defect is small then multiple Read operations after one Write operation are needed to detect it [12], [15]. This fault is regarded as a deceptive multiple read destructive fault (DMRDF) [12]. The sensitizing sequence for DMRDF is $(1\ w0\ r0)^k$ or $(0w1r1)^k$ which means that k read operations after a one Write operation are executed. Note that if a cell with DMRDF, then the cell content is correct when the (k- 1)th Read operations are executed, and the cell content is changed when the kth read operation is executed but the read data of the kth Read operation is still correct. Therefore, an additional read operation is necessary to observe the fault. In this paper if a resistive-open defect in a RAM cell causes a DMRDF, then we look the defect as a hard-to-detect defect Hence, the required number of successive read operations for detecting the resistive-open defect in the pull-up path of a SRAM cell is related to the size of the defect. To further investigate the characteristics of the resistive-open defect, we use a circuit-level simulator, Hspice, to analyze the relation between the number of successive read operations and the size of the resistive-open defect using 0.18m TSMC CMOS Technology.

Consider an SRAM cell is operated at 1.4 ns. If the bit-line capacitance (Cb) is 25 fF, then one successive read operation is required from 0.99 to 1.13M ;Three successive read operations are required for detecting the defect which size is ranged from 0.979M to 0.99Ω. If the defect size is smaller than 0.979M , then the SRAM cell can be operated correctly.

Proposed Reliability Enhancement Scheme:-

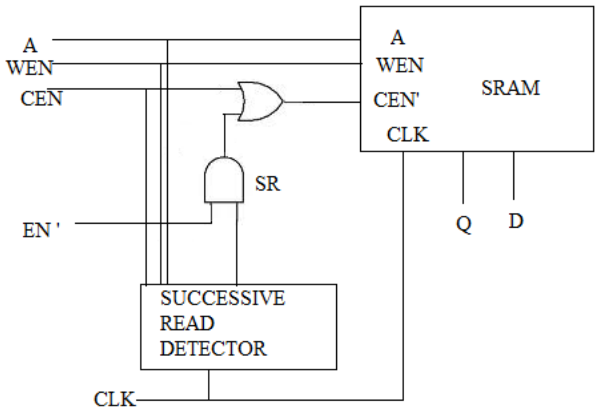


Fig.6 Reliability Enhancement circuit

B. Design-for-Reliability Scheme

As Section II-A describes, the size of the resistive-open defect has heavily in uence on the number of successive read operations used to detect the dynamic fault manifested by the defect. If the resistance of a resistive-open defect in an SRAM is small, then the SRAM cell may be operated correctly. However, the resistance of the resistive-open de-fect may become large during the lifetime of RAMs. Once the defect size is larger than the threshold value, it causes a dynamic fault in the RAM. Therefore, the lifetime of RAMs is reduced, i.e., the reliability of RAMs is reduced. Also, the number of successive read operations required to sensitize the defect is decreasing with the increase of the defect size. One simple approach to prolong the lifetime of a RAM cell with this type of latent defects is to prevent the RAM cell from successive multiple read operations. Thus, if no successive multiple read operations on a RAM cell is allowed, then the larger defect size can be tolerated for the RAM cell. For example, consider a SRAM cell with 25 fF bit- line load is operated at 1.4 ns. Assume that no successive multiple read operations on the SRAM cell is allowed. Then the SRAM cell still can work correctly when the defect size is smaller than 1.13 M . However, if successive multiple read operations on the SRAM cell are allowed, then the SRAM cell only can work correctly when the defect size is smaller than 0.979 M . Therefore, the time of resistance degradation from 0.979 M to 1.13 M is the prolonged lifetime of the SRAM. Fig. 6 shows the proposed reliability-enhancement circuit (REC) for RAMs. The REC major consists of a Successive Read Detector (SRD). The SRD checks whether successive multiple read operations on the same position are executed. The enable signal (EN) determines if the SRAM is protected by the REC. If EN is logic 1 then the successive read(SR) output is blocked. So, CEN'=CEN. If EN is logic 0, then CEN'=CEN□SR, where □ denotes a bit-wise OR operation. Therefore , if SR=1(i.e., successive read operations on the same address is detected), then CEN' is logic 1 regardless of the logic value of CEN and the SRAM is in idle state Fig. 7 depicts the detail implementation of the SRD. An example is illustrated to explain the operation of the SRD. Assume that a SRAM executes three successive Read operations on the addresses A1, A1, and A2. The address A1 is stored in the Address Register when the SRAM executes the first Read operation, since the control signal of the multiplexer is 1 (CEN = 0 and WEN = 1). Also, the Y will be logic 1 after the first Read operation is done. Subsequently, if the SRAM exe- cutes the second Read operation, the input

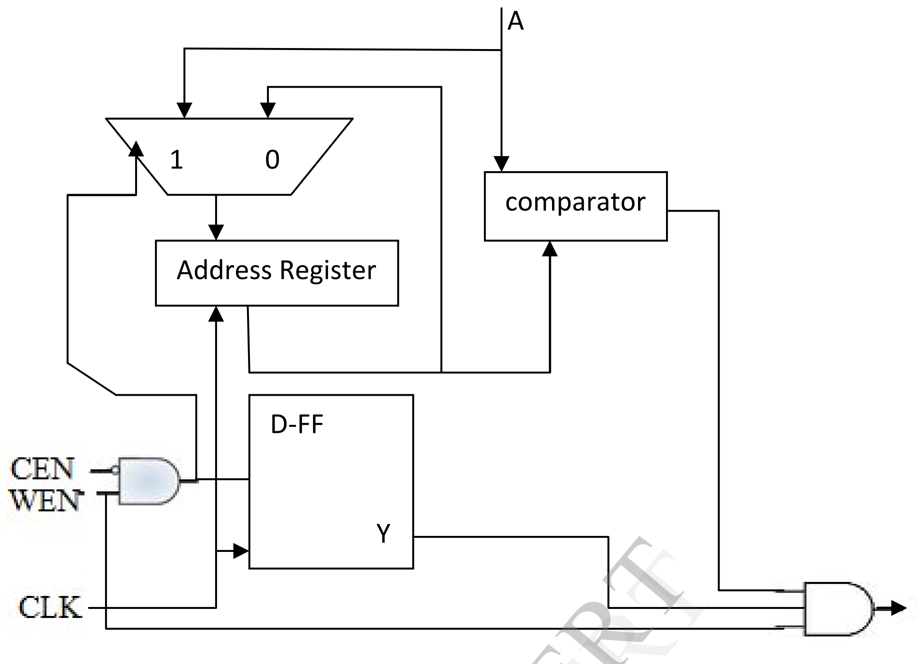


Fig. 7. Detail implementation of the Successive Read Detector.

address A1 is compared with the address A1 is compared with the address A1 stored in the address register. Since the address of the second read operation is same as the first one, the output z of the comparator is logic 1. Therefore, SR=Y&WEN&Z= 1 when the second Read operation is executed, where & denotes a bit-wise AND operation.So the CEN of the SRAM is forced to logic 1 and the SRAM is in idle state. The data at the output of the SRAM is the same as that of the first Read operation . Thus, we still can get the right data from the SRAM even the SRAM is in idle state. Finally, the SRAM executes the third Read operation and the output of the Comparator is logic 0, since the address stored in address register is different from the input address. Therefore, the SR becomes 0 and the SRD becomes transparent to the SRAM.

Simulation Results-
Mentor Graphic’s ModelSim has been used to verify the functionality and timing constraints of Verilog coded BIST module, Repair redundancy array and Reliabilty Enhancement circit. The full architecture containing all these modules has been successfully synthesized using Xilinx ISE 9.1i. Design Synthesis Report shows that a total of 136 slices, 107 slice flip-flops, 203 4 input LUTs and 13 bonded IOBs have been used in the synthesis. The simulation waveform of a fault-free SRAM is shown in Figure8 . The top module here is ‘bist’ which comprises of glue logic and interfacing of BIST Controller (including test collar), MUT, fault diagnosis module, and repair array. As the START signal goes high, indicating the start of test, the first

March element M0 of March SS algorithm is executed. This being a write signal, no values are read out from the memory to be compared with expected or correct values and hence the output FAULT waveform of comparator is showing high impedance for some initial clock cycles. As Read operation starts at the beginning of execution of M1 element, the values from MUT are read out ad compared with the expected values. The FAULT waveform shows a 'low' .

CONCLUSION:-

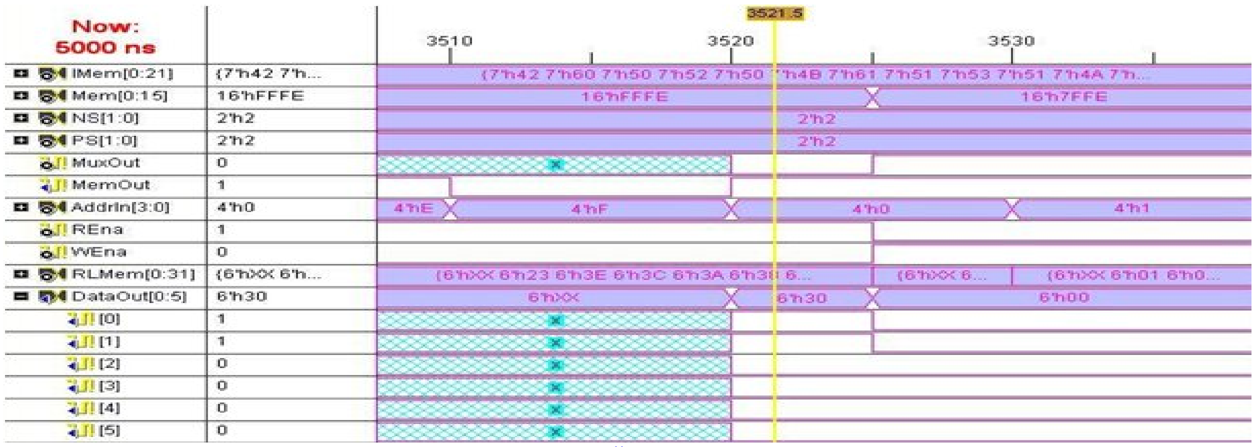
It works in two modes. Those are Normal Mode and Test mode. PS and NS signals represents these two modes. If PS is 1, it works in Test Mode and PS is 2, it works in Normal Mode. In Test Mode it detects the faults and stores faulty address and data in redundant logic memory (RLMem). In Normal Mode if WrEna is high, it compares the AddrIn with stored redundant logic address, if match found, it will write the data into the RLMem otherwise it will write into the Normal Memory (Mem). In Normal Mode if RdEna is high, it compares the AddrIn with stored redundant logic address, if match found, it will read the data from the RLMem otherwise it will read from Normal Memory (Mem).

DataOut comprises of Fault Pulse, Fault Address and Data. Fault Address is 8 and Data is 0. In MemOut it is showing 1. But in MuxOut we are getting the corrected data.

REFERENCES:-

[1] International SCHEMATIC ,"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)
[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.Himadioui ,A.J. Vande Goor and M..Rodgers," March-ss: A test for all statics simple RAM faults ", In proc .of IEEE, Internnational workshop on memory technology, design and testing, pp.95-100,bendor,france,2002
[6] N.Z. Haron , S.A.M Junos , A.S.A. Aziz, "Modeling and simulation of microcode built-in self test architecture for embedded memories", in proc. Of IEEE international symposium
[7]S.Borri, M.Hage,Hassan,L.Dilillo,P.Girard,S.Pravossoudovitch,and A.virazel," Analysis of dynamic faults in embedded –SRAM:implications for memory test", vol-21,no2,pp.169-179
[8]W. needham ,c.prunty,and E.H.yeoh," High volume microprocessor test escape, an analysis of defects our tests are missing" pp.25-34
[9] S.hamdioui,R.Wads worth, j.d.reys, " importance of dynamic faults for new SRAM Technologies"pp.29-34

Simulation Results of BISR with REC:-


Now: 5000 ns
3521.5
3510
3520
3530
IMem[0:21] {7'h42 7'h...
Mem[0:15] 16'hFFFE
NS[1:0] 2'h2
PS[1:0] 2'h2
MuxOut 0
MemOut 1
AddrIn[3:0] 4'h0
REna 1
WEna 0
RLMem[0:31] {6'hXX 6'h...
DataOut[0:5] 6'h30
[0] 1
[1] 1
[2] 0
[3] 0
[4] 0
[5] 0
{7'h42 7'h60 7'h50 7'h52 7'h50 7'h4B 7'h61 7'h51 7'h53 7'h51 7'h4A 7'h...
16'hFFFE
16'h7FFE
2'h2
2'h2
4'hE
4'hF
4'h0
4'h1
{6'hXX 6'h23 6'h3E 6'h3C 6'h3A 6'h3...
{6'hXX 6...
{6'hXX 6'h01 6'h0...
6'hXX
6'h30
6'h00