ISSN: 2277-9655
Impact Factor: 1.852

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

Design and Implementation of BISR for 3d Multiple SRAMS with Redundancies in a SOC

C.V.Keerthi Latha[*1], R.Hema latha[2]
[*1]M.E. Osmania University, Hyderabad, India
[2] PHD, Associate Professor, Osmania University, Hyderabad, India

Abstract

Embedded memories contain several hundreds of memory cores which constitute a significant portion of the chip area for typical system-on-chip (SOC) designs. With the shrinking transistor size and aggressive design rules, memory cores are easily prone to manufacturing defects and reliability problems. As these circuits have higher complexity and more sharing signals than logic blocks, they have higher failure possibilities In order to solve this problem; designers usually add redundancy to embedded memories. Most of faults are single cell transient fault; the area of spare is effectively utilized by replacing defected cell with spare cell. Continuing advancements in semiconductor technology have made sure that the integrated circuit industry keeps following the Moore's law, which predicts doubling the circuit density at a constant rate. This has been possible due to continuous scaling of CMOS transistor size and innovations in packaging. BISR is actually known and available for regular structures such as memory blocks, but is little difficult to implement on irregular logic. So the repairable memories play a vital role in improving the yield of chip.In this paper we present the efficient Reconfigurable Built-in Self Repair (Re BISR) circuit which increases repair rate. The proposed repair circuit is Reconfigurable for less area, used to repair multiple memories with different in size and redundancy. Built-in self-repair (BISR) techniques are widely used for enhancing the yield of embedded memories The techniques used for yield improvements in memories are Built In Self Test (BIST) and BIRA.BIST will verify the memory location by using MARCH CW algorithm .BIRA will perform built-in redundancy-analysis using BIRA algorithm for redundancy allocation A shared parallel BISR can test and repair multiple RAMs simultaneously. Typically, many RAMs with various sizes are included in an SOC. Memory designers usually employ efficient built-in redundancy-analysis (BIRA) algorithms which can cost-effectively be realized with built-in circuitries that are required for BISR schemes. Which are done by using spare rows and/or spare columns and spare I/O

Keywords: ReBISR, BIRA, BISR, MBIST, MARCH CW.

Introduction

With a trend of system-on-a-chip, a circuit or a system needs higher capacity of embedded memories. RAM is major component in present day SOC. When systems are fabricated in emerging nano-technologies it indicates a rising level of static and dynamic faults, due to new fault mechanisms Reliability is a key aspect of any SRAM chip. To detect the faults that can occur within a memory chip, extensive testing is carried out by both manufacturers and users of those chips. Fault detection in 3D memories is even more important due to additional processing involved in building a multi-layered chip.

Most of these techniques belong to the dedicated BISR scheme in which each memory has a self-contained BISR circuit, eg.,[2]-[10] Typically, the BISR circuit only represents a small portion of the corresponding memory area. However, it is common that there are several hundreds of memory cores in a complex SOC. If each memory core with redundancy has a dedicated BISR circuit, then area of the BISR will increase dramatically. . Therefore, efficient yield enhancement techniques for memory cores are essential. Built in self-repair (BISR) technique is a widely-used and efficient approach for the yield improvement of memory cores with redundancy [6]-[10].

Although soft errors are less susceptible which affect the reliability of SRAM memory chips they are still a cause of concern These errors can occur due to excess variations in node voltages, resulting in flipping of the data stored in the cell .In 3D memory chips, integration can result in reduced lengths of global wires and increased lengths of local wires; resulting in increased chip density, better latency, wide bandwidth and lower power consumption [5].

ISSN: 2277-9655
Impact Factor: 1.852

ORGANIZATION OF MEMORY

The simplified organization of a 2^{m+n}–bit memory chip is shown in Figure.1 Activating one of the 2^m block lines is performed by the row decoder

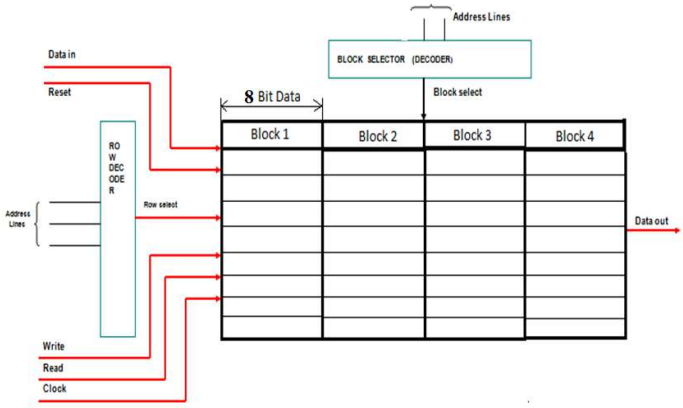

Figure 1: Simplified organization of a memory chip

SRAM Operation Modes

In access mode, SRAM users can decide whether the BISR is used base on their needs. If the BISR is needed, the Normal-Redundant words will be taken as redundancy to repair fault. If not, they can be accessed as normal words.

Table 1. SRAM Operation Modes

Modes	Repair Selection	Operation
Test Mode (test=1)	Default: Repair (bisr=1)	Access normal words Repair faults and test
	Don't Repair (bisr=0)	Access normal words Test only
Normal Mode (test=0)	Repair (bisr=1)	Access normal words Repair faults and write or read SRAM
	Don't Repair (bisr=0)	Access normal-Redundant words. Write or read SRAM only.

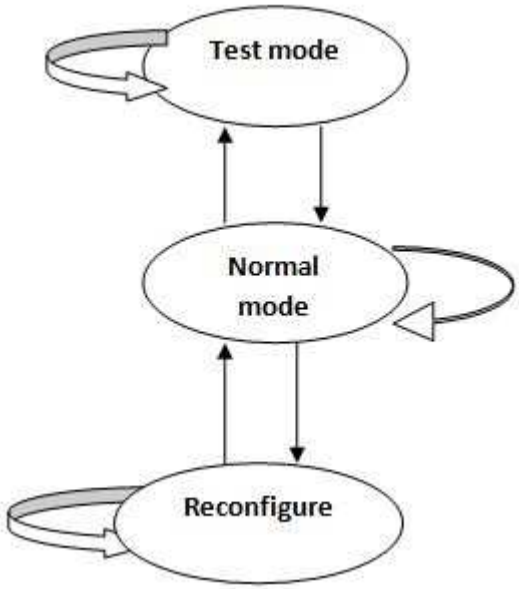

Figure 2: The State Transition Diagram of the Memory Selector

At the beginning, the reset signal, forces the memory selector to send all the memories to the normal mode. In this state all memory cores can be accessed normally, i.e., the test circuits are transparent to the user. When the memory selector receives the Test signal from the processor, the selected memory core enters the test mode and receives the test commands from the processor. If a fault is detected during the test process (when the test algorithm is running), the processor pauses the test algorithm and runs the ESP algorithm to find out how to reconfigure the address of the faulty cell.

This is done by the memory selector that forces the memory core to enter the reconfiguration mode. When the reconfiguration process finishes, acknowledged by the selected memory core, the processor sends the Done signal to resume the memory testing process, and the memory goes back to the test mode.

Built in Self Test (BIST)

Top level architecture

At the top level, the BISTR circuits together behave as a wrapper to the memory. Figure 4 below illustrates the direction of information flow in the system data, address, read strobe, write strobe, and block enable are input from a memory bus, to a multiplexer (MUX). The BIST also inputs its own version of these signals to the MUX, with BIST block enable as the MUX control signal During a test, BIST is control; during writes it sends addresses and data to the memory and during reads

ISSN: 2277-9655
Impact Factor: 1.852

it sends addresses to the memory and expected data to BISR.

A memory failure is determined by BISR when its CAM overflows because too many repairs have been attempted

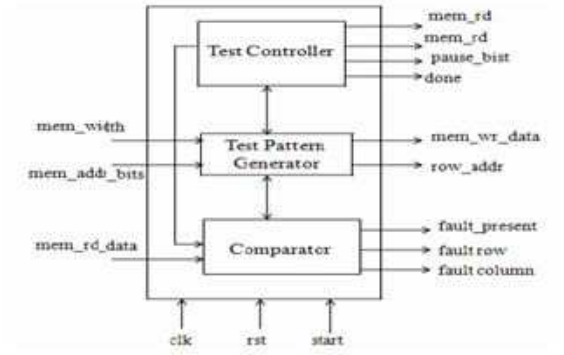

Figure 3: Block Diagram of BIST

MBIST is used to test the on chip memories. The block diagram of the proposed MBIST is shown in Figure 3 TPG is used to generate the test sequences which are applied to RAM during testing. Comparator is used to compare the output responses with the expected responses and decision is made whether the RAM is faulty or fault free. A controller is a hardware realization of a memory test algorithm, usually in the form of an FSM.The memory test algorithm here is Marching1/0 algorithm.

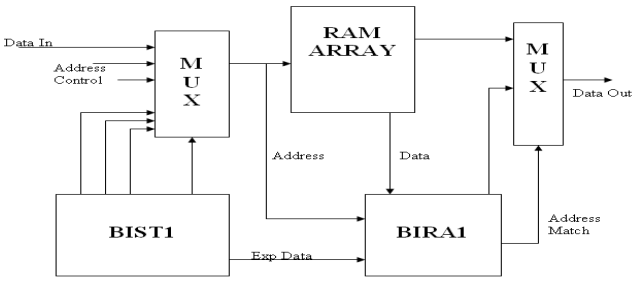

Figure 4: Top level architecture of BIST

Functionality of March CW Algorithm

This algorithm involves successive writing and reading of 0's and 1's into the RAM. The fault location details like fault present, fault row and fault column addresses are given at the output pins. The input pin start is made one when the BIST starts the testing process. The output pin done and progress are resulted to one and zero respectively, after the MBIST process is completed, so that the ReBIRA circuit can start analysis.

A March CW test consists of a finite sequence of March elements{ ⇓⇑w0;⇑(r0,w1); ⇑(r1,w0); ⇓(r0,w1);⇓(r1,w0);⇑⇓(r0) }. The finite state machine used to design the controller is shown in Figure 5. As shown below, the algorithm can be implemented using 7 state transitions (including initial state). In each of the states (except the initial state) either a 'read' operation or a 'write' operation or a combination of both will be taking place. The state machine is in the initial state (ME1) as long as the mode is zero. Once the mode becomes '1', it starts transiting from one state to the other. In ME1 state, '0' is written into each address location of the memory array starting from address 0.

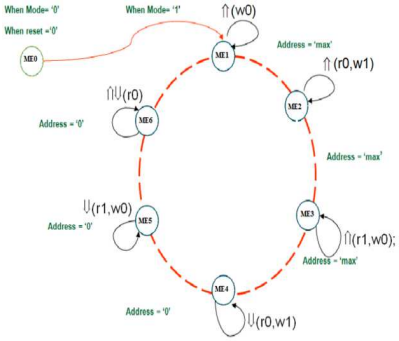

Figure 5: State diagram for March CW algorithm

The March CW algorithm, and its modifications, is a popular algorithm for memory testing. This algorithm, which consists of 11 operations (11n), writes and reads words of 0s, followed by writing/reading words of 1s, in both descending and ascending address spaces. The original March CW algorithm consists of the following steps:

1. Write 0s to all locations starting at the lowest address (initialization).
2. Read 0 at lowest address, write 1 at lowest address, repeating this series of operations until reaching the highest address.
3. Read 1 at lowest address, write 0 at lowest address, repeating this series of operations until reaching the highest address.
4. Read 0 at highest address, write 1 at highest address, repeating this series of operations until reaching the lowest address.
5. Read 1 at highest address, write 0 at highest address, repeating this series of operations until reaching the lowest address.
6. Read 0 from the lowest address to the highest address.

In this type testing occurs simultaneously with normal functional operation. This form of testing is usually accomplished using coding technique or duplication or comparison. Once the address reaches its maximum value, the state gets transited to ME2. In ME2,

ISSN: 2277-9655
Impact Factor: 1.852

the data written in ME1(i.e., '0') is read from an address location and '1' is written immediately into that address.

As and when the address reaches its maximum value, the state changes to ME3. In ME3, the data written in ME2 (i.e., '1') is read from an address location and '0' is written immediately into that address. In states ME2 and ME3 the address is in incrementing order. The controller goes to ME4 after the address reaches its maximum value in ME3. From ME4 the address gets decremented from its maximum value. In ME4 and ME5, the operations are same as those in ME2 and ME3 respectively but the address is decrementing. The state transition takes place when the address reaches its minimum value. In ME6, the data written during ME5 state (i.e., '0') is read from the memory array.

BISR Block Diagram in 3D

In block diagram of the ReBISR design shown in Figure6, RAM details table is used for storing the configurations of RAMs which includes the memory data width and depth, number of spare rows and number of spare columns. SMC is the main block that acts as a controller for generating the control signals during testing and repairing. The controller controls the BIST and ReBIRA circuit. The controller reads the RAM detail table and sends the RAM configurations like number of rows and number of columns to BIST and number of spare rows and number of spare columns to ReBIRA circuit.

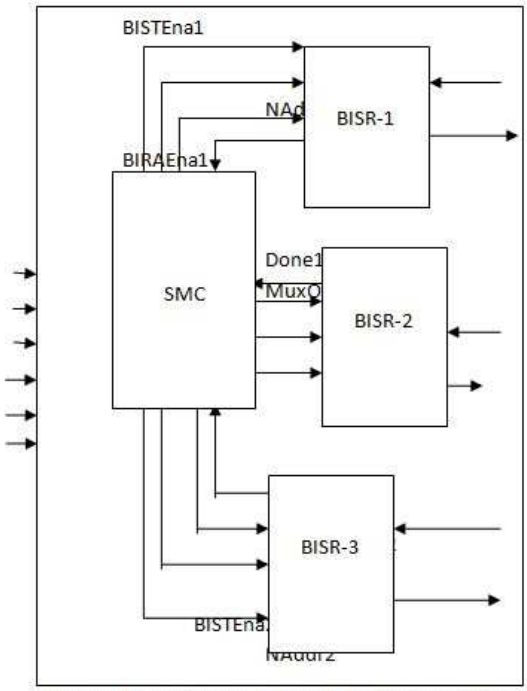

Figure 6:.Block Diagram of ReBISR for Multiple RAMS

MBISR in Test Mode

During RA, the MBIRA first records the unusable faulty spare elements and then allocates spare elements according to the faulty rows/columns of the main memory based on the test result. That is, the MBIST tests the spare memory first to determine the available spare elements then tests the main memory and simultaneously executes the RA process. When the spare elements are not enough for repairing the faults, the Go signal is de-asserted to abort the MBIST. Otherwise, when the test is done, the Ready and Go signals are asserted. Whenever the MBIST begins testing, the MBIRA starts the RA process by detecting the Mode signal

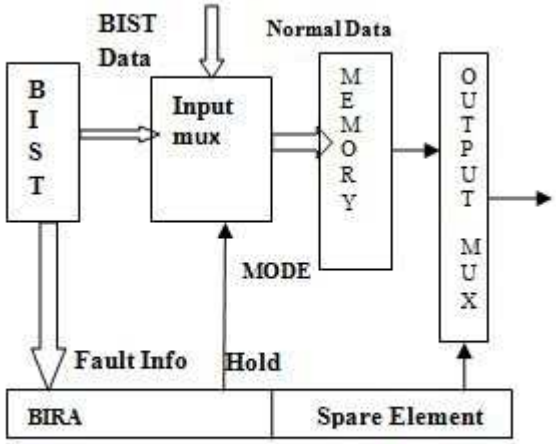

Figure 7: Modified Diagram of MBISR

During RA, the MBIRA checks the 2-bit Detect M/S signals in the MBIST for the main and spare memory faults, respectively, and analyzes each faulty address from A Fail within the same clock period. The MBIRA can use multiple clock cycles to pre-analyze the faulty addresses without suspending the MBIST, because it gets the addresses from the MBIST before the addressed data are read and determined faulty, especially for a high-speed pipelined memory.

MBIRA Address Remapping in Normal Mode: In the normal mode, the MBIRA remaps the faulty main memory addresses to those of the spare memory that have been determined during the RA process. Therefore, the spare memory has the same Chip Enable (MemEna) signal and data-in (Data) signals as the main memory, but it receives the remapped address (Addr) and write-enable (WEN) signals from the MBIRA.The MBIRA also can select the correct spare data-out (Data) to replace the faulty data-out (Temp) for the output .If a processor is addressing fault-free data in the memory, the

ISSN: 2277-9655
Impact Factor: 1.852

MBIRA de-activates WEN and switches Data, i.e., it disables the spare memory. Only if the MBIRA detects faulty main memory address on Addr from the processor, it concurrently remaps Addr to FAddr according to the previous RA result and stores Data in spare register.

The BISR Architecture procedure can be activated when turning on the power. Moreover, it can also be started by activating the BISREna pin. Upon the BISR procedure is started, the BIST circuit generates the test patterns specified by the March CW algorithm to test the spare memory array first during the BIST Spare Array session. In order to avoid using failed spare row blocks to repair failed row blocks in the memory array, it is necessary to test the spare array first.

During this session, if the BIST detects a fault, it is paused and issues a fail signal (Fail=1), to activate the BIRA circuitry. Upon detecting faults, BIRAEna module sets the corresponding spare row blocks unusable, and it catches the fault information of the fault through the faulty address and performs the redundancy analysis procedure. If the spare array does not have enough useable spare blocks to repair the faulty cells, the fail indicator (FAIL) goes to high. This signal indicates that the memory chip cannot be repaired. When the whole memory array has been tested (Done = 1) and the repair procedure is successful (Fail = 0), then the system can operate normally. Once the BIRA completes the redundancy analysis procedure, it informs the BIST to resume the test process through a signal. During the BIRA process, the repaired addresses are stored in the Spare Registers (Look-Up Table). If the memory cannot be repaired, the unrepairable signal is asserted. After the BIST Spare Array session is finished, the BIST Memory Array session follows. Finally, if a memory is unrepairable, the user can program the threshold value through the program signal and repeat the repair process.

Experimental Results

Topmodule_Normal

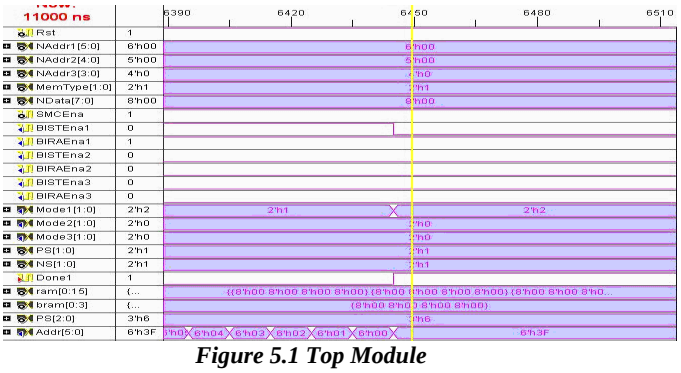

Figure 5.1 Top Module

The above report in Fig 5.1, explains that it enables SMC Block, clock and reset signals first and when present state is 3'h6 then makes Done=1 saying that it has completed its testing and ready to allocate spare memory.

SMC:

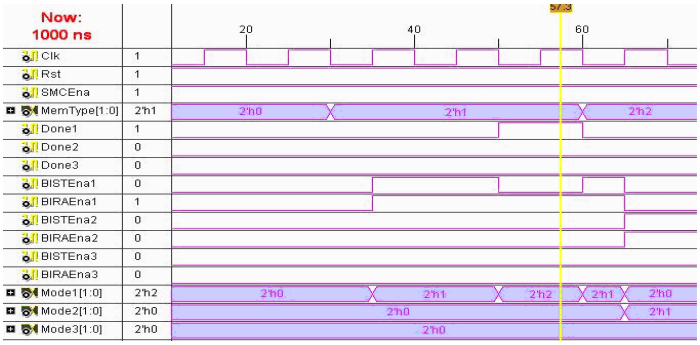

Figure 5.2 State Machine Controllers

The above report in Fig 5.2, explains that when memory type is 2'h1, enables BISTEna1 and BIRAEna1 blocks of BISR1 and if completes its testing, it makes Done=1, and changes to BISR-2 by enabling BISTEna2=1 and BISTEna2=1

BISR_16X32_ME6:

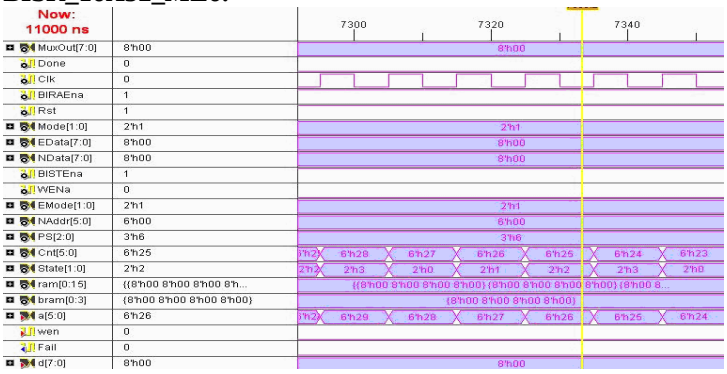

Figure 5.3 BISR in last state

The above report in Fig 5.3, explains that for BISTEna=1 and BIRAEna=1and having in State=2'h2 it makes wen=0 then Data (d) is read from Memory (ram) to MuxOut

BISR_16X32_ErrorWrite:

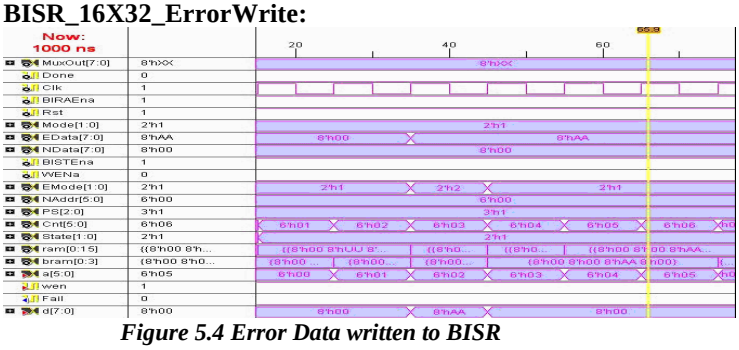

Figure 5.4 Error Data written to BISR

The above report in Fig 5.4, explains that an Error 8'hAA is introduced in one of the location of the memory, addressed 6'h05.

ISSN: 2277-9655
Impact Factor: 1.852

BISR_16X32_ME6_ErrorRead:

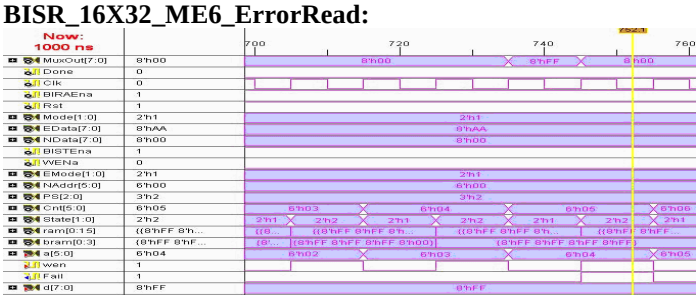

Figure 5.5 Error Data Read from BISR

The above report in Fig 5.5, explains that when present state is in 3'h2, NData is not equal to Data(d) , then makes Fail=1, saying that location addressed 6'h05 is faulty, then MuxOut is loaded with Data(d) instead of NData.

BIRA_16x32:

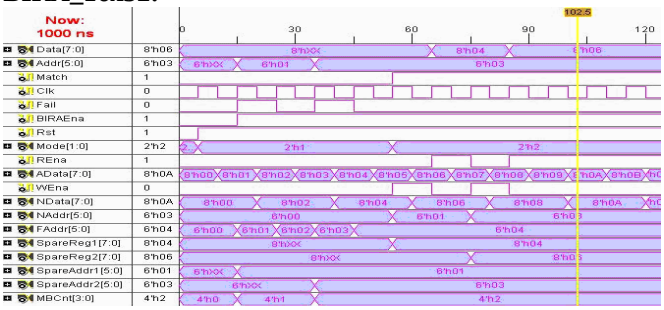

Figure 5.6 BIRA

The above report in Fig 5.6, explains that when reading data from memory it compares Addr(6'h03,6'h04) and SpareAddr(6'h03,6'h04), if they are same then it makes Match=1,then instead of refering memory it will go to SpareRegister(Sparereg2[7:0]) and get the Data

BIST_16x32:

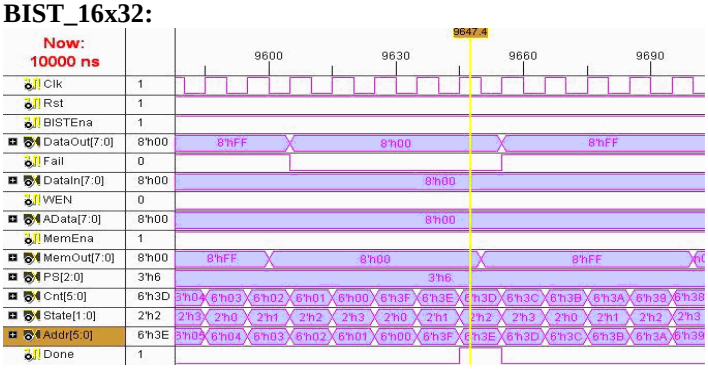

Figure 5.7 BIST in last state

The above report in Fig 5.7, explains that when PS is in 3'h6 it has finished its read and write operations in all memory locations addressed(from 6'h00 to 6'h FF) and makes Done=1 to indicate that its end of BIST for Memory 1

Conclusion

A reconfigurable BISR scheme for repairing multiple repairable RAMs with different sizes redundancies has been presented. BISR architecture is proposed which uses a multiple memories for repairs. The statistical analysis shows that this type of redundancy is able to achieve a very high Fault Coverage and Repair Ratio.

Our new 3D structure maintains the same read/write stability. This design utilizes the Spare Registers to allocate 3D redundancy elements. We can also program the parameter of the ReBIRA scheme to meet different requirements. It uses a spare memory generated from the same memory generator as for the main memory without any built-in redundancy. Other important features of this BISR scheme are its programmability, low access time penalty, and speed testing by reusing the on-chip processor core.

The programmability is again due to the proposed reconfiguration mechanism of our architecture requires negligible hardware overhead. According to experimental results, it also concludes that our approach improves the repair rate significantly. Simulation results has been verified that the proposed BISR architecture performs well even for high defect densities and requires a low memory area overhead, which had drastically improved the repair rate in comparison with the typical BIRA scheme. Especially for the defective RAMs with transient faults, the proposed scheme can provide a higher repair rate.

Future challenges in embedded SOC memory testing will be driven by the following items:

1. **Fault modeling**: New fault models should be established in order to deal with the new defects introduced by current and future (deep-submicron) technologies.
2. **Test algorithm design**: Optimal test/diagnosis algorithms to guarantee high defect coverage for the new memory technologies en reduce the DPM level.
3. **BIST**: The only solution that allows at-speed testing embedded memories
4. **BISR**: Combining BIST with efficient and low cost repair schemes in order to improve the yield and system reliability improve the yield and system reliability**.**

References

[1] Mincent Lee, Li-Ming Denq, and Cheng-Wen Wu,: "Memory Built-In Self-Repair Scheme Based on Configurable Spares" IEEE Transactions On Computer-Aided Design Of Integrated Circuits And Systems, Vol. 30, No. 6, June 2011

ISSN: 2277-9655
Impact Factor: 1.852

[2] T.-W. Tseng, J.-F. Li, and C.-C. Hsu, "ReBISR: A reconfigurable built-in self-repair scheme for random access memories in SOCs," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 18, no. 6, pp. 921–932, Jun. 2010

[3] C.-D. Huang, J.-F. Li, and T.-W. Tseng, "ProTaR: An infrastructure IP for repairing RAMs in SOCs," IEEE Trans. Very Large Scale Integr.(VLSI) Syst., vol. 15, no. 10, pp. 1135–1143,Oct.2007

[4]T.-W. Tseng, J.-F. Li, C.-C. Hsu, A. Pao, K. Chiu, and E. Chen, "A reconfigurable built-in self-repair scheme for multiple repairable RAMs in SOCs," in Proc. Int'l Test Conf. (ITC), Santa Clara, Oct 2006, Paper 30.2, pp. 1–8

[5]R. Patti, "Three-dimensional integrated circuits and the future of system-on-chip designs," Proc. of the IEEE, vol. 94, no.6,pp.1214-1224,June2006.

[6]L.-T. Wang, C.-W. Wu, and X. Wen, Design for Testability: VLSI Test Principles and Architectures. San Francisco, CA: Morgan Kaufmann, 2006.

[7]M. Nicolaidis, N. Achouri, and L. Anghel, "A diversified memory builtin self-repair approach for nanotechnologies," in Proc. IEEE VLSI Test Symp. (VTS), Napa Valley, Apr. 2004, pp.313–318.

[8]Y. Zorian and S. Shoukourian, "Embedded-memory test and repair: Infrastructure IP for SoC yield," IEEE Design & Test of Computers, vol. 20, pp. 58–66, May-June 2003.

[9]Y. Zorian, "Embedded memory test&repair: Infrastructure IP for SOC yield," in Proc. Int. Test Conf. (ITC), Baltimore, MD,Oct.2002,pp.340–349

[10]R. Rajsuman, "Design and test of large embedded memories: An overview," IEEE Des. Test Comput., vol. 18, no.3,pp.16–27,May2001

[11]R. Rajsuman, "Design and test of large embedded memories: An overview," IEEE Des. Test Comput., vol. 18, no.3,pp.16–27,May2001

[12]S. Runyon, "Testing big chips becomes an internal affair", IEEE Spectrum, pp. 49–55, Apr. 1999