Self-Repair Method to SRAM

[1]K.Naveen, [2]Md.Abdul rawoof, [3]CH. Srinu

Assistant Professor
ECE, MLRIT, Dundigal, Hyderabad, India

Abstract: Built-In Self-Repair (BISR) with Redundancy is an effective yield-enhancement strategy for embedded memories. This paper proposes an efficient BISR strategy which consists of a Built-In Self-Test (BIST) module, a Built-In Address-Analysis (BIAA) module and a Multiplexer (MUX) module. The BISR is designed flexible that it can provide four operation modes to SRAM users.
Each fault address can be saved only once is the feature of the proposed BISR strategy. In BIAA module, fault addresses and redundant ones form a one-to-one mapping to achieve a high repair speed. Besides, instead of adding spare
words, rows, columns or blocks in the SRAMs, users can select normal words as redundancy. The selectable redundancy brings no penalty of area and complexity and is suitable for compiler design. A practical 4K × 32 SRAM IP with BISR circuitry is designed and implemented based on a 55nm CMOS process. Experimental results show that the BISR occupies 20% area and can work at up to 150MHz.

KEYWORDS: **Built-In Self-Test (BIST) Built-In Self-Repair (BISR) Multiplexer (MUX)**

INTRODUCTION:

The area occupied by embedded memories in System-on-Chip (SoC) is over 90%, and expected to rise up to 94% by 2014[1]. Thus, the performance and yield of embedded memories will dominate that of SoCs. However, memory fabrication yield is limited largely by random defects, random oxide pinholes, random leakage defects, gross processing and assembly faults, specific processing faults, misalignments, gross photo defects and other faults and defects [2].

To increase the reliability and yield of embedded memories, many redundancy mechanisms have been proposed [3-6]. In [3-5] both redundant rows and columns are incorporated into the memory array. In [6] spare words, rows, and columns are added into the word-oriented memory cores as redundancy. All these redundancy mechanisms bring penalty of area and complexity to embedded memories design. To solve the problem, a new redundancy

scheme is proposed in this paper. Some normal words in embedded memories can be selected as redundancy instead of adding spare words, spare rows, spare columns or spare blocks.

Memory test is necessary before using redundancy to repair. Design for test (DFT) techniques proposed in 1970 improves the testability by including additional circuitry. The DFT circuitry controlled through a BIST circuitry is more time-saving and efficient compared to that controlled by the external tester (ATE) [7]. However, memory BIST does not address the loss of parts due to manufacturing defects but only the screening aspects of the manufactured parts [8]. BISR techniques aim at testing embedded memories, saving the fault addresses and replacing them with redundancy. In [9], the authors proposed a new memory BISR strategy applying two serial redundancy analysis (RA) stages. [10] Presents an efficient repair algorithm for embedded memory with multiple redundancies and a BISR circuit using the proposed algorithm. All the previous BISR techniques can repair memories, but they didn't tell us how to avoid storing fault address more than once. This paper proposes an efficient BISR strategy which can store each fault address only once.

2.1 Basic BIST Architecture:

The various components of BIST hardware are the test pattern generator (TPG), the test controller, circuit under test (CUT), input isolation circuitry and the output response analyzer (ORA). This is shown in the figure below.

2.6 BIST Types

They are 1. LBIST (Logical Built in Self Test) 2. MBIST (Memory Built in Self Test)

2.6.2 Memory BIST (MBIST)

With the advent of deep-submicron VLSI technology, core-based SOC design is attracting an increasing attention. On an SOC, popular reusable cores include memories, processors, input/output circuits, etc.

Memory cores are obviously among the most universal ones - almost all system chips contain some type of embedded memory. However, to provide a low cost-cost test solution for the on-chip memory cores is not a trivial task.

2.7 FAULTS

Stuck – at Fault: One or more logic values in the memory system cannot be changed i.e. the logic value of a cell or a line is always 0 or 1.

For example, one or more cells are "stuck at" 1 or 0. Stuck at faults are also useful for modeling faults in other parts of the memory system.

Address decoder faults:

Any fault that affects address decoder With a certain address, no cell will be accessed.

A certain cell is never accessed.

With a certain address, multiple cells are accessed simultaneously.

A certain cell can be accessed by multiple addresses.

Coupling fault: There exist two or more cells that are coupled. A pair of memory cells, i and j, are said to be coupled if a transition from x to y in one cell of the pair, say cell i, changes the state of the other cell, that is cell j, from 0 to 1 or from 1 to 0 i.e. a write operation to one cell changes the content of a second cell. This, of course, does not necessarily imply that a similar transition in cell j will influence cell in a similar manner.

Transition fault: A cell or a line that fails to undergo a 0 to 1 or a 1 to 0 transition.

3.1 Memory BIST (MBIST):

MBIST is used to test the on chip memories, where the CUT in the BIST structure is replaced with the on chip memories (RAM, SRAM, DRAM and Flash memory).The block diagram of the proposed MBIST is shown in the below figure 3.6. TPG is used to generate the test sequences which are applied to RAM during testing. Expected responses are compared with the output responses and decision is made whether the RAM is faulty or fault free.

3.3 Programmable MBIST

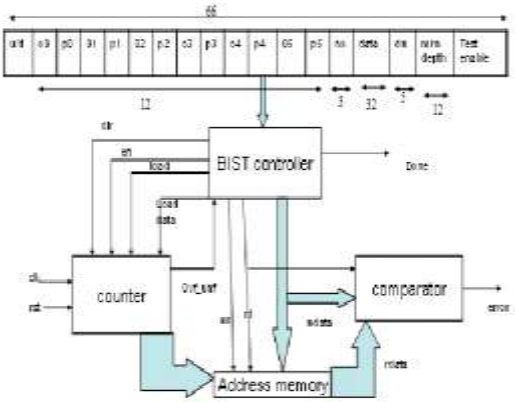
Most of Memory BIST approaches concerns the programmability of the memory test algorithm. To enabling programmability of all components of memory test, test algorithm, test data, address sequence.The programmable memory BIST proposed has several advantages:

- It enables programming both test algorithms and test data.

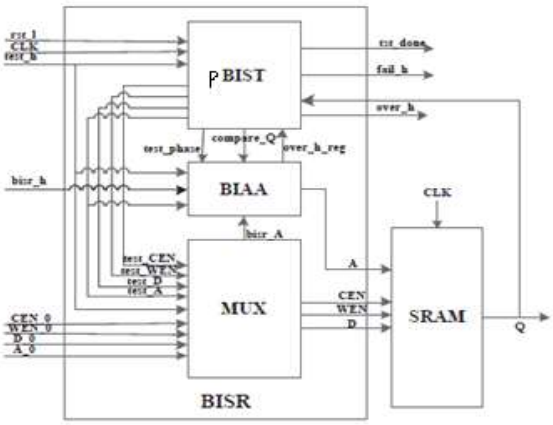
- It implements test algorithm programmability at low cost, by extracting the different levels of hierarchy of the test algorithm and associating a hardware block to each of them, resulting on low cost hardware.

- It enables low-cost implementation of full-data programmability by adapting the transparent memory test approach in a manner that uses the memory under test for programming the test data.

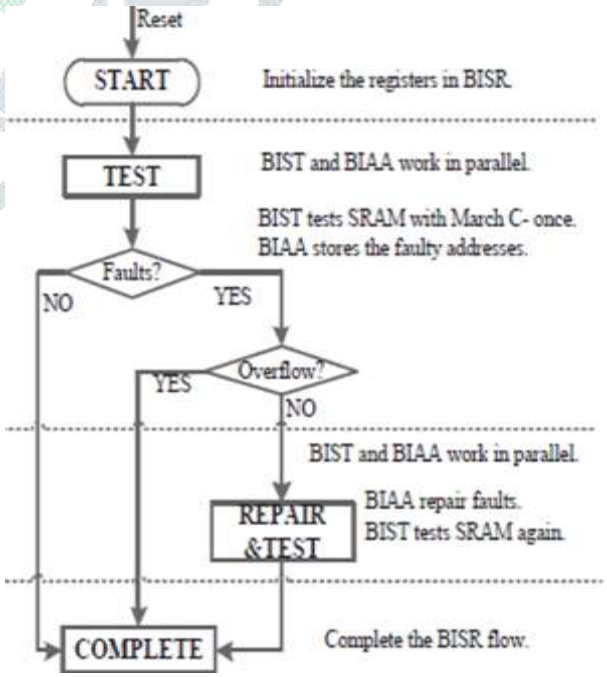
The architecture for programming march test algorithms proposed in the fig. This architecture uses an instruction register specifying the current match test sequence by means of several fields indicating.



5.5 OVERALL BISR ARCHITECTURE



5.6 BISR PROCEDURE



IMPLEMENTATION USING MARCH ALGORITHMS

The algorithms in most common use are the March tests. March tests have the advantage of short test time but good fault coverage. Test sequences or test algorithms for memories are known under the name of March tests. The test is "marching" through the memory. A March test consists of a sequence of March elements. A March element has a certain number of operations that must be applied to all memory cells of an array. The addressing order of a March element can be done in an up (↑), down (↓) way or (↕) if the order is not significant. A March primitive can be a write 1 (w1), write 0 (w0), read 1(r1) and read 0 (r0) that can be performed in a memory cell. The test is composed of March elements represented between (). March tests are the simplest tests (optimal) to detect most of the functional faults.

There are many March tests such as March C-, March SS, March0March1, March LR with BDS and so on. TABLE I compares the test length, complexity and fault coverage of them. 'n' stands for the capacity of SRAM.

Algorithms	Test length	Complexity	Fault coverage
March C-	10n	O(n)	AF, SAF, SOF,CF
March SS	22n	O(n)	AF, SAF, SOF, CF
M0M1	4n	O(n)	SAF, SOF
March LR with BDS	23n	O(n)	AF, SAF, SOF, CF

Similarly, also observed the simulation results for,

- March M 0 M 1:

  {↕ (w0, r0); ↕ (w1, r1)}

- March SS(Simple Static):
  { ↕(w0); ↑(r0,r0,w0,r0,w1); ↑(r1,r1,w1,r1,w0);
  ↓(r0,r0,w0,r0,w1); ↓(r1,r1,w1,r1,w0); ↕(r0)}

- March LR(Realistic Linked) with BDS (Background Data Sequence):

  {↕ (w00);DOWN (r00, w11);

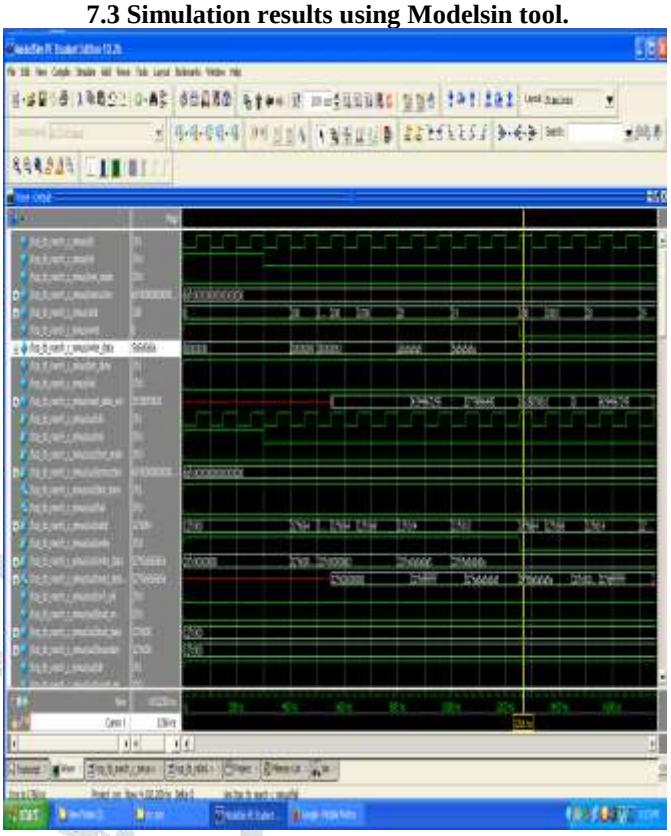
  UP (r11, w00, r00, r00, w11); DOWN (r11, w00);

  UP(r00,w11,r11,r11,w00);
  DOWN(r00,w01,w10,r10);

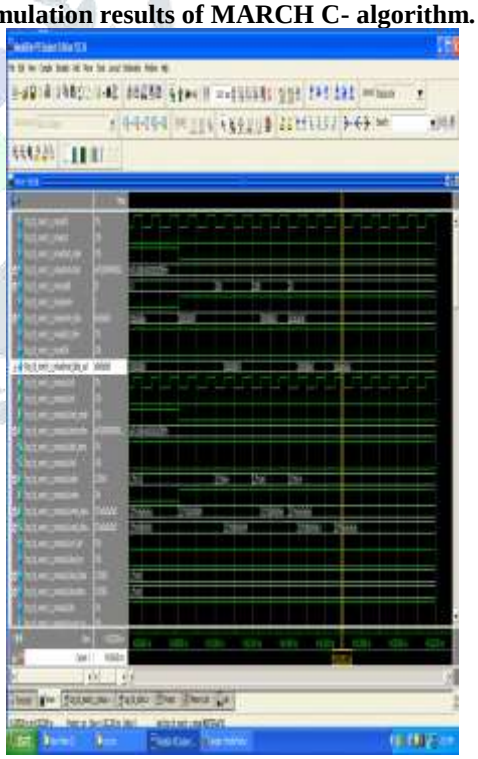
  UP(r10,w01,r01); ↑(r01)}

In above steps, "up" represents executing SRAM addresses in ascending order while "down" in descending order.

7.3 Simulation results using Modelsin tool.



7.3.1 Simulation results of MARCH C- algorithm.



7.4.2 Synthesis report of ProposedBISR

CONCLUSION

An efficient BISR strategy for SRAM with programmable BIST and selectable redundancy has been presented in this paper. It is designed flexible that users can select operation modes of SRAM. The BIAA module can avoid storing fault addresses more than once and can repair fault address quickly.

As these benefits are obtained at low area cost, the proposed memory BISR becomes highly attractive not only for test chips dedicated to new memory and/or process debug but also for integrating it into final products. The function of BISR has been verified by the post simulation. The BISR can work at up to 150MHz at the expense of 20% greater area.

REFERENCES

[1] SemiconductorIndustryAssociation, "Internationaltechnologyroadmapfor semiconductors (ITRS),2003edition,"Hsinchu,Taiwan, Dec.2003.

[2] C. Stapper, A. Mclaren, and M. Dreckman, "Yield model for ProductivityOptimizationofVLSIMemoryChipswithredundancyand Partiallygood Product,"IBMJournalofResearchandDevelopment, Vol. 24, No.3,pp.398-409, May1980.

[3] W.K.Huang,Y.H.shen, andF. lombrardi,"New approaches for repairsofmemorieswithredundancybyrow/columndeletionforyield enhancement,"IEEE TransactionsonComputer-AidedDesign,vol.9, No. 3, pp. 323-328,Mar. 1990.

[4] P.MazumderandY.S.Jih,"Anewbuilt-in self-repairapproachtoVLSImemoryyield enhancementby usingneuraltypecircuits,"IEEE transactionsonComputerAidedDesign,vol.12,No. 1,Jan,1993.

[5] H.C.Kim,D.S.Yi,J.Y.Park,andC.H.Cho,"ABISR(built-inself-repair)circuitforembeddedmemory withmultipleredundancies,"VLSI andCAD 6thInternationalConference,pp. 602-605,Oct. 1999.

[6] Shyue-KungLu,Chun-LinYang,andHan-WenLin, "EfficientBISR TechniquesforWord-Oriented EmbeddedMemorieswithHierarchicalRedundancy ,"IEEEICIS-COMSAR,pp.355-360, 2006.

[7] [C.Stroud,ADesigner'sGuide to Built-InSelf-Test,KluwerAcademicPublishers,2002.

[8] Karunaratne.MandOomann.B,"Yieldgainwithmemoryisr-acase study,"IEEE MWSCAS,pp. 699-702,2009.

[9] I.Kang,W.Jeong,andS.Kang,"High-efficiencymemoryBISRwith twoserialRAstages usingsparememories,"IETElectron.Lett.,vol. 44, no.8, pp. 515-517,Apr.2008.

[10] Heon-cheolKim,Dong-soonYi, Jin-youngPark, andChang-hyunCho, "A BISR (Built-In Self-Repair) circuit for embedded memory with multipleredundancies,"inProc.Int.Conf.VLSICAD ,Oct.1999.

[11] M.Sachdev,V.Zieren,andP.Janssen,"Defect detectionwithtransient currenttestinganditspotentialfordeepsubmicronCMOSICs,"IEEE International Test Conference,pp. 204-213,Oct. 1998.

[12] MentorGraohics, MBISTArchitectProcessGuide, SoftwareVersion8.2009_3,Aug2009, pp. 113-116.

[13] "Efficient Built in self repair strategy for embedded sram using selectable redundancy,"Huamin Cao, Ming Liu, Hong Chen, Xiang Zheng, Cong Wang and Zhihua Wang.

[14] "Memory BIST with Address Programmability,"AymenFradi Michael Nicolaidis, Lorena Anghel.