INTERNATIONAL JOURNAL OF RECENT TRENDS IN ENGINEERING & RESEARCH
Impact Factor 4.101 ISSN -2455-1457
RECENT ADVANCES IN ELECTRONICS & COMMUNICATION ENGINEERING (RAECE-2K17)

Built-In Self-Repair Scheme for Memory Systems

P.Sandeep
Asst.Prof.Dept. Of ECE,
VITS, Hyderabad.

M.Annapurna
M.Tech, Dept. of ECE,
VITS, Hyderabad.

Abstract:Built-in self-repair (BISR) technique is widely used to repair embedded random access memories (RAMs) due to the introduction of more and more system-on-chip and other highly integrated products, for which the chip yield is being dominated by the yield of on-chip memories and repairing embedded memories by conventional off-chip schemes is expensive. This paper presents an BISR generator, which automatically generates register transfer level BISR circuits for repairing RAMs, is based on a redundancy analysis (RA) algorithm that enhances the essential spare pivoting algorithm, with a more flexible spare architecture, which can configure the same spare to a row or a column to fit failure patterns more efficiently.

Key Words: Built-in self-repair (BISR), embedded memory, memory repair, memory testing, redundancy analysis.

I.INTRODUCTION

Built-in self-repair (BISR)[1] isnecessary for system-on-chip (SoC) and other highly integrated products. In modern SOCs, embedded memories occupy significant amount of the chip area [2]. Furthermore, RAMs are subject to severe design rules, such that they are more prone to manufacturing defects. Thus, memories concentrate the large majority of defects. As a matter, Built-In Self-Repair is gaining importance. Built-in self-repair (BISR) technique has been widely used to repair embedded random access memories (RAMs). If each repairable RAM uses one self contained BISR circuit (Dedicated BISR scheme) then the area cost of BISR circuits in an SOC becomes high. This, results in converse effect in the yield of RAMs.1
Fig 1 shows the general architecture for BISR[3]

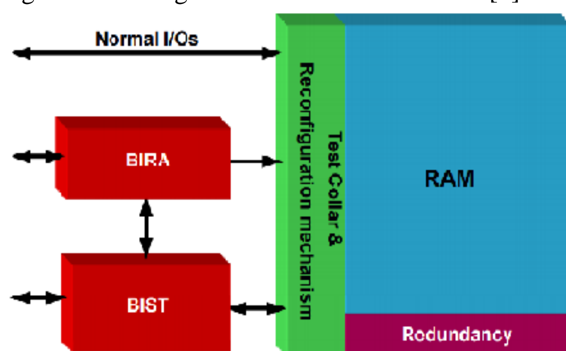

Fig1. General Architecture for BISR

This paper presents a reconfigurable BISR scheme for repairing RAMs. An efficient redundancy analysis algorithm is proposed to allocate redundancies of defective RAMs. In the BISR, a reconfigurable built-in redundancy analysis (BIRA) circuit is designed to perform the redundancy algorithm for various RAMs.

TheBISR is based on an RA algorithm that enhances the essential spare pivoting (ESP) algorithm, with a more flexible spare architecture, which can configure the same spare to a row or a column to fit failure patterns more efficiently. The proposed BISR circuit is small and it supports at-speed test without timing penalty during normal operation.

II.PROPOSED BISR

A. BISR FEATURES

The proposed BISR generator targets large embedded memories or stand-alone memories that require redundancy repair. It takes the memory specification and then generates the RTL BISR circuit, which is a logic circuit that canbe synthesized using a commercial available tool. The BISR circuit has a scalable architecture and it implements an enhanced version of our ESP [4] RA algorithm for efficient 2-D repair. The BISR circuit can concurrently analyze a memory for repair (spare allocation) during at-speed test by the BIST circuit. There is no test time overhead and only a little logic area overhead.

B. BISR Architecture and Operation

Fig. 2 shows the architecture of the proposed memory built-in redundancy-analyzer MBIRA) scheme, whose details follow.

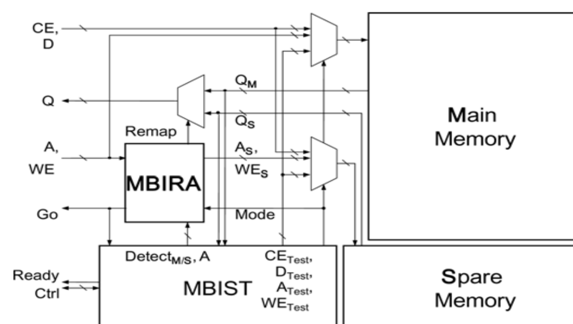

Fig. 2 Proposed BISR

MBIST Operation:

The BIST is used to test main memory and spare memories for faults. For the field reset-triggered repair, the BIST executes the default March-CW test algorithm [5] after each reset, by connecting some control pins (Ctrl or BISTEna) to VDD or ground.

After detecting faulty addresses, it gives them to MBIRA circuit.

MBISR RA in Test Mode:

During RA, the MBIRA first records the unusable faulty spare elements and then allocates spare elements according to the faulty rows/columnsof themain 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. When the spare elements are notenough for repairing the faults, the Go signal is de-asserted toabort the MBIST. Otherwise, when the test is done, the Readyand Go signals are asserted.

Whenever the MBIST begins testing, the MBIRA starts theRA process by detecting the Mode signal. During RA, theBIRA checks the 2-bit DetectM/S signals from the MBISTfor the main and spare memory faults, respectively, and analyzes each faulty address from a Fail within the same clockperiod.

The MBIRA can use multiple clock cycles to pre-analyzethe faulty addresses without suspending the BIST, because itgets the addresses from the BIST before the addressed dataare read and determined faulty.

MBIRA Address Remapping in Normal Mode:

In thenormal mode, the MBIRA remaps the faulty main memoryaddresses to those of the spare memory that have been determinedduring the RA process. Therefore, the spare memoryhas the same chip-enable (CE) and data-in (D) signals as themain memory, but it receives the remapped address (AS) andwrite-enable (WES) signals from the MBIRA. The MBIRAalso can select the correct spare data-out (QS) to replace thefaulty data-out (QM) for the output (Q).

If a processor is addressing fault-free data in the memory, the BIRA de-activates WES and switches Q to QM, i.e., itdisables the spare memory. Only if the MBIRA detects faultymain memory address on A from the processor, it concurrentlyremaps A to AS according to the previous RA result. In thatsituation, the MBIRA activates WEs in the same clock periodwhen we perform the Write operation, or assert remap to selectthe spare QS during the Read operation. Although connectingCE directly to the spare memory wastes additional power dueto the unnecessary read, it avoids timing penalty for all theRead operations.

The MBIRA enters the normal mode by detecting the deassertion of Mode signal that switches the memory input multiplexer from the test to the normal inputs. In fact, the BIRA can switch between test and normal modes arbitrarily according to the mode signal.

BIRA ALGORITHM

A. ESP Algorithm and ProposedImplementation:

ESP is a greedy RA algorithm with a specified threshold. During testing, ESP marks essential-repair lines (rows and columns whose number of faults exceeds the threshold) and then ignores the incoming faults on those essential lines. The ESP with threshold 2 will remap the faulty address with the spare memory locations starting from more number of faults in a row or column. Figure 3 shows an example fault map to illustrate the algorithms.

Fig.4 is an ESP example with threshold 2, two spare rows and two spare columns, where rows 1, 5, 6 and column 5 are essential-repair lines and cell (5, 5) has an orthogonal fault. Due to the lack of a spare row for the essential-repair row 6, this case will fail during the spare allocation phase, which allocates spare rows, columns or both to specific table items after testing.

Column Address / Row Address

	0	1	2	3	4	5	6	7
0								
1			a	b	f	g		
2								
3								
4								
5						c	d	e
6						h	i	j
7								

Fig. 3Memory block with defect sequence.

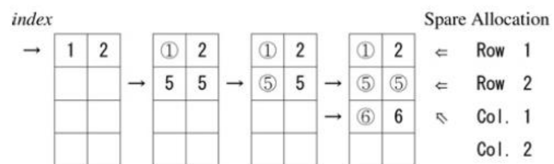

Fig.4 ESP with threshold 2, two spare rows and two spare columns.

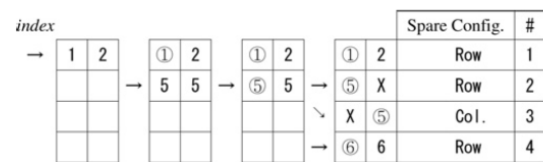

Fig. 5 ESP with threshold 2 and 4 pivoting spare elements.

Fig.5 has the same faults as in Fig.4, but now each spare element can be used to replace a faulty row or a faulty column. Therefore, the faults become repairable. Every spare is initially in the default configuration and configured to repair a row/column when one of the address-coordinates is marked as an essential-repair row/column.

For example, Spares 1, 2, and 4 are configured as rows, and Spare 3 is configured as a column when their essential-repair flags (circles) are set. Determination of default configuration depends on the memory structure and failure diagnosis. If an

address is marked simultaneously as an essential-repair row and an essential-repair column [e.g. Address (5, 5) in Fig.3], the address is split into two items (e.g., spare 2 and 3 in Fig.5, where "X" means don't-care) to avoid configuration conflict of the spare. Also when there is a single fault in a row or column instead of remapping entire row or column to spare element, inverting that single fault bit will be efficient. The proposed architecture is implemented in Xilinx and simulation result is shown Fig 6,RTL schematic is shown Fig 7.

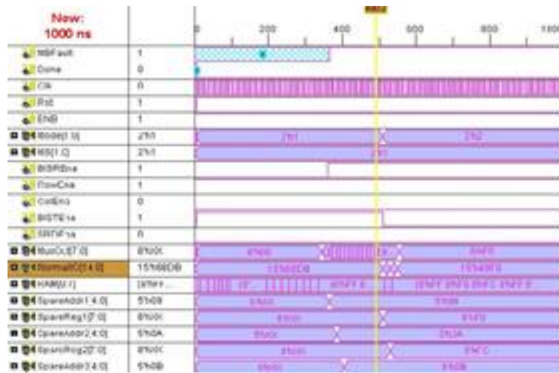

Fig. 6 Top Module Waveform

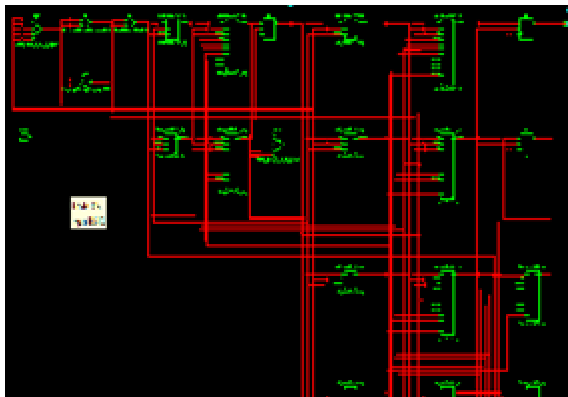

Fig 7 RTL Schematic

III.CONCLUSION

Proposed BISR circuit successfully repair the faults using reconfigurable spares. This Architecture gives a cost-effective BISR generator, which enhances the BIST generator with repair option for general embedded memories. It generated synthesizable RTL BISR circuits with little area overhead and high repair rate.It enhanced the ESP RA algorithm for traditional 2-D or block redundancy repair with the proposed pivotal or configurable spare architecture, which can allocate the spare elements in row, column that the address bits can define. It also removed the spare allocation phase of ESP for lower area overhead and shorter analysis time .This BIRS scheme can also be applied to multiple memories on a chip.

REFERENCES

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

[2]Tsu-Wei Tseng, Jin-Fu Li, and Chih-Chiang Hsu "ReBISR: A Reconfigurable Built-In Self-Repair Scheme for Random Access Memories in SOCs" in IEEE transactions on very large scale integration (vlsi) systems, vol. 18, no. 6,pp.921-932, June 2010.

[3] Jhin-Fu li,Memory Built in Self Repair:Advance Reliable systems(ARES) Lab,Department of Electrical Engineering,National Central University,Jhongli, Taiwan

[4] C.-T. Huang, C.-F. Wu, J.-F. Li, and C.-W. Wu, "Built-in redundancy analysis for memory yield improvement," *IEEE Trans. Reliab.*, vol. 52,no. 4, pp. 386–399, Dec. 2003.

[5] C.-F. Wu, C.-T. Huang, K.-L. Cheng, and C.-W. Wu, "Fault simulation and test algorithm generation for random access memories," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 21, no. 4, pp. 480–490,Apr. 2002.