RE-CONFIGURABLE BUILT IN SELF REPAIR AND REDUNDANCY MECHANISM FOR RAM'S IN SOCS

Ravichander Bogam[1], M.Srinivasa Reddy[2]
[1]Department of Electronics and Communication Engineering
St. Martins Engineering College, Hyderabad, India
[2]Department of Electronics and Communication Engineering,
MLRIT, Dundigal, Hyderabad, India

Abstract
The main objective of this paper is to detect and repair the faults for Random Access Memories in SOCs, by using a Built In Self Repair BISR Scheme a complex IC that integrates the major functional elements of a complete end product into a single chip is known as System on Chip (SOC). With the trend of SOC technology, high density and high capacity embedded memories are required for successful implementation of the system. Typically, many RAMs with various sizes are included in an SOC and they occupy a significant portion of the chip area. RAMs have more serious problems of yield and reliability than any other embedded cores in an SOC so, Keeping the memory cores at a reasonable yield level is thus vital for SOC products. For such purpose, memory designers usually employ redundancy repair logic using spare rows and/or columns to improve the yield. To maximize the yield, redundancy analysis is necessary. Conventionally, redundancy analysis (RA) is performed on the host computer of the ATE. Embedded memories are harder to deal with Automatic Test Equipment (ATE). The Re-BISR technique is a promising and popular solution for enhancing the yield of memories with the redundancy logic

Keywords: SOC, BIRA, ATE, Re-BISR & Redundancy algorithm

I. INTRODUCTION

The advancement in IC technology enabled the integration of all the components of a system into a single chip. A complex IC that integrates the major functional elements of a complete end product into a single chip is known as System on Chip (SOC). It enables the designers to move everything from board to chip. SOC incorporates a portable / reusable IP, Embedded CPU, Embedded Memory, Real World Interfaces, Software, Mixed-signal Blocks and Programmable Hardware. Reduction in size, lower power consumption, higher performance, higher reliability, reuse capability and lower cost are the benefits of using SOC.

With the trend of SOC technology, high density and high capacity embedded memories are required for successful implementation of the system. Memories are the widely used cores on the SOC products. Typically, many RAMs with various sizes are included in an SOC and they occupy a significant portion of the chip area. But, due to the continual advances in the manufacturing process of IC, provide designers the ability to create more complex and dense architectures and higher functionality on a chip. Although it benefits the end user, but these manufacturing process advances are not without limitations. In particular, embedded high density memories in combination with these process limitations can result in poor over all yields. That is, RAMs have more serious problems of yield and reliability than any other embedded cores in an SOC. Depending upon the application and design, much of the low yield can be attributed to faults in the memory. Keeping the memory cores at a reasonable yield level is thus vital for SOC products. For such purpose, memory designers usually employ redundancy repair logic using spare rows and/or columns to improve the yield.

However, redundancy increases the silicon area and thus has a negative impact on yield. To maximize the yield, redundancy analysis is necessary. Conventionally, redundancy analysis (RA) is performed on the host computer of the ATE if it is an online process or on a separate computer if it is an offline process. Either way it is time consuming since RA algorithms are complicated and the memories that implement the redundancies are usually large. Moreover, embedded memories are harder to deal with Automatic Test Equipment (ATE). The BISR (Built in Self Repair) technique is a promising and popular solution for enhancing the yield of memories with the redundancy logic. Furthermore, redundancies of memories are not just for defects, redundancies can also be used to recover yield due to process variation in addition to yield recovery for defects.

II. PROPOSED REDUNDANCY MECHANISM

A. *Memory Organization*

As SOC size is shrinking, the major area on SOC is occupied by embedded memories. Thus memories in chip will decide the yield of the SOC. Increase in yield of memories in turn increase the yield of SOC. SOC yield increases from 2% to 10% by improving the memory yield from 5% to 10%.The techniques used for yield improvements in memories are Built In Self Test (BIST) and BISR. Many algorithms are proposed for spare allocation for defected memories. The redundancy is of 1D (only spare row or column) or 2D(spare row and spare column) . The redundancy analysis (RA) algorithm for 1D is simple compare to 2D. a reconfigurable MBIST architecture is proposed by adding some data processing unit and address processing unit. The redundancy organization memory is divided into various segments. In which spare row and columns are used differently. Spare rows are used to replace entire row in the memory and the columns are divided in several spare column groups. Here the access time and area cost is induced due to additional multiplexers.

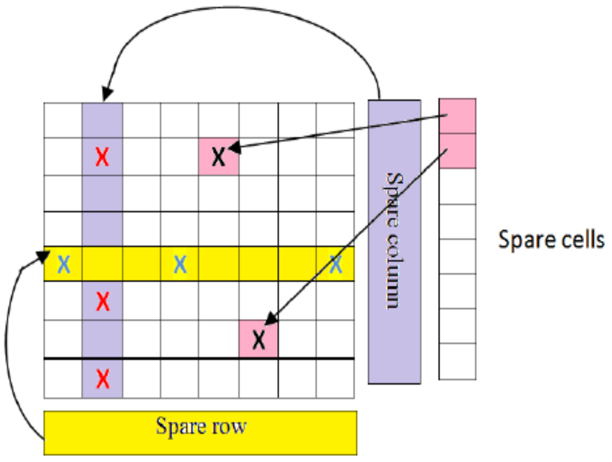

Fig: 1 The proposed redundancy organization

The figure shows an example of 8*8main memory modules along with 1 sparerow,1 spare column and spare cells .since most of the memory faults are single cell defects, here spare cells are used or better utilization of spare elements. the row/column having multiple defects is remapped with corresponding spare row/column. the single defects in the main memory are remapped with spare cells. by this redundancy organization the area of spare is efficiently utilized

B. *Built in Self repair(BISR)*

The memory is tested by external test hardware or by on chip dedicated hardware (memory BIST). The second testing strategy is the preferred method for embedded memories. The memory BISR (MBISR) concept contains an interface between memory BIST (MBIST) logic and redundancy wrapper for storing faulty addresses. This allows using already existing MBIST solutions. The MBIST controller output must provide three signals to the wrapper logic during test.

· A fail signal to store data in the fuse register

· The expected data that is compared to the results of RAM data

· The failing address

Figure 2 shows the block diagram of a MBISR scheme for a RAM, which consists of four major components

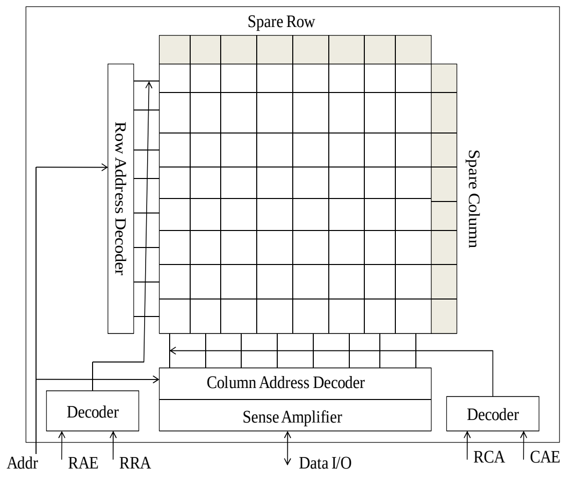

Fig: 2 Conceptual diagram of an 8*8 bit oriented repairable RAM with one spare row and one spare column

1) **Repairable RAM:** A RAM with redundancies and reconfiguration circuit is called as a repairable RAM. Figure 2 depicts an example of an 8*8 bit-oriented RAM with 1 spare row and 1 spare column. If a spare row is allocated to replace a defective row, then the row address of the defective row is called row repair address (RRA). Then a decoder decodes the RRA into control signals for switching row multiplexers to skip the defective row if the row address enable (RAE) signal is asserted. The reconfiguration of the defective column and the spare column is performed in a similar way, i.e., give a column repair addresses (CRA) and assert the column address enable signal to repair the defective column using the spare column.

2) **BIST Circuit:** It can generate test patterns for RAMs under test. While a fault in a defective RAM is detected by the BIST circuit, the faulty information is sent to the BIRA circuit.

3) **BIRA Circuit:** It collects the faulty information sent from the BIST circuit and allocates redundancies according to the collected faulty information using the implemented redundancy analysis algorithm.

As figure 2 shows, the overall RAM BISR flow is roughly described as follows. Firstly, the BIST tests the repairable RAM. If a fault is detected, then the fault information is stored in the BIRA circuit. Then, the BIRA circuit allocates redundancies to replace defective elements. As soon as the repair process is completed, the repair signatures are blown in the fuse box. Subsequently, the repair signatures are loaded into the fuse register first and then are shifted to the repair registers (i.e., registers in the wrappers for storing RRA, RAE, CRA, and CAE data) in normal operation mode. Finally, the repairable RAM can be operated correctly.

Normally memories are dense and covers a large portion of the chip area, thus dominate the yield of the chip. Keeping the memory cores at a reasonable yield level is thus vital for SOC products. For such purpose, memory designers usually employ redundancy repair—using, e.g., spare rows and/or spare columns of cells—to improve the yields

III. RECONFIGURATION TECHNIQUES

Redundancy analysis and repair procedures are interrelated. During testing, when failures are identified and located, a redundancy analysis procedure determines which failures can be repaired using the given redundant rows and columns. On the basis of this redundancy analysis, either a hard or a soft repair procedure is executed.

A. **Hard repair** In general, hard repair uses fuses, antifuses, or laser programming to disconnect rows and columns with faulty bits and replace them with redundant rows or columns. This method, long used by commodity memory vendors, has been adopted by ASIC and SOC vendors and is currently the most widely used method for stand-alone as well as embedded memories.

B.**Soft repair** Soft repair uses an address-mapping procedure to bypass the faulty address location. This scheme links BIST and power-on, so that every time power is switched on, memory is tested via BIST. During this testing, the addresses of all failing locations are stored separately, and an address mapping procedure maps them onto redundant fault-free addresses. Although a dedicated finite-state machine serves to implement an address-mapping procedure, it can also be implemented efficiently through an on-chip microprocessor or other logic cores.

IV. ARCHITECTURE OF REBISR

Figure 3 shown below is the simplified block diagram of the proposed ReBISR scheme for repairing multiple RAMs in an SOC. Four repairable RAMs with various sizes having different number of redundancies are considered in our proposed ReBISR scheme as shown in the below figure. All these RAMs are word oriented memories and there configurations are as listed in the below table

RAM No	Data Width * Memory Depth	No. of Spare Rows	No. of Spare Columns
RAM 0	16 * 32	2	2
RAM 1	32 * 64	2	3
RAM 2	128* 64	3	2
RAM 3	256*64	0	0

Table 1: Configurations of RAMs

The table 3.2 shows the stuck-at-faults in the four repairable RAMs at their respective fault row and fault column locations as shown below.

RAM Number	Fault Row	Fault Column	Stuck-at-fault
RAM 0	6,7 14	18 28, 29	0 1,1
RAM 1	15, 16 28	43 61	1,1 0
RAM 2	-	-	-
RAM 3	243	31	0

Table 2: Location of stuck-at-faults

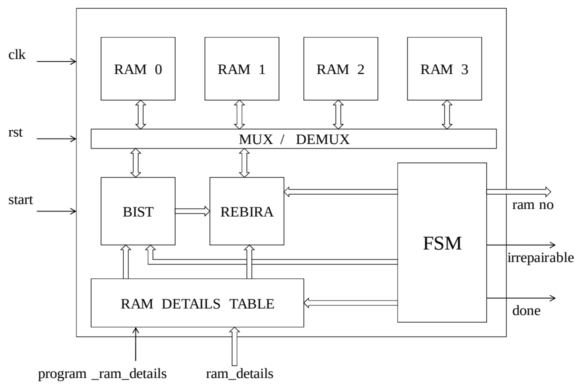

Fig 3: Block diagram of the proposed REBISR scheme for repairing multiple RAMs

The block called RAM details table is used for storing the configurations of RAMs which includes the memory data width, memory depth, number of spare rows and number of spare columns. RAM details table is of size 4*16. FSM is the main block that acts as a controller for generating the control signals during testing and repairing processes.

The overall RAM ReBISR flow is described as follows. Before the BIST circuit and the ReBIRA circuit start testing and repairing the RAMs, the RAM configurations (details) should be known by these two circuits. This is done by the FSM, where it generates the necessary control signals that are required for sending the RAM configurations from RAM details table to BIST and ReBIRA circuits. Because, the memory depth and memory data width are required by the BIST circuit for testing the RAM and the number of spare rows and number of spare columns are required by the ReBIRA circuit to perform the analysis before repairing. The process of entering the RAM configurations into the RAM details table is described as follows. At this stage, the ReBISR completed the repairing of all the RAMs. Hence, the output did results high and ReBISR remains in the same state.

When reset is high, all the locations in the RAM details table are filled with zeroes. Else, if the signal program_ram_details is high, the RAM details are entered into RAM details table through the pin ram details using the write pointer. As the details are entered one by one, the write pointer is incremented by 1.Once the RAM details table is full, the write pointer stops incrementing and holds the value.

Once the RAM details table is full, the BIST and ReBIRA circuits start the testing and repairing processes of RAMs one by one. If the BIST circuit detects a fault, then the fault information is exported to the ReBIRA circuitry, and then the ReBIRA performs redundancy allocation on the fly using the rules of the implemented redundancy algorithm. The redundancy algorithm implemented in our scheme is Range Checking First Algorithm (RCFA). The ReBIRA allocating redundancy on the fly means that the redundancy allocation process and the BIST process are performed concurrently. The proposed ReBIRA scheme uses a local bitmap (i.e., a small bitmap) to store fault information of the faults detected by the BIST circuit. The bitmap or the fault table present in the ReBIRA circuitry is of size 4*64 in our proposed REBISR scheme. Once the local bitmap is full, the BIST is paused and the ReBIRA allocates redundancies according to the faulty information. After the ReBIRA allocates a redundancy to repair a corresponding faulty row or column, the local bitmap is updated and the BIST is resumed. This process is iterated until the test and repair process is completed. The repair signatures from the ReBIRA circuit are then sent to the repair registers that are present in the repairable RAMs. Repair signatures include repair register data (defective row/column address), repair register address (the address location in the row/column repair registers for storing the defective row/column address) and repair register write signal. The repairing procedure involves the entering of the repair register data in the repair registers. When the repair register write signal is high, then the repair register data is written in the repair registers at the address location specified by the repair register address. The BIST tests the RAMs once again (after the testing and repairing processes) to ensure that there are no faults present.

Whenever the memory is accessed later i.e. after repairing, when the defective address is arrived, then the address decoder decodes the row/column repair address to control signals for switching row/ column multiplexers to skip the defective row/column. And, the control is immediately transferred to the relevant location either in the spare row/spare column since there is one to one correspondence between repair registers and spare elements. This is nothing but the address mapping procedure. The ReBISR FSM is shown in the below figure and the ReBISR flow using states is roughly described as follows.

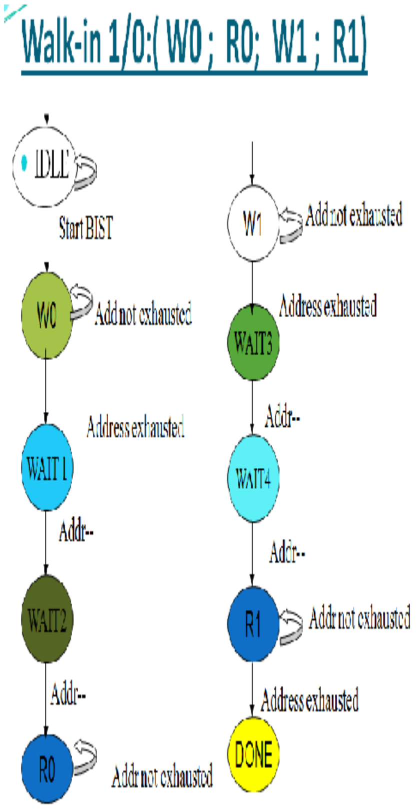

Fig 5: Re-BISR FSM

4.1: Re-BIRA

It collects the faults which are exported by the BIST and use the local bit map (fault table). If the local bitmap is full, BIST paused and ReBIRA allocates redundancies according to the fault information. If the local bitmap is full, BIST paused and ReBIRA allocates redundancies according to the fault information. This process is continues for all the RAMs

1	0	1	0	0	1
0	0	0	0	0	0
1	0	1	0	1	0
0	0	1	0	0	1
1	1	0	0	1	0
0	1	1	0	0	0
1	0	1	0	0	0
0	1	1	0	1	1

Fig 6: Local bit map

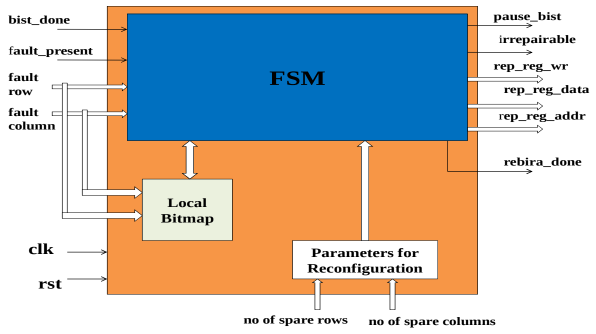

Fig 7: Block diagram of Re-BIRA

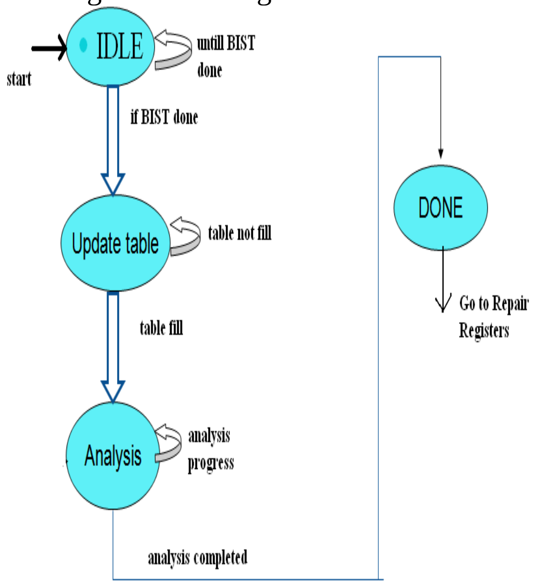

Fig 8: FSM for Re-BIRA

Re-BIRA uses Range Check in first algorithm and is shown below

NFRE > NFCE = Allocates Column
NFRE < NFCE = Allocates Row
NFRE = NFCE = Available Spare Elements
If spare elements are exhausted the RAM is unrepeatable

BIRA Schemes	RAM Type	BIRA Architecture	No. of slices used	Area Cost	Time Cost	Clock Cycles
ReBIRA	WOM	Shared	2850	Low	Long	3,53,200 ns

V.CONCLUSION

A reconfigurable BISR scheme for repairing multiple repairable RAMs with different sizes and different numbers of redundancies has been presented. An efficient BIRA algorithm for 2-D redundancy allocation has also been introduced. The BIRA algorithm has been realized in a reconfigurable BIRA hardware, such that it can support different RAMs. Experimental results show that the ReBISR scheme incurs low area cost when compared with the dedicated BISR. Also, the reconfigurable BISR scheme has greater flexibility than the dedicated BISR scheme as the former one supports the repair of multiple memories. Therefore, our ReBISR scheme has low area cost compared with the other BISR scheme for general applications.

References

1. John F.Wakerly, Digital Design Principles and practices, Fourth Edition.
2. Samir Palnitkar, Verilog HDL A guide to Digital Design and Synthesis, SunSoft Press 1996.
3. Kim, Y. Zorian, G. Komoriya, H. Pham, F. P. Higgins, and J. L. Lweandowski, “Built in self repair for embedded high density SRAM,” in *Proc. Int. Test Conf. (ITC)*, Oct. 1998, pp. 1112–1119.
4. M. Nicolaidis, N. Achouri, and S. Boutobza, “Optimal reconfiguration functions for column or data-bit built-in
5. self-repair,” in *Proc. Conf. Des., Autom., Test Eur. (DATE)*, Munich, Germany, Mar. 2003, pp. 590–595.
6. M. Nicolaidis, N. Achouri, and S. Boutobza, “Dynamic data-bit memory built-in self-repair,” in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des. (ICCAD)*, San Jose, CA, Nov. 2003, pp. 588–594.
7. P. Ohler, S. Hellebrand, and H.-J. Wunderlich, “An integrated built-in self-test and repair approach for memories with 2D redundancy,” in *Proc. IEEE Eur. Test Symp. (ETS)*, Freiburg, May 2007, pp. 91–99.
8. P. Ohler, S. Hellebrand, and H.-J. Wunderlich, “An integrated built-in self-test and repair approach for memories with 2D redundancy,” in *Proc. IEEE Eur. Test Symp. (ETS)*, Freiburg, May 2007, pp. 91–99.