Design and Implementation of Repair-aware Test Flow for Multi-Memory

Gang Wang and Huajun Chen

Key Laboratory of Computer System and Architecture, Chinese Academy of Sciences, Beijing 100190
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China 100190
Loongson Technology Corporation Limited, Beijing, China 100190
wanggang@ict.ac.cn

ABSTRACT

A complex SoC typically consists of numerous of memories in today's digital systems. This paper presents a test/ repair flow based on memory grouping strategy and a revised distributed BIST structure for complex SoC devices. A gated selecting method is added to the distributed BIST structure. Also, this paper for the first time proposes a robust post repair stage based on BIRA and memory grouping in test flow. Simulation results by mathematical method show that the proposed test flow has achieved a significant increase in yield of memories.

KEYWORDS

Test Flow, Test, BISR, Multi-Memory

1. INTRODUCTION

Today's rapid improvement in Nano-scale process technologies enables us to create complex chips, such as system-on-chip(SoC) design. However, as CMOS technologies scale down, some physical phenomena show up their impacts on circuit correct function, especially in memory areas. Moreover, memory cores are usually designed with the most aggressive design rules in the design areas. Memory core is the most prevailing core in such complex SoC design[1].Also, the cost to repair failures of embedded memories in a SoC is more expensive than that of commodity memories because a relatively large die is wasted. Memory cores usually occupy a significant portion in the chip area and dominate the manufacturing yield of the chip. Due to the challenge of the design of automation and the complex fabrication process for combination memories and logic as well as the large die size, SoC suffers from relatively lower yield, and necessitates yield optimization technologies [2]. To improve the yield of memories as well as chip, efficient testing and repair technologies for memory blocks in SoC are essential.

So far Built-In Self-Test(BIST) and Built-In Self-Repair(BISR) are two dominant methods for memories testing and repair[3]. A lot of works had contributed to the BIST and BISR[4], and many of them had been as the industrial standard. Many of BIST structures have been proposed to reduce the power consumption and test time as well as the area of the SoCs. To enable BIST

for memory memory arrays are usually equipped with spare elements, either spare rows or spare columns, and external tester has been used to test the memory arrays and configure the spare elements to improve the yield of memories. In [1] and [5], the authors mentioned that there are three stages to test the memory core with BISR scheme, that is (1) test (with BIST circuity), (2) diagnosis/repair (with built-in repair analyzer) and (3) re-test (with BIST circuity). However, this paper proposes another test flow which includes an important stage, post repair, and has made some changes in the state-of-the-art test flows. It can significantly improve the yield of the memory blocks. In [6], the author mainly concerns on how to minimize the test time and area cost by using automatic generation of memory BISR circuity and uses memory grouping techniques which only used for BISR to manage the BISR circuits.One contribution of this paper is that we present a grouping method to go through the overall test flow including BIST and BISR, and another important contribution is that we use a gated selecting technology, gating off some unnecessarily memories to reduce the power consumption, to select some memory blocks that need to re-test.

Section 2 presents the details of test flow and memory grouping strategy used in this paper. Section 3 shows the works related to the BIST. Section 4 illustrates implementation details of BISR. Section 5 proposes the stage re-test and post repair used in this paper. Section 6 shows simulation and analysis by mathematics method. Section 7 is the conclusion of this paper.

2. PRELIMINARIES

In this section, we introduce the test flow and memory grouping strategy.

2.1. Grouping technology

In order to reduce the test time and make test and repair easier, the memories under test are divided into more than one test sessions[6]. The memories in the same test session can be tested simultaneously, using a distributed BIST structure. Fig.1 shows an example of prime results for 8 memories which are divided into three test sessions: TS1={M1, M2, M3}, TS2={M4, M8} and TS3={M5, M6, M7}. Since we have a distributed BIST control scheme, memories in different test session can be tested simultaneously and that can significantly reduce the test time. Since long distance can cause unnecessarily coupling faults between the neighboring lines and hence increase the power consumption, it is necessary that we should place the distance at a top priority when grouping memories[8] and the physical distribution of the blocks on a floor plan of devices is a very important criterion. Another important reason is that putting neighboring memories together is very convenient in the light of BIST controlling.

Also, it should consider the power constraint of the same type of BIST control scheme in each test session. The goal of memory grouping is to select as much as memories into different test sessions under the constraints of both the minimal distance and power consumption. It can be achieved through Greedy Algorithm. The main idea of the algorithm is showing below simply .

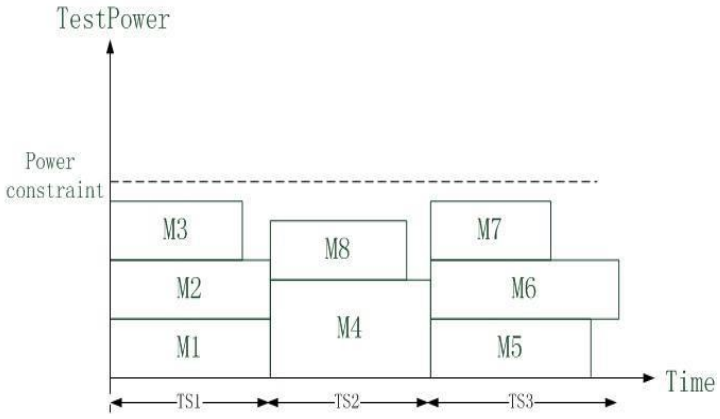


Fig. 1 . An example of test grouping results of 8 memories

Initially, it needs to know power consumption of all memories under test before it performs BIST. For each memory, compute the distance to each distributed BIST and then store them into the table. From the table, select memory blocks whose distance is the minimal, as long as the corresponding group is under the power constraint after adding this memory's power consumption getting from the table, and assign to the corresponding group, and then delete this distance from the table. Again choose minimal distance from the table, and repeat the procedure until all memories are located. In this memory grouping, we mainly consider the distance, between the memory and

BIST structure, and the power consumption which each test session can afford. We use a shared BIRA instead of distributed BIRA in session IV. A group of memories are executed under the control of BIST, in parallel, and based on two important conditions: (1) The BIST power dissipation of each test session must be less than that of the BIST device's limits, (2) the memory blocks in each test session have to be physically adjacent on a floor plan,that is, two-dimension (2D). Also,grouping such memory blocks can contribute to reduce the power consumption of control lines that connect these memory blocks together in the same test session[9].

2.2. Test flow

In this section, we mainly model a test flow, as Fig. 2 shows, a revised memory test flow, that differs from the typical memory BISR flow[1][10]. In this revised flow, it includes four stages, Test prepare, BIST and repair, Retest, Post repair. Before we discuss the proposed flow, it is essential to depict the framework of test. Fig. 3 depicts the memory cluster of a dedicated test design including spare rows and spare columns in each memory. This circuit differs from that of [1].It consists of many separated memory blocks, and each has a wrapper (including a test collar and a remapping circuit), instead of treating a system as a whole memory block. And, this circuit control structure is made up of distributed BISTs and a shared BIRA, both are connected by the stream of faulty information. When the BIST circuit tests and detects a fault of memory, it sends the faulty information to the BIRA. The BIRA circuit receives the detected faulty information and stores the information, after analysis it according to the RA

(Redundancy Analysis) algorithm, allocates redundancies. The control part communicates with outside control signal by the JATG ports[11]. The most striking parts of the circuit are distributed BISTs and a shared BIRA, which will be discussed below.

Now, we discuss the revised memory test flow. In stage1, test prepare, it collects the physical information of each memory. The two most important information in this paper is memory's location in SoC and test power. Subsequently, the test flow performs a memory grouping algorithm to determine the test sequence of all memories under memory distance, between memories and BIST in each test session, and power constraint. All memories are divided into an unknown test session. Each test session is controlled by one distributed BIST, but all sessions shared one BIRA. In addition, we also need information such as memory size and memory assignment.

In stage2, BIST and repair, the BIST structure tests memories within test session got from stage1 simultaneously. Once a fault is detected, it performs the BIRA (Built-In Repair-Analysis), which uses the bitmap, and allocates redundancy according to the bitmap[12]. If the related test session is not completed, that is, BIST is not done in one test session, this process will be iterated. If there are no faults detected in all memories of one test session, then it symbols a sign of repaired or no-faulty.

In stage3, Re-test, this stage mainly make sure that the repaired memories from stage2 in some test session are not faulty in the same test session. It makes the memory system more reliable.

In stage4, Post repair, it can improve yield of memories. Post repair mainly focuses on a whole test session. If there are some unrepairable memories which are due to lacking of redundancies in itself in stage2, or some memories which had been repaired fail to pass the re-test stage, then they will become a repaired memory as long as there are redundancies in its own test session.
Some memories, which have redundancy units after repaired, can allocate the redundancy to another memory which needs to be repaired in the same test session according to the information of memory grouping. If there are unrepairable memories after all redundancies have been fully used in one test session, then these memories is utter unrepairable. This test flow can significantly improve memory's yield since it treats one test session as a whole to test and repair.

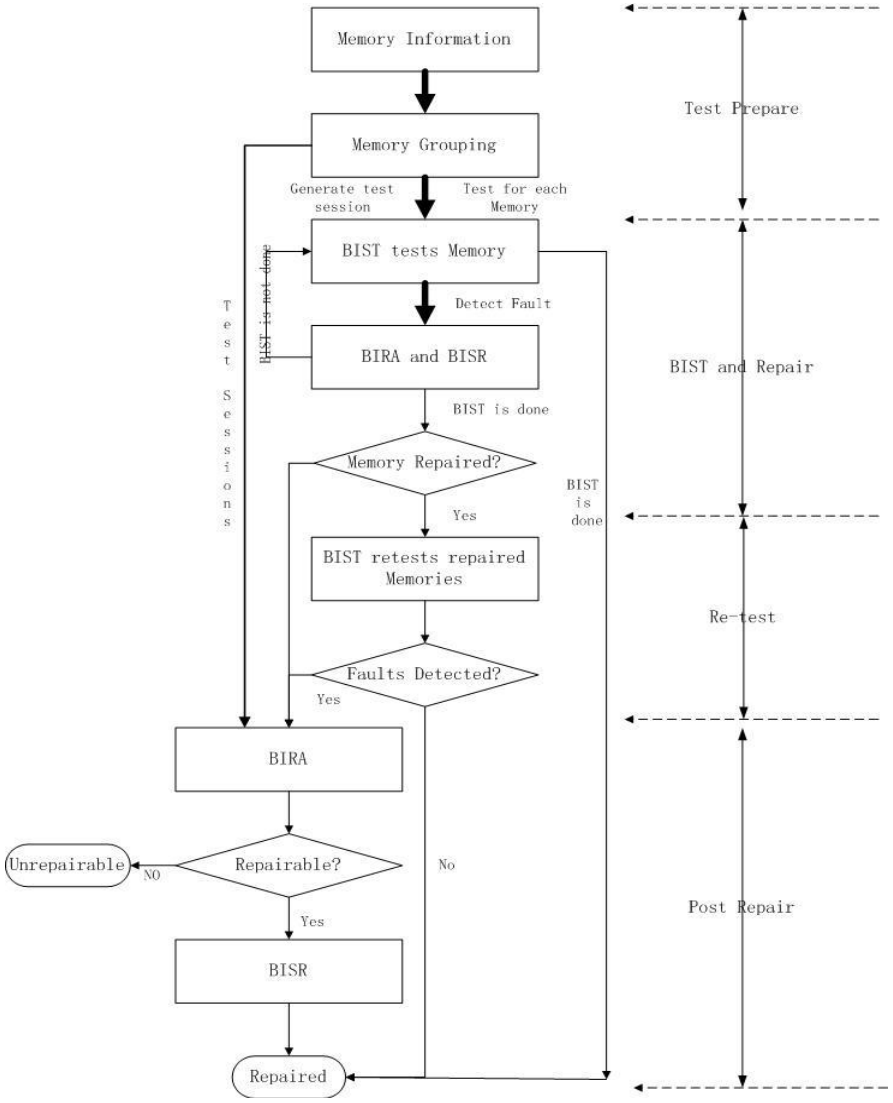


Fig.2. A revised memory test flow

3. RELATED TO BIST

BIST (Built-In Self-Test) is a main strategy in the memory test. Also, it costs a lot of time to execute test procedure and power, especially in complex VLSI devices[9]. The major case of concern is the test power and time during BIST execution. First, BIST results in considerable high switching activity rate, therefore,causing higher power consumption than that of normal operation. Second, BIST must test all elements of the memory,especially nowadays using large scaled and high density memory in embedded chip, resulting in the longer test time, regarding to executing BIST on device under test. Therefore, it is essential to take a overall device level BIST realization into consideration, instead of a single memory block.

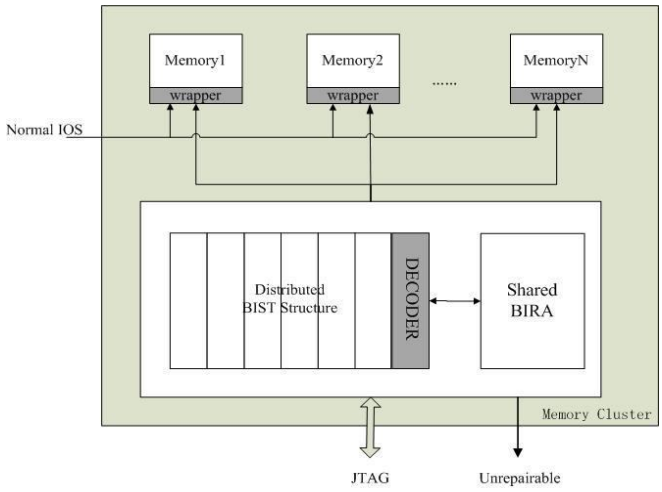


Fig.3. Overall framework of a dedicated test design

To solve those problems, a new BIST structure should be introduced to illustrate present difficulty situations. In this paper, a distributed BIST structure will be presented which serves for shared BIRA to improve the yield.

In BIST control structure mentioned here, seeing Fig. 3, it is a distributed architecture which can be well suitable to all memory blocks. Using grouping strategy and test session we mentioned in session II, all the BIST executions can be achieved simply by a few instructions through JATG ports. As different test sessions have different BIST structures which are connected by uniform protocol for communication between BIST controller and its embedded memory blocks,therefore, all test sessions which consist of various number of memory blocks can be executed in parallel , thus it can significantly reduce the test time.

For distributed BIST structure, BIST scheduling is an analysis process, deciding that which memory block belongs to which test session according to the grouping strategy mentioned at section II. In order to fully illustrate the process of a complex device, it should take three different elements into considerations: the structure information of blocks, the circuit hierarchy and clock domains [9].The clock domain is out of the scape of this paper. We can applicate the grouping strategy according to the structure information of memory blocks and the circuit hierarchy. Since only through grouping can we attain various test sessions, all following works are based on the test session as well as the distributed BIST structure. Also the grouping can guarantee that the power dissipation is lower than that of without using the grouping strategy, since when it separates memory blocks into test sessions , it takes the power consumption into considerations during test partitioning and scheduling.

The most striking part of this BIST structure is that it contains a encoder circuit which is very useful to the BIRA process. Before the distributed BIST starts to test, the JTAG ports send some instructions to the encoder to decide which distributed BIST performs the BIST. This technique[13] depends on gating off some portions of distributed BIST structure during BIRA and BISR, and , it can also be used during BIST when some portions do not need to be tested, including that opening some distributed BIST to start test and disabled some portions. In this

paper, a gated encoder scheme is presented. Here, in order to easily illustrate these processes, an example will be presented briefly. The overall distributed BIST structure is split into three identical distributed BISTs, as Fig. 4(a) shows. In order to clearly present this structure, they will be depicted them at different levels. Fig. 4(a) shows the overall BIST structure splits into three sub-BIST structures. The RUNBIST pin is connected to the input of each distributed BIST, and the outputs of those BIST are connected to those corresponding test sessions. A simple encoder is used to produce the gating signals for three BIST structures, as Fig. 4(b) shows. T0 and T1, two decode signals, have different sources during different stages, we will discuss them below. This decoder circuit just requires two inverses and an exclusive-NOR gate. The three gating signals to its BIST are g1, g2 and g3. As the clock CLK is transmitted to all of the three BIST structures, when T0 = 0 and T1 = 0, it causes g1 = g2 = g3 =1, which can activate all distributed BIST structures. However, when T0 = 0 and T1 = 1, it cause g1=1 and g2 = g3 = 0, in this case, it actives the first distributed BIST, BIST1, and simultaneously gates off the other BIST structures. The other combinations of T0 and T1 can similarly control corresponding distributed BIST structures.

During Re-test and Post repair stages, when one or more specified BIST structures, including the corresponding test session, are needed to retest, it only gates on the corresponding session and off others, which can reduce retest time significantly. Also, it is because this method gates off some non- required test sessions that it can reduce the power consumption.

This encoder circuit can be easily comprehended by existing ATPG tools and the generated patterns can be used directly without any modification. Also, we don't consider clock tree during our discussion. In this paper ,we assume that the number of the distributed BIST used here is three. However, in the large number of memory blocks, it will require more test pins to process them. The BIRA and BISR will be discussed in the next section[13].

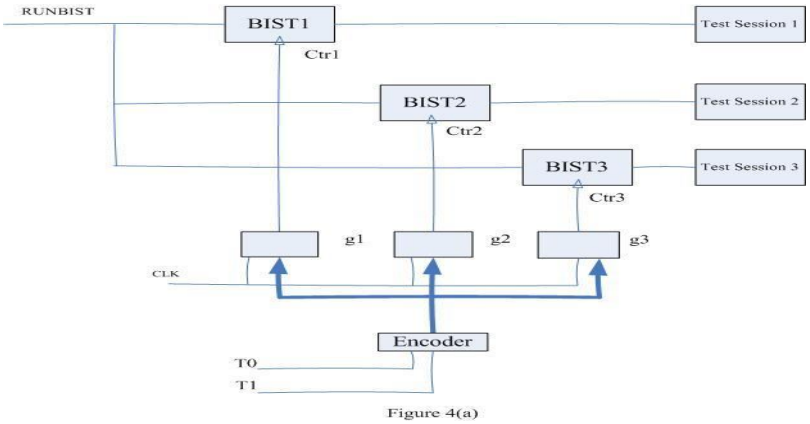


Figure 4(a)

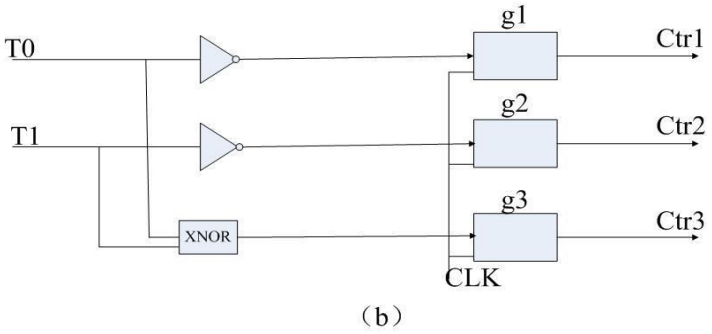


Fig.4. (a) A gated distributed BIST structure with encoder (b) Encoder and gating circuit

In order to enhance the memory yield and subsequently the overall chip yield, it is necessary to perform Built-In Self-Repair(BISR). However, BISR procedures which need a mount of redundancies are interacted with redundancy analysis, it is necessary to discuss the redundancy analysis, which determines how to use the spare rows and columns to repair faults found in the BIST. In our method, we use a shared BIRA instead of a individual BIRA for each memory.

In this stage, we mainly focus on analyzing and repairing faults in single memory , but store all the faulty information including unused in this stage, however, delete it once faulty units is repaired, which can be used later.

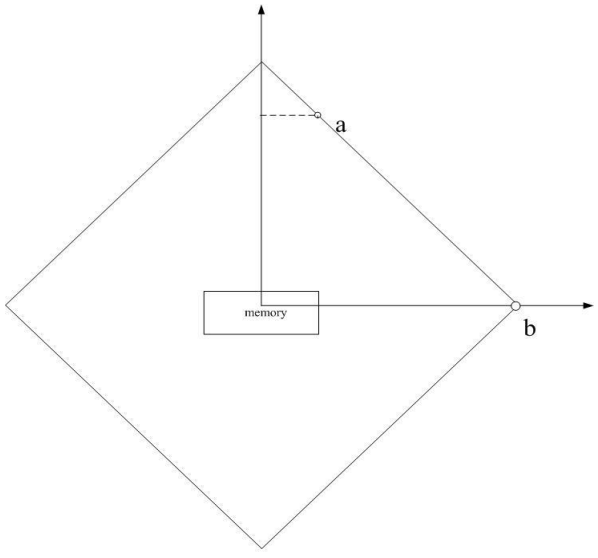
We consider a memory array with r spare rows and c spare columns [9], both of which can decide the repair rate. In this paper, we don't consider the types of faults[14]. There are a lot of redundancy analysis in literature now. It can first use the method of redundancy analysis according to mathematical analysis and some lemma in [9] to decide whether the memory can be repairable or not. However, since we need to store all have-not-repairable information, we must deal with another considerable issue, the volume of stored faulty information. We can use state-of-the-art methods to compress the information, but it is necessary to add test session's number (such as TS1, TS2 and TS3) and corespondent memory block's number in the test session to the item of faulty information, so that it can be distinguished by later procedures. Also, in this stage, if we find some memories are unrepairable , we don' t deal with them and just leave them in the faulty table. Once redundancy analysis has been finished, either a hard repair or a soft repair procedure is executed. (Generally, it is advisable to choose the hard repair which uses fuses, anti-fuses or laser programming to disconnect rows and columns with faulty bits and replace them with redundancy rows and columns in the same memory block in this stage[15]. However, in the post repair stage, it's better to use soft repair in the same test session.) Repaired memories go into the stage of retest. The terminated time of both test/repair stage and re-test stage are various[1].

5. RETEST AND POST REPAIR

In retest stage, as Fig. 2 shows, the primary goal is to guarantee the repaired memory in good condition after the hard repair. With the continuing reduced size of CMOS and the high density of IC, it may cause some other memory units into faulty memories during the hard repair stage. So it is necessary to check again whether the repaired memories are in good condition. Also, we just check the repaired memories in same test session and don't care about the unrepairable memories

from the repair stage. Thus it can reduce the test time to check repaired memory blocks. When some repaired memories are faulty, we store the faulty information similarly to that of the BIST and repair stage. If no more faults are found, we can declare that the memory have been repaired. However, in this stage, it uses shared BIRA 's information to the encoder the distributed BIST to decide whether each BIST structure gates on or off. If faults are found in retest, it's necessary to go to the stage of post repair.

In state-of-the-art methods, there are no such steps, which is for the first time proposed by this paper. Since our overall test flow based on the memory grouping strategy presented in section II, after the post repair (including test and repair procedures), it can be significant to improve the yield of chips. Before we discuss post repair stage, it is necessary to give the conception of feasible region of the memory block and test session in order to get more effective understanding of this stage[6]. Generally, we assume that the selecting algorithm, which selects one or more memory blocks to repair itself or others in the same test session, uses either horizontal or vertical segments in the two-dimension (2D) layer , and the central point of square is origin. Fig. 5(a) shows a feasible region of a memory block which belongs to one test session. From the figure it can be seen the distance between any point of sides and the center point, memory block, are equal. In this paper, we don't take the three-dimension (3D) into consideration[16], since it needs to store more data information and more complexity algorithm. For example, as Fig. 5(a) shows, according to our assumption, the distance between a point and central point is equal to the distance from b point to central point. Also, we call this feasible region of a memory as memory region. Now, we discuss the memory region of test session, named session region. As we can see in Fig. 5(b), there are some overlap areas in session region and we can use any memory which owns the overlap area to repair. During post repair, the selecting algorithm is to determine which memory can be used for repairing another memory.



(a)

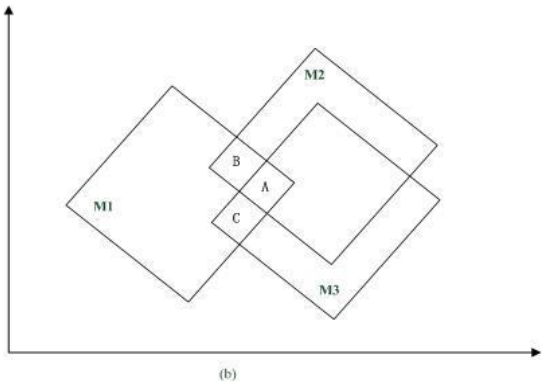


Fig.5. Feasible region : (a) a memory block , (b) a test session

To simplify the analysis, we choose redundant units just according to the distance between the faulty memory and other memories which have additional spare rows or columns for repairing in test session. That is, we choose the minimal distance for use. Many such selecting algorithms are available in literature[6].

In following session, we mainly focus on memories within one test session instead of the whole memory system. First, according to faulty information stored in shared BIRA memory, BIRA analyzes which test session can be repairable, by comparing the number of faulty units of the test session with the number of available redundancy rows and columns units in the same test session. If the former is equal or less than the later, it can be concluded that this test session can be repairable. Otherwise, this test session is partial repairable, and it also needs further to determine which memory blocks are unrepairable according to selecting algorithm. For those repairable blocks, signing a signal to it and repair it. Soft repair should be useful to repair them because during post repair most repairable units belong to different memories in the same test session (Soft repair uses a address-mapping procedure to bypass the faulty address location, so it doesn't destroy the lines between CMOS [2]). Since it uses the soft repair and there are few chances to cause other memory units into faulty memory , it need not to retest after post repair stage.

Since this stage treat the test session which involves many memory blocks as a whole, it can improve the repairable rate for the test session and hence improve the overall system's repairable rate.

6. SIMULATION AND ANALYSIS

The proposed test flow can be widely used in the future, although it is for the first time proposed. And no pubic benchmark exists until now, therefore, we use an example of a SoC including 8 memory blocks to show and demonstrate the procedure of the proposed flow. In TABLE1, memories are named M1, M2, ..., M8 and the characteristics of memory blocks that are used for our simulation are shown. TABLE1 illustrates the basic information of 8 memories in which (Row × Col) denotes the combination of the the row number and the column number in each memory; (r , c) denotes the number of spare rows and spare columns respectively; power column represents the power consumption, in mW, during BIST for each memory; and the followed

columns, BIST1, BIST2, BIST3, separately show the distance between memories and each distributed BIST structure, in units (one unit represent a certain distance). Next, a mathematical method will be presented to analyze the proposed test flow.

According to the proposed test flow, the test session should be decided first by the grouping algorithm. Here the power constraint for each distributed BIST is 350 mW, respectively. By performing the grouping algorithm, eight memories illustrated in session II are divided into three test sessions: TS1 = { M1, M2, M3}, TS2 = { M4, M8 } and TS3 = { M5, M6, M7 }. And memories in the same test session can be tested simultaneously, as presented in session II. We assume faulty numbers for each memory, as TABLE2 shows. The second row shows the available redundancy number of each memory achieved from TABLE1. The third row is the number of faulty units we assumed. By the illustration of usage, this proposal test flow is easily understood.

From statistics that we give, without the stage of post repair, three memory, M1, M2 and M8, will be assigned to the signal, unrepairable, since the number of redundancies is less than the number of faulty units. According to the formation of repairable rate:

$$repairable\ rate = \frac{the\ number\ of\ repaired}{the\ total\ of\ tested}$$

The repairable rate of eight memories, 62.5%, can be got. However, using the proposed method, which includes the retest and post repair, it can get higher repairable rate.

During the post repair stage, we deal with faulty units within test session, which can use one memory's redundancies for another units by soft repair. Therefore, as long as there exists free redundancy units in test session, it can repair the faulty units in the same test session. After the stage of post repair ,the repairable rate of eight memories reaches 100%. The repairable rate can be significantly improved in this case.

Based on simulation above, it is obvious that the proposed flow is promising in improving repairable rate. However, there are a few slight drawbacks in the chip area and test time. Compared to the improved repairable rate, those drawbacks can be compensated, especially in the trend of continuous reducing size and improved frequency of chip.

TABLE 1: BASIC INFORMATION OF 8 MEMORIES IN THIS EXAMPLE

# Mem	Size (Row×Col)	Redundancy (r ,c)	Power (mW)	BIST1	BIST2	BSIT3
M1	2048×32	(1, 1)	90	3	4	4
M2	4096×16	(2, 1)	100	1	6	5
M3	8192×16	(2, 1)	160	2	3	5
M4	8192×32	(2, 2)	300	4	2	6
M5	4096×32	(2, 2)	150	5	6	3

M6	8192×8	(2, 1)	80	4	5	1
M7	2048×64	(1, 2)	120	6	6	2
M8	4096×8	(2, 1)	40	5	2	3

TABLE 2: REDUNDANCY INFORMATION AND FAULTS UNITE ASSUMED IN THIS EXAMPLE

#Mem	M1	M2	M3	M4	M5	M6	M7	M8
Redundancy	2080	5028	8224	16448	10056	8208	4160	5012
Faults units	2540	5800	5210	11200	9800	6300	3500	6500

7. CONCLUSION

This paper presented a revised test/repair flow and an effective distributed BIST structure based on memory grouping algorithm for large number of cores in SoC. In the flow, a robust post repair stage is proposed for the first time. Simulation results show that it's useful to improve the yield of memory cores. Our future work is to investigate more effective way to store faulty information and the optimization techniques to reduce circuity areas and test time as well as power consumption.

ACKNOWLEDGMENTS

The authors would like to thank members of Research Center of Microprocessor Technology, Institute of Computing Technology , Chinese Academy of Sciences.

REFERENCES

[1] Chih-Sheng Hou, Jin-Fu Li, Che-Wei Chou, "Test and Repair Scheduling for Built-In Self-Repair RAMs in SOCs," Electronic Design, Test and Application, 2010. DELTA '10. Fifth IEEE International Symposium on, 2010, pp.3 - 7.

[2] R. Rajsuman, "Design and test of large embedded memories: An overview," Design & Test of Computers, IEEE, 2001, pp.16 - 27.

[3] D. Czysz, G. Mrugalski, J. Rajski, J. Tyszer, "Low Power Embedded Deterministic Test", VLSI Test Symposium, 2007. 25th IEEE, pp. 75 - 83 .

[4] C. -D. Huang, T. -W. Tseng, J. -F. Li, "An Infrastructure IP for Repairing Multiple RAMs in SOCs," VLSI Design, Automation and Test, 2006 International Symposium on, pp. 1- 4.

[5] T . Yoneda, Y. Fukuda, H. Fujiwara, "Test Scheduling for Memory Cores with Built-In Self-Repair," Asian Test Symposium, 2007. ATS '07. 16th, pp. 199 - 206.

[6] Tsu-Wei Tseng, Chih-Sheng Hou, Jin-Fu Li, "Automatic generation of memory built-in self-repair circuits in SOCs for minimizing test time and area cost ," VLSI Test Symposium (VTS), 2010 28th, pp. 21 - 26.

[7] Jaeyong Chung, Joonsung Park, J. A. Abraham, Eonjo Byun, Cheol-Jong Woo, "Reducing test time and area overhead of an embedded memory array built-in repair analyzer with optimal repair rate," VLSI Test Symposium (VTS), 2010 28th, pp. 33 - 38.

[8] S. Irobi, Z. Al-Ars, S. Hamdioui, "Detecting Memory Faults in the Presence of Bit Line Coupling in SRAM Devices," Test Conference (ITC), 2010 IEEE International, 2010.

[9] Y. Zorian, "A distributed BIST control scheme for complex VLSI devices," VLSI Test Symposium, 1993. Digest of Papers., Eleventh Annual 1993 IEEE, pp. 4 - 9

[10] Tsu-Wei Tseng, Jin-Fu Li, Chih-Chiang Hsu, Alex Pao, Kevin Chiu, Eliot Chen, "A Reconfigurable Built-In Self-Repair Scheme for Multiple Repairable RAMs in SOCs," Test Conference, 2006. ITC '06. IEEE International , pp. 1 - 9.

[11] H. -c. Lihn, "Reusable, Low-cost, and Flexible Multidrop System JTAG Architecture, " Test Conference, 2006. ITC '06. IEEE International.

[12] P. Ohler, S. Hellebrand, H.-J. Wunderlich, "An Integrated Built-In Test and Repair Approach for Memories with 2D Redundancy ," Test Symposium, 2007. ETS '07. 12th IEEE European, pp. 91 - 96.

[13] J.Saxena, K. M. Butlter, L. Whetsel, "An analysis of power reduction techniques in scan testing," Proc.International Test Conference, Baltimore, MD, USA, 2001, pp. 670 - 677.

[14] P. Bernardi, M. Grosso, M.S. Reorda, Y. Zhang, "A programmable BIST for DRAM testing and diagnosis ," Test Conference (ITC), 2010 IEEE International, 2010.

[15] T. Kawagoe, J. Ohtani,M. Niiro,T. Ooishi, M. Hamada, H. Hidaka, "A built-in self-repair analyzer (CRESTA) for embedded DRAMs," Test Conference, 2000. Proceedings. International, 2000, pp. 567 - 574.

[16] E. J. Marinissen, "Test challenges for 3D-SICs: All the old, most of the recent, and then some new! " Test Conference, 2009. ITC 2009. International.