An Optimal Memory BISR Implementation

Maddu Karunaratne
University of Pittsburgh at Johnstown, PA, USA
Email: maddu@pitt.edu

Bejoy Oomann
Genesys Testware Inc., Fremont, CA, USA
Email: bejoy.oomann@genesystest.com

Abstract—**Although integrated circuits (IC) shrink in size as the fabrication technology progresses, circuit designers always attempt to incorporate as much functionality as possible into a single die. Processor chips, particularly those used in data communications, are known to have the highest transistor density because they contain the highest percentage of embedded memories. In some cases, embedded memories occupy 50-80% of the die area. With inherently high density of memories, the manufacturing yield can become very poor even in a mature process. Also to keep the test cost affordable, having Built-In Self Tests (BIST) for memories is essential although testing would not improve the yield by itself. It has become a common practice to implement some type of memory repair scheme along with BIST in memory dominant IC designs. In this writing, such an integrated scheme is referred as Memory Built-In Self Repair (MBISR.)**

This paper elaborates a few practical criteria on designing and implementing built-in self-test circuits for testing and repairing a large number of embedded memories of different types and sizes in a single Integrated Circuit (IC). Various test architectures presented in this paper provide for different optimizing criteria such as test time, routing feasibility, silicon overhead, and dynamic power compliance. The repair circuits are based on the most popular and widely accepted built-in-self-test strategy, and are power aware, repair friendly, and supports scan based testing of random glue logic in SOC designs. These features are useful primarily in SOC testing because such designs typically contain many memories that are large but repairable.

Without an effective repair scheme, the production yield of a SOC containing a large numbers of embedded memory types and instances may severely be compromised. We selected one of the optimizing criteria as the main objective, and made the relevant repair scheme implemented on a processor chip using a mature 0.18 micron process due to its low cost of fabrication. The repair scheme allowed self testing and repair at both wafer and package levels. We present the silicon data showing the actual Return On Yield (ROY) due to the built-in repair scheme when the repair scheme was dynamically controlled at test time.

Index terms— **Memory, BIST, Dynamic, Repair, Yield, Redundancy**

I. INTRODUCTION

Rapid advances in the semiconductor manufacturing process provide designers ability to create ultra large and complex integrated circuits. To fully utilize the abundance of silicon real state on a chip, designers embed large high-density memory structures along with microprocessor cores and macro-cells in contemporary integrated circuit chips. While most of those embedded memories are Random Access Memories (RAM) in the form of static RAMs or dynamic RAMs, some of the other dense regular structure types being used include ROM, First-in First-Out (FIFO) memory, Last-In First-Out memory (LIFO), Content Addressable Memory (CAM), and Cache Memory.

The amount of memory area in a die could vary from 20% to 80% depending on their target applications, based on the information provided by many customers of Genesys TestWare Inc. over the years. In certain microprocessor chips, memory area may cover over 70% of the overall silicon area. Since the embedded memory structures are fabricated with high density, most of the failed integrated circuit (IC) parts would reveal that their defects were present in the memory areas of the defective chips. This mandates that memories be tested thoroughly in order to screen the ICs. In addition, embedded DRAMs and large SRAMs require some type of repair action be taken to increase the final yield.

Due to low cost and effectiveness, the most often used method for testing embedded memories is Built-In Self-Test (BIST) which is shown in Fig 1. The main advantages of MBIST over other memory test techniques are:

(a) Test pattern generation is automatically done and hence requires only a short test development time.
(b) Test equipment can be very inexpensive since the test structures are all inside the IC chip and therefore a single hardware tester can launch memory tests and perform logic tests, too.
(c) Tests can be applied automatically at the normal speed of the device thereby enabling testing of dynamic faults.
(d) External test data volume is extremely low compared to the millions of clock cycles in the actual BIST test itself that is necessary to test a memory.
(e) Due to the built-in nature, the tests can be implemented to work at board level or system level, beyond the chip level, unlike other memory test methods.
(f) Embedded memories are not accessible via device pins in most cases due to the high routing overhead of pin multiplexing. If the number of chip pins is less than the

© 2013 ACADEMY PUBLISHER
doi:10.4304/jcp.8.9.2167-2174

number of memory ports, pin multiplexing is not feasible at all.

Fig 1 shows a conceptual block diagram of a Built-In Self-Test scheme for a single memory embedded between the input and output logic blocks which are either synthesized or core based. Difficulties in controlling inputs of the memory directly through the input pads and observing the memory response at the IC chip output pads are circumvented by placing the test generator and the response analyzer blocks at the boundary of the memory core as shown. The controller block has to be present to manage the test and functional modes of the circuit and coordinate the data movements during tests. This scheme is very effective with large or small embedded memories in applying millions of test cycles for various patterns aimed to detect defects identified in regular dense two dimensional arrays like RAMs. One main reason for its effectiveness is that tests can be applied at the design functional speed of the device which in-turn helps detect signal coupling and transient defects.

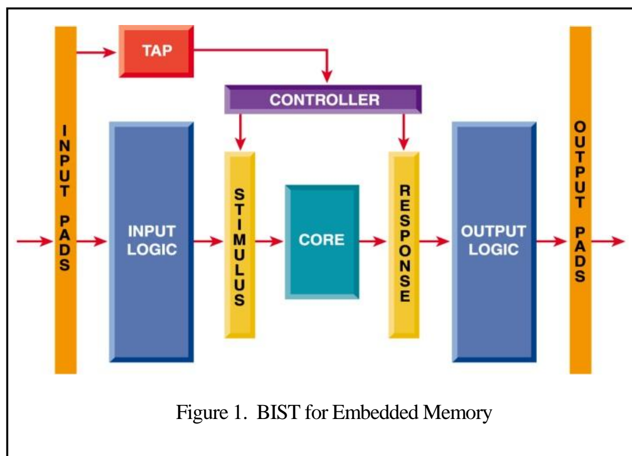


Figure 1. BIST for Embedded Memory

However, MBIST or any other pure testing would not address the loss of parts due to manufacturing defects but only the screening aspects of those parts. As more of the die area is allocated for memories, the yield loss becomes a concern. Due to the high density and the relatively large area occupied by memories, spot-defect causing foreign matter in silicon fabrication chambers has a much higher probability of causing a fault on memory areas than on logic area of the chip. Therefore, even in a mature process technology, random defects can cause yield loss that manufactures would like to see reduced.

In complex SOC designs, the die area is usually limited by the routing constraints and I/O pad counts. Therefore, it is highly cost effective to also add extra logic for redundancy analysis and repair schemes for embedded memory to recover defective parts. For stand-alone memories, simple memory testers can identify the location of defective data cells, and this locality information is then programmed in non-volatile fuses to repair the memory chip if they were designed for repair. Fuses are then programmed either electrically or optically [1-4].

The repair of the memories found to be defective by BIST runs is achieved in a *dynamic repair* scheme based on BIST faulty data gathered for each memory, and is modeled after what is typically known as Soft Repair described in research literature [2,5,7]. In dynamic repair, memory repair information is not stored in fuses; rather they are recalculated each time the chip is powered up. One of the drawbacks of dynamic repair is the possibility of system failures due to voltage and temperature sensitive memory defects that are not detected at power-up, but would manifest in to a failure during its normal operation after power-up due to heating effects. Note that the BIST schemes can test and repair memories when the device is in the test mode which can happen only at power-up. The prevalence of voltage and temperature sensitive memory defects is dependent on the margins of the memory design and IC manufacturing process. Several built-in schemes to enable testing of these defects are described in [5]. Built-in Diagnosis and Repair schemes always need additional logic gates (hence silicon area) beyond that required for mere detection of defects.

In this paper, we focus on high-level practical issues on designing and implementing built-in self-test circuits for testing and repairing a circuit of embedded memories of different sizes and types. We presented the general idea of such issues in the conference paper [10] without providing a concrete example of how a MBIST scheme may be implemented for a particular objective. In Section II of this paper we discuss the MBIST architecture and essential functionality required for a memory test and a repair scheme. In Section III, we elaborate five different criteria that designers must consider before a test scheme is decided for a SOC chip. Section IV gives a lengthy description of a real repair scheme details implemented for a commercial SOC design under a select criterion described in Section III. Section V presents the real silicon test and repair data for the processor chip with analysis for return on yield showing the effeteness of the MBIST scheme implemented. Finally, Section V presents the conclusion of this work.

II. ADAPTING MBISR

We have used the ArraytestMaker™ software product from Genesys Testware Inc. with its Dynamic Data Repair feature in many commercial designs in various configurations to test and repair different size and types of memories. Most of the MBIST-based soft repair schemes use 1-D redundancy (i.e. one dimensional redundancy in data lines), and they fall into the broad category of Soft Repair schemes [2, 7]. This approach has very low impact on area and timing, and it does not use any pre-specified redundant data columns, nor does it use any complex reconfiguration schemes. A pre-specified (use only that column to mask the defective data column) data column would require a higher cost of routing because the data column to be used for repair is tagged to one particular data wire and any faulty wire would have to be routed to that fixed data line (static repair).

The ArraytestMaker test automation tool has a wide range of features with different test algorithms, diagnosis and repair capabilities, and shadow logic testing capabilities. It supports various RAM testing algorithms. The most popular algorithm among IC designers is the comprehensive IFA13 [3]. This algorithm requires only a reasonably small 13N test time, where N is the number of individual address locations of the

© 2013 ACADEMY PUBLISHER

memory. As an example, for a 128Kx8 memory, N is 128K with K being 1024. For a simplified hardware implementation, the automation test tool extends it to an algorithm named IFA15, giving a slightly longer 15N test time while retaining the same basic IFA13 algorithm. It detects *Address Faults*, various *Stuck-at Faults*, *Transition Faults*, and various *Coupling Faults*. Additionally, the tool adds test structures in order to apply all the primary data background tests [3] and hence the tests can cover *Neighborhood Pattern Sensitive Faults*, too.

A block diagram of the test structures placed around a typical memory is shown in Fig 2, and the individual test structure modules are labeled BIC, APG, DPG, FBS, and ORE. BIC is the *BIST Interface Controller* module which coordinates the application of the memory tests, detection of the errors, and the repair scheme. BIC is activated by the IC chip test interface such as a standard JTAG test interface. *Address Pattern Generator* logic is in the APG module that creates the address values in a regular controlled pattern to sweep the entire address range both up and down under the direction of BIC module. DPG is the *Data Pattern Generator* module generating a predetermined algorithmic (IFA15 algorithm) bit pattern being channeled (multiplexed) to the data input bus of the memory.

The block marked FBS contains logic elements of the *Functional to BIST Selector* module which selects the test path or the normal functional path for the data flow through the memory. The ORE is the *Output Response Evaluator module* which compares the memory output data against the expected data pattern (at memory output) being generated by DPG to determine whether any data bits is faulty or not. If a data error is detected and a data redundancy bit exists, the faulty data line is bypassed at both memory input and memory output by the combined actions of BIC and ORE test modules and that repair configuration is kept in test registers in BIC and ORE modules. The circuit logic in DPG and APG blocks heavily depend on the BIST algorithm and the size of the memory.

BIST status and control registers in the memories without repair schemes are connected together to form a single shift register for the purpose of shifting out the test status information and to keep the data management on the hardware tester simple. Similarly, BIST status and control registers in all the memories with repair logic are connected into another separate shift register. Both shift registers can be controlled by IEEE 1149.1 Test Access Port (TAP) controllers using six private JTAG instructions. Typically a functional clock is used to run the BIST sequences, but the test clock TCK is used for shifting out the BIST diagnosis data. These shift register chains are made available for Automatic Test Pattern Generation (ATPG) tools to extract any additional fault coverage improvements.

III. Optimizing Criteria

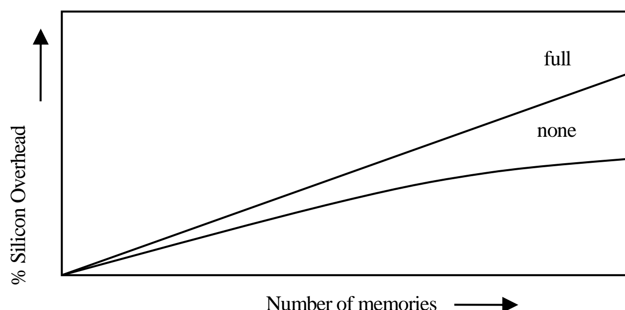
Fundamentally, different BIST test algorithms [3] introduce different cell area overhead, routing area overhead, and test application time in addition to covering different types of defects. The BIST concept shown in Fig 1 can be partitioned in to circuit design blocks shown in Fig 2 that was described earlier in Section II. The test blocks named DPG, APG, BIC and ORE can be placed anywhere in the design hierarchy of the chip allowing various sharing schemes with them. Each memory has a dedicated FBS block to switch between the test path and functional data paths and to repair the faulty lines dynamically under the control of BIC. We can consider several different high level architectures of testing a group of embedded memories to suit different optimizing criteria described as follows:

A. Criterion 1 - Optimum silicon area

Although the current designs typically offer ample die area to accommodate test structures, certain circuits may call for area optimized test structures given that any area used up by test structure is an area not available for the actual design functionality and test logic really does not add any end value to a chip. When such minimum test overhead area is desired for chips with a very large number of memories (such as 250 RAM instances in a network processor) we can reduce the area overhead by sharing certain BIST blocks among a group or memories which are in the same physical area of the die. Such a group is called a *cluster*.

While it is possible to share only one or more of the DPG, APG, BIC, and ORE blocks, the optimum area for a cluster can be achieved by sharing all of those blocks by a cluster. Grouping of memories into clusters depends on several factors such as physical proximity of memories, and their individual address and (particularly) data sizes. Additionally, considerations should be given to the maximum allowed power consumption (criterion 4), voltage and current levels of power grid in various circuit blocks, and to the total test application time (Criterion 2) acceptable for the project.

The Graph 1 below shows how the die area changes as the number of memories eligible for sharing increase. When BIST structures are not shared (*none* curve), the addition of more memories simply increases the area in a linear way if they are of the same size and type. When sharing is attempted to the maximum possible level (*full* curve), incremental addition of more memories tends to reduce the rate of die area increase at higher memory counts.



Graph 1: Die Area Change

Fig. 2 shows the hierarchical block diagram for a cluster of two memories with maximum sharing of BIST and repair structures. Module TOP-LEVEL BIST contains the BIC, ORE, DPG, and APG blocks common to all the memories. Each memory always has its own FBS module. Repair information is stored in FBS, ORE and BIC blocks. The test time, however, is the longest of any configurations and is

© 2013 ACADEMY PUBLISHER

determined by the cluster with the longest cumulative test time. BIC selects each memory sequentially for testing and repair. Clusters can be tested concurrently. Designers can encompass all the memories into a single cluster giving the smallest area overhead, but it causes the longest test time and the hardest routing challenge for test block connections.

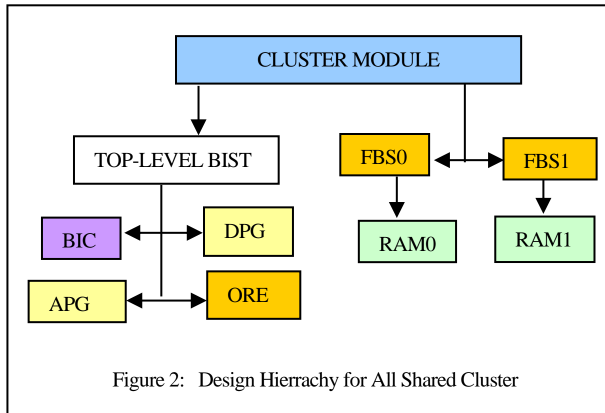


Figure 2: Design Hierarchy for All Shared Cluster

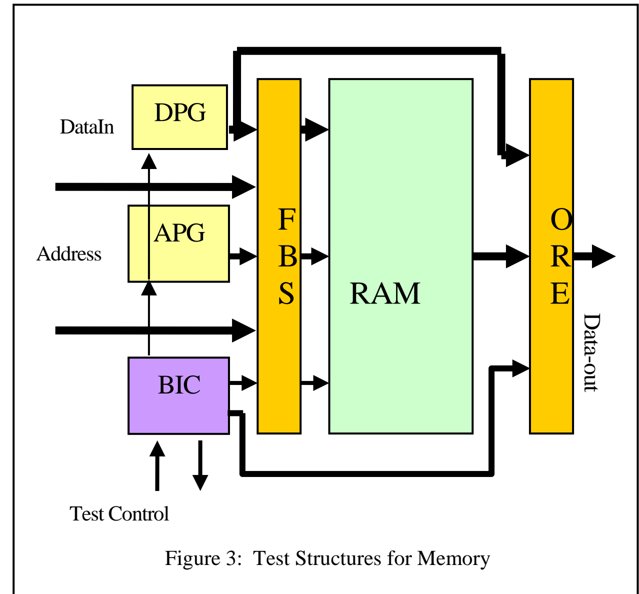


Figure 3: Test Structures for Memory

B. Criterion 2 – Optim um test time

The BIST runs and repairs are launched at chip power-up even on the final system board. Some systems require that self tests and system self-diagnosis be done as quickly as possible implying that BIST runs be done at the optimum time. We can achieve minimum test time for all the embedded memories if we were to run the tests concurrently, which suggests that each tested memory have its own dedicated BIST blocks as shown in Fig 3. At the top hierarchy level, the design simply connects the test signals of these memories in to form two scan chains – one for repairable memories, and the other for those without repair logic.

However, for large SOCs with a high count of BISTed memories, test engineer has to consider power requirements of concurrent data write-operations of RAMs that are using the power islands in the chip. Criterion 4 further discusses the MBIST power implications, which would be the major limiting factor against achieving the shortest test time even when silicon die area is available for unshared BISR structures.

We made a memory repair scheme implemented with emphasis on test time, on a processor chip using the mature 0.18 micron process due to its low cost of implementation. The particular repair scheme allows Self Testing and Self Repair at both wafer and package levels. The implementation details are discussed in section IV of the paper. The silicon data came from a customer of Genesys Testware Inc. who wishes to remain anonymous for various reasons. However, the data presented are real silicon wafer data extracted using hardware test equipment at a commercial testing facility. We will describe the real test data, and yield improvements achieved as a validation of the repair criteria described here.

C. Criterion 3 – Optimum Routing Feasibility

For clusters with a large number of memories or very wide memories, routing the connections from shared BIST blocks to each memory or FBS blocks may pose a challenge to CAD tools. We have the highest routing flexibility when each cluster contains only one memory. However, practical implementations may dictate clusters be formed with the capabilities of routing tool in mind. To keep routing feasible, some clusters would have fewer, but larger memories while others would have many small memories.

In case of large clusters, sharing all the BIST blocks may also cause routing issues. Therefore, to maintain routing feasibility, cluster size, cluster content, and the BIST sharing strategy have to be carefully planned and/or experimented with. One way to simultaneously optimize the routing feasibility, the test time and silicon area is through the partial sharing of the BIST circuitry among memories as shown in Fig 4. In this case BIC, DPG and APG modules are shared but not ORE. Routing is improved by avoiding multiplexing of memory outputs to a single (shared) ORE. Since all the memories are accessed simultaneously, test time would not increase.

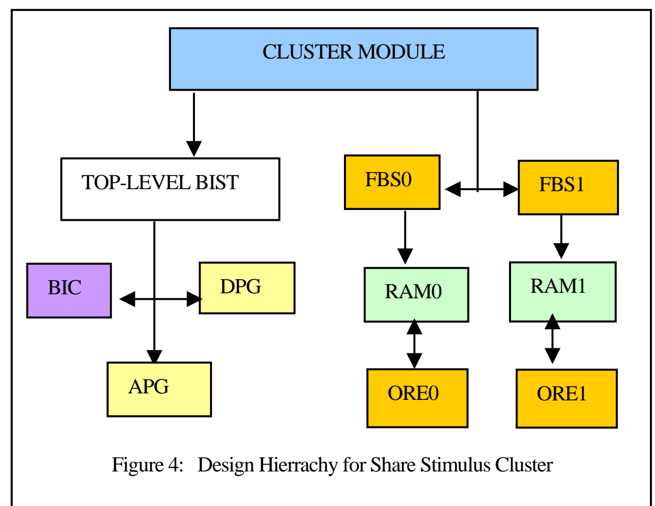
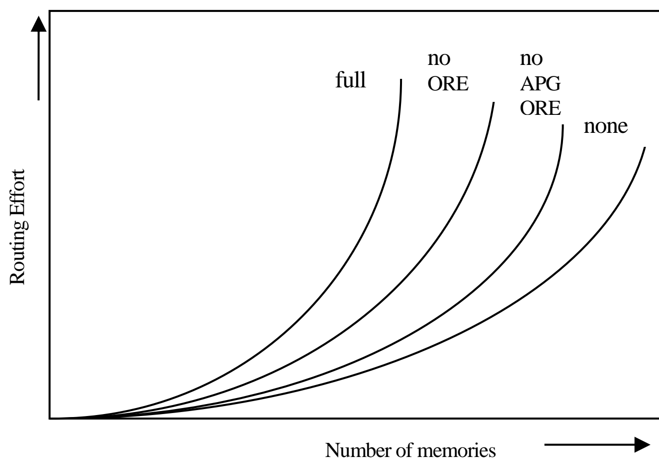


Figure 4: Design Hierarchy for Share Stimulus Cluster

© 2013 ACADEMY PUBLISHER

Graph 2 shows the qualitative behavior of an effort in routing the circuit with BIST and BISR circuit blocks when different sharing levels are considered. With full possible routing the efforts increases sharply and some cases end in failures. As more memories are added for the effort the required effort by routing tools again increases. The no share case is the easiest as expected, but it has some challenges if the memories are very wide, too. The cases with other combinations such as just OREs are not shared, and both APG and OREs are not shared fall in between as shown. It needs to be noted this behavior is highly case dependant and also memory proximity dependent as well.



Graph 2: Impact on Routing

D. Criterion 4 – Optimal Power Design

Modern SOC designs contain several different voltage levels, power islands and a complex power rail grid in order to reduce leakage and switching power in the functional mode. When a circuit operates, memories embedded throughout the chip typically operate at much slower frequencies than the activity (memory read and write operations) frequencies of BIST algorithms applied in the test mode. Simultaneous launch of at-speed BIST runs on clusters may cause excessive voltage drops (IR drop) that the power islands were not designed to accommodate, since power grid designs typically do not consider test requirements. Excessive IR drops may fail the BIST runs making test results unreliable, thus invalidating the memory test scheme. To allow design teams to achieve optimal power grids, DFT engineers need to manage the clustering and perhaps the clock frequency of the BIST runs.

A memory utilizes energy in its core structure array, its decoding logic for row and columns, and its sense drivers. Since the memory core energy use is much higher than the others we would only consider reducing core power usage here. As a strategy, we can consider the energy released by read and write operations of each memory, perhaps the worst case from each cluster and consider where the memory is physically placed, and use the power island specifications to validate that the BIST runs (at selected frequencies) would not violate the power rail allowances. Designers can obtain memory power usage information from memory or macro-cell vendors, or from the design library. Normally, separate shift registers formed by interconnecting BIST logic clusters are created for embedded test circuits. Each shift register can be independently operated. One way of optimizing the BIST architecture for low power designs is to allow several top level BIST and repair clusters corresponding to different power islands as shown in Fig 5. One cluster can be operated at a time using independent BIC modules to avoid excessive IR drop during memory testing.

E. Criterion 5 – Optimal Failure Map Generation

After the first batch of manufactured chips has been validated for functional correctness, the design is characterized for process corners, design parameters and yield. It is very useful to be able to generate failure maps for all embedded memories in a failing chip. A failure map tabulates all the failing bits in a chip. A failure map can be generated from the data logs generated by the Automatic Test Equipment (ATE) while applying MBIST patterns. However, some ATE has limitations on the number of consecutive vectors that can be logged. For example, some ATEs can log only up to 16384 consecutive vectors. If the length of the BIST shift register exceeds this limit, it may not be possible to obtain a memory failure map. One way of meeting this tester constraint is to allow multiple top levels BIST shift registers as shown in Fig 5.

Considering the above five criteria, the most practical implementation would be to consider all of them without attempting to optimize only one of them at the expense of others. It is clear that criteria 3 and 4 constrain the BIST and repair implementations in such a way that if they are violated, the designed circuit cannot operate or be tested effectively in silicon.

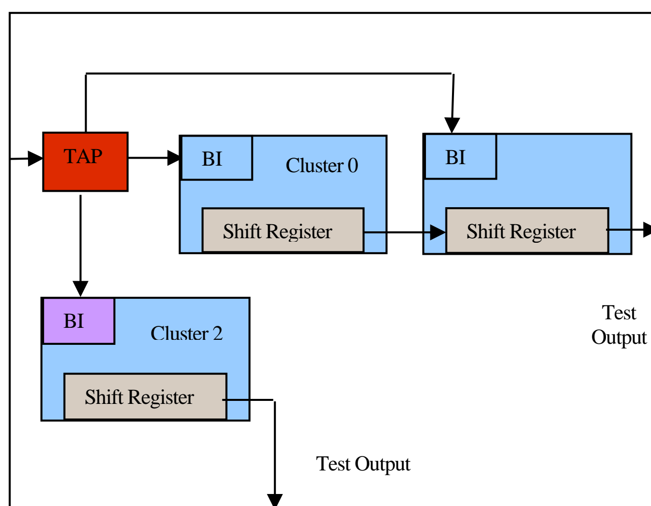


Figure 5: Top Level Block Diagram of Independent Clusters

IV. Implementation of Repair Criterion

Since the focus of this paper is memory test criteria, we present here the data relevant only to the embedded memories in the design we studied. The processor was designed with 47 distinct SRAMs and FIFOs with their total instance count at 90 in the complete netlist. Their size varied from few hundred bits to 250K bits. The deepest memory

© 2013 ACADEMY PUBLISHER

had 14 address lines, and the broadest was 46 bits wide. To maintain shortest test time under criterion 2, no attempt was made to share BIST structure logic among different memories which would have been to combine both criteria 1 and 2. A shared scheme would have resulted in less silicon overhead, but the test time would have been higher because the testing would have to be done sequentially among shared memories.

For small sized memories, the probability of containing a random defect within a small memory is much smaller than for larger memories. The silicon overhead will be proportionally much higher, too. Therefore, we did not add repair logic to small memories of less than 128 Kbits in size. All the larger ones were added extra data columns. If the data width is more than 16 bits, two data bit redundancy columns were recompiled (memories were compiled at the time this test strategy was planned). With this scheme we had 26 memory instances of 1 bit data redundancy, 26 instances of 2 bit, and 38 instances of no redundancy at all. However, all the memories contained BIST circuitry for detection of errors.

MBIST status and control registers in the memories without repair schemes were connected together to form a single shift register for the purpose of shifting out the test status information and to keep the data management on the hardware tester simple. The time required for moving such test data is negligible compared with the actual run time of the tests and hence no effort was made to have shorter multiple scan chains. Similarly, BIST status and control registers in all the memories with repair logic were connected into another separate shift register. Both shift registers were controlled by IEEE 1149.1 Test Access Port (TAP) controllers using six private JTAG instructions. We used the functional clocks to run the BIST sequences, but the test clock, TCK was used for shifting out the BIST diagnosis data. These shift register chains were made available for Automatic Test Pattern Generation (ATPG) tools to extract any additional testability or fault coverage improvements such tools would achieve.

Genesys Testware tools generated the top-level test bench to automatically drive the BIST and its repair schemes using the shift registers and JTAG instructions. Additionally it created a Waveform Generation Language (WGL) file for the tester for screening dies in the wafers (wafer sort) and subsequent testing of packaged parts. Both the test bench and the WGL file contain several procedural steps to self-test the BIST logic structures independent of the memory using the shift registers. This makes it easy to isolate faults in the BIST logic from that of the memory. Logic simulation and validation of the BIST scheme was done on a mixture of gate and RTL level design description using a standard commercial Verilog simulator.

The basic external control steps required for launching the tests were:

(a) Load the *run instruction* into the TAP controller.

(b) Force the TAP controller into Run Test Idle TAP state. This will launch the memory self test already setup in step (a).

(c) Supply an adequate number of system clocks (Test tools provided this number as an upper limit) to allow completion of the self-test for the entire circuit. BIST run will stop automatically at the end of the test sequence or upon the first non-repairable error (more defects found than the redundancy would repair). Tests are applied independently for each memory since no sharing of BIST structures was implemented. Even if one memory is found faulty, each of the other memories would still continue until successful completion or errors occur in each one.

(d) Load the *shift instruction* into TAP controller and force it to shift-DR TAP state.

(e) Shift out the resulting self test data and compare them with expected data. For immediate detection of any defective memory on the tester, the tool implemented the BIST so that the first bit value in each memory data segment shifted out is the pass/fail bit for that memory.

V. MBISR SILICON TEST DATA

After the initial sample parts were tested for memories and working parts were found, we altered the manufacturing process to produce wafers that would fall into the process margins, in order to validate the BIST repair scheme. In this section we will describe the test data, and yield enhancement statistics. It should be mentioned that a client of Genesys Testware Inc. provided the data described here, and the customer wishes to remain anonymous.

For testing the embedded memories , we applied the functional clock at 105 MHz, and the test clock at 10 MHz for shifting the registers. We were able to make the foundry adjust two silicon parameters: threshold voltage (Vt), and polysilicon critical dimension (CD). of the manufacturing process by 10% from their nominal values in both directions (positive and negative) to obtain nine different split lots. The other parameter planned, gate oxide thickness (OT) was unaltered due to foundry restrictions. The results of this experimental silicon data are shown in Table 1. Rows numbered 1 through 9 correspond to each split-lot, Rows 10 through 12 show statistical analysis of the data of the preceding rows.

TABLE 1.
MBIST REPAIR DATA ANALYSIS

	A	B	C	D
Split Lot No.	Parts under BIST	BIST found defects	BIST parts repaired	BIST repair yield %
1	115	42	26	61.90
2	113	29	15	51.72
3	102	28	17	60.71
4	106	16	11	68.75
5	98	37	30	81.08
6	97	30	20	66.66
7	112	45	25	55.55

© 2013 ACADEMY PUBLISHER

8	104	21	17	80.95
9	86	32	24	75.00
total	933	280	185	n/a
mean	104	31	21	67.00
Std Dev	9	9	6	11.00

Each of the 9 wafers contained 165 useful dies for a potential total of 1485 parts in all. However, wafer loss is inevitable due to cutting, handling, packaging, etc. We screened the wafers using JTAG tests, open-short contact tests, and some functional tests applied at-speed for logic area. This screened out I/O pad and gross logic failures to provide us with reasonable good dies to start memory tests. After the dies were cut and packaged, we screened out parts that failed the parametric tests and logic tests that may include parts due to damaged pins, opens, shorts, bond defects, etc. For the passing parts given in Column A of Table 1, we applied MBIST using JTAG controls with no repair allowed. After the tests were completed we shifted out the BIST status registers onto the tester channels and captured the signatures for each part tested. These tests were done at both the minimum and maximum supply voltages of both core and device pin pads.

We processed the signatures for any errors in any of the memories and binned the parts that showed errors under MBIST. For each part, we compared the error status signature recorded under the minimum voltage test against that recorded under maximum voltage. Our intent was to remove the parts that would show errors with voltage sensitivity. We saw no such error signature variations and we attribute it to the mature process we were using for manufacturing. Any such variations would indicate that the defects were sensitive to supply voltages. Column B contains the count of parts that had at least one defect in a memory (all the memories were BISTed). If we had not implemented the repair scheme all these parts would have been discarded as failures. By masking out certain fields in the shifted out BIST signatures of each embedded memory, we determined whether the memory has any uncorrectable faults. For example, if the memory had two data faults, but only one redundant bit, then the memory is uncorrectable.

Column C shows how many parts out of those in column B were actually repaired by our dynamic repair technique which we gathered by analyzing the shifted out BIST signatures. Column D contains the count of parts that were *repaired* using *only one redundant bit* (there were two redundant bits in some memories) out of the defective parts in column B. Column D is limited to the parts repaired using only one bit per memory even when the part would have been repaired using both bits if two redundant bits were available. We used the shifted out BIST signatures of the memories to decide whether a memory was repaired using one bit or both bits if available. We did not calculate the overall yield gain based on 165 dies per wafer since the emphasis here is how our repair scheme worked with the optimization criterion.

Totally, a 185 packaged parts were repaired out of 280 memory failed parts using only the single bit data redundancy and repair across the 9 way split lot wafers. It is on the average of 20% of MBIST tested parts, or 1 in 5 were repaired. However, a better meaning is present if we consider (column D) that 67% of the MBIST failed parts were completely repaired with a standard deviation of 11%. This is a significant high percentage of repair effectiveness. We should also consider the fact that small memories were not equipped with repair logic in the device.

We also noticed on several split lots that the increase in overall yield due to 2-bit redundancy was less than 1% which was not surprising since the possibility of two or more spot defects occurring on the same memory, particularly with 2-bit repair redundancy, is extremely low. However, we were not able to collect this data on all 9 split lots. When 67% MBIST repair rate is converted to the overall die counts of wafers, the increase is about 12% in the overall yield. Therefore, the area overhead (5%) is much less than the increase in overall yield (12%) from the single bit repair scheme. Considering the loss in cutting, packaging, bonding, handling, etc, this 5%:12% ratio would translate to be over 1:3 factor in favor since 12% of the cost of the finished products is lot higher than 5% of the cost of silicon on wafer. We also observed that failures were static and single bit redundancy was able to repair a significant portion of the defective parts from the wafers made by intentionally varying the manufacturing process in nine different ways**.**

V. Conclusion

We described the MBIST along with dynamic soft-repair and how they apply to system on chip type designs with many embedded memories. We further presented five different optimizing criteria (test time, area overhead, routing feasibility, power grid compliance, and failure mapping) that design and test engineers have to consider at higher decision making level before implementing BIST and repair schemes on complex designs.

We selected one criterion, the test time, and implemented single bit and dual bit data redundancy on 52 instances of embedded memories inside a single die of a commercial processor design for dynamic soft repair using BIST techniques. The remaining 38 small memories were added BIST structures for tests only, and the design was manufactured using a clean mature process. The design had about 9 megabits of total memory bits and we added about 5% silicon area (estimated by synthesis tools) overhead for single bit redundancy only, to repair two thirds of parts that had memory defects only without any penalty in the functionality or speed of the device

Since the single bit repair scheme recovered two thirds of all memory failures, the maximum repair yield gain possible with schemes of 2 or more data bits would be limited to one half of the yield gain provided by the single bit repair technique. It should be noted that more memories (38) had no repair circuitry than memories with 2-bit repair capability (26), and the possibility of having two defects on a 2-bit repair memory is remote as well. Therefore, we would not have expected to see any noteworthy yield increase by full 2-bit repair scheme on all the split lots beyond 1% increase

© 2013 ACADEMY PUBLISHER

seen on some of them, as mentioned earlier. Also, it would have been achieved at a much higher area overhead than the single bit repair scheme. Hence, we infer that the single bit data redundancy provided the optimal Return On Yield (ROY) on silicon investment over 1:3 or 300% as pointed out. This ROY was materialized on another optimization strategy (criterion 1) we had presented, although our main objective was to optimize the memory test time (Criterion 2.).

As discussed, if the optimizing the silicon area was the main objective, then a major scheme of sharing MBIST structures would have been implemented with a significant increase in test time, and routing difficulties would have been overcome by relaxing the sharing effectiveness. In conclusion, we strong feel that for large SOC designs, the embedded memory testing and repair solution need to be a compromise of these five orthogonal optimizing configurations.

REFERENCES

[1] R. McConnell, U. Moller, and D. Richter, "How We Test Siemmens Embedded DRAM Cores," in *Proceedings of International Test Conference*, 1998, pp. 1120-1126.

[2] F. Higgins, et. el., "Built-In Self Repair for Embedded High Density SRAM," in *Proceedings of International Test Conference*, 1998, pp. 1112-1119.

[3] A. J. van de Goor, "Testing Semiconductor Memories: theory and practice," John Wiley & Sons, 1991.

[4] M. Nicolaidis, N. Achouri, and S. Boutobza, "Optimal reconfiguration functions for column or data-bit built-in self-repair," in *Proceedings of Design, Automation and Test in Europe Conference and Exhibition*, 2003, pp. 590 – 595.

[5] Shyue-Kung Lu, and Shih-Chang Huang, "Built-in self-test and repair (BISTR) techniques for embedded RAMs," in *Records of International Workshop on Memory Technology, Design and Testing,* Aug. 2004, pp. 60 - 64 .

[6] R. Zappa, et. el., "Micro programmable built-in self repair for SRAMs," in *Records of International Workshop on Memory Technology, Design and Testing*, Aug. 2004, pp. 72 – 77.

[7] M. Nicolaidis, N. Achouri, and S. Boutobza, "Dynamic Data-bit Memory Built-In Self- Repair," in *Proceedings of ICCAD*, 2003, pp. 588-594.

[8] S. Shoukourian, V. Vardanian, and Y. Zorian, "A methodology for design and evaluation of redundancy allocation algorithms," in *Proceedings of IEEE VLSI Test Symposium*, April 2004, pp. 249 – 255.

[9] Aifredo Benso, et. al., "A Family of Self-Repair SRAM Cores,", in *Proceedings of IEEE International On -Line Testing Workshop*, July 2000, pp. 214-218.

[10] M. Karunaratne, and Bejoy Oomann, "Optimizing BIST and Repair logic for Embedded Memories," in *Proceedings of 51st IEEE International Midwest Symposium on Circuits and Systems*, August 2008.

[11] W. Jeong et al., "A Fast Built-in Redundancy Analysis for Memories with Optimal Repair Rate Using a Line-Based Search Tree," *IEEE Trans. Very Large Scale Integration*, vol. 17, no. 12, pp. 1665-1678, December 2009.

[12] T. Han, et. al., "High Repair Efficiency BIRA Algorithm with a Line Fault Scheme," *ETRI Journal*, vol.32, no.4, pp.642-644, August 2010.

[13] T.W. Tseng and J.F. Li, "ReBISR: A Reconfigurable Built-In Self-Repair Scheme for Random Access Memories in SOCs," *IEEE Trans. Very Large Scale Integration*, vol. 18, no. 6, pp. 921-932, June 2010.

[14] Yuejian Wu and Andre Ivanov, "Low Power SoC Memory BIST," in Proceedings of 21st IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, DFT 2006.

© 2013 ACADEMY PUBLISHER