A BIST-based Solution for the Diagnosis of Embedded Memories Adopting Image Processing Techniques

D. Appello, A. Fudoli, V. Tancorre
STMicroelectronics
Cornaredo, Italy

F. Corno, M. Rebaudengo, M. Sonza Reorda
Politecnico di Torino
Dipartimento di Automatica e Informatica
Torino, Italy

Abstract

This paper proposes a new solution for the diagnosis of faults into embedded RAMs, currently under evaluation within STMicroelectronics. The proposed scheme uses dedicated circuitry added to the BIST, selecting the failure data, and the ATE test program to schedule the data extraction flow. Testing is possible through a standard IEEE 1149.1 TAP, and allows the access to multiple cores with a P1500 compliant solution. The approach aims at implementing a low-cost solution to diagnose embedded RAMs with the goal to minimize the ATE costs and the time required to extract the diagnostic information. In our approach, the ATE drives the diagnostic scheme and is dedicated to the classification of faults, only, allowing adopting low-cost equipment. The proposed solution allows a scalable extraction of test data, whose amount is proportional to the available testing time. In order to accelerate the fault classification image processing techniques have been applied. The Hough transform has been adopted to analyze the bitmap representing the faulty cells. Preliminary experimental results show the advantages of the proposed approach in terms of time required to complete a diagnostic process.

1. Introduction

The market demand for highly integrated solutions in a single chip is constantly growing. Today very deep submicron technologies (VDSM) are allowing the integration on a single chip of entire systems (SOC). These devices typically embed unstructured logic blocks (micro-cores, DSP-cores, glue logic), a large spectrum of analog functions blocks and memories.

As far as memories are concerned, the elements over we wish to focus in this first analysis are the following:

- the percentage of area of the SOC occupied by memories is growing
- memories are usually the densest circuitry implemented, and therefore the most critical also for defectiveness.

These two facts induced individuating solutions to reduce the impact of defectiveness introducing some redundancy repair capabilities, in order to improve the yield [1-11].

Nevertheless, also to facilitate the effectiveness of repair, the overall defect level should be kept under control, and therefore requires the ability to deeply analyze the root cause of failures. This process typically requires a physical failure analysis (FA) on the parts. Despite the growth in the complexity of doing FA on VDSM technologies, FA of memories is simpler that analyzing unstructured logic because fault localization is theoretically possible. This means that, at least, the search space of the defect on the silicon can be restricted to a limited area, starting from deterministic information, provided by the test application sequence. By the way, there are other important factors limiting the effectiveness of failure analysis. FA is by nature a costly activity, and also requires significant amount of time to be completed. Therefore, to optimize the overall process, the importance of focusing on the most relevant source of failures first is essential. Use of statistical tools, applied over large databases of information usually helps. To follow this approach, the efficient combination of more factors is required: use data coming from a massively produced memory, collect relevant data, keep under control testing time and more in general, keep under control the cost of test.

Traditionally test of memories, both stand-alone and embedded, have been performed using dedicated hardware available on the ATE. These normally required some DFT, allowing the direct access from the chip boundary of the embedded memory relevant signals. Test application was normally achieved using dedicated HW/SW ATE options, usually able of generating in an algorithmic way the test stimuli and responses. One of the advantages of *external testing* was the ability of the ATE to collect all (or most) the failure information, and then, allowing reconstructing their topological location (bit-map). Despite fail bit-map is a powerful way for the advanced diagnosis of memories, their usage present also some drawback.

This is typically due to two reasons: size of bit-maps extracted from ATE is difficulty manageable in a manufacturing environment (for collection and analysis), testing time and test cost can become relevant issues. External test of memories is nowadays being replaced by BIST, for many different reasons. Consequently, also diagnosis must cope with this scenario. For many reasons, BIST is the envisaged solution for testing memories. In our previous work [2] it has been pointed out how BIST can be used for memory diagnosis, through some design for diagnosis effort. This paper focuses over two main aspects: the partitioning of diagnosis resources between the device and the ATE, and the problem of fault classification. We are facing it exploiting some technique inherited from image processing like the adaptive Hough transform. The goal is to extract from the test manufacturing flow relevant data, allowing analyzing the failure population and individuating the most relevant failures causes. Furthermore the adopted solution aims to satisfy the requirements of the manufacturing and of the engineering phase.

2. Architecture of the Proposed Approach

The STMicroelectronics design flow includes BIST solutions for testing embedded memories since several years; BIST for RAMs, for example adopted a March algorithm and included both a test and a debug mode. The latter was intended to allow the extraction of detailed diagnostic information from faulty chips, and was based on letting each BIST module evolve in single-step mode and produce complete information about the output behaviour of the RAM after each step of the implemented March test. The existing solutions had some limitations like the long time required extracting all the required information for diagnosis, the dependency on ATE performances features and the difficult extendibility to SOCs including multiple BIST cores.

The main idea behind our work is to extract from the BIST diagnostic structures a sufficient amount of information to achieve fault classification and identification. The constraints in which we searched and defined our solution are:

- a limited increase in the amount of area overhead
- a modular approach, suitable to the design automation tools
- a scalable solutions, allowing defining the quantity of diagnostic information to be extracted, budgeting the testing time
- using low-performances HW ATE, leveraging instead software resources, available through the system workstation.

The proposed architecture is graphically outlined in Fig. 1 and is based on an existing BIST module originally aimed at supporting fault detection, only. Some slight modifications have been introduced in this module to support diagnosis. The BIST module is connected to the external ATE through a TAP.

The proposed architecture can be divided in the following logic levels:

- *BIST module*: the unit in charge of executing the test algorithm on the embedded RAM
- *DBIST module*: the unit in charge of controlling the diagnostic execution of test algorithm on the embedded RAM; it is in charge of gathering the diagnostic information
- *Wrapper*: the interface towards the embedded memory is IEEE P1500 compliant; it wraps the DBIST module and supports the execution of the test and diagnosis algorithm; it is in charge of receiving the commands and sending the diagnostic information by means of the TAP controller through a suitable TAM (Test Access Mechanism) [12]
- *TAP Controller*: it decodes the commands sent by the ATE through the TAP and sends them to the BIST module through the Wrapper
- *Test Access Port* (TAP): the external interface towards the ATE is IEEE 1149.1 standard compliant, and allows to send the commands for executing the test and diagnosis algorithm and to read the results.

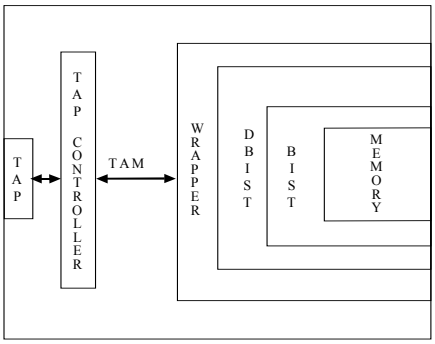


Figure 1: Test and diagnosis system architecture.

The Wrapper is in charge of interacting with the BIST module and providing some higher-level facilities, which can be activated from the outside through the TAP interface. These facilities correspond to TAP instructions and allow starting, pausing, and continuing the March algorithm, and extracting the information concerning the last detected error. Thanks to these facilities, the ATE can gather all the diagnosis information that is accessible when the memory is not embedded (i.e., the output value produced at each read operation executed by the March algorithm).

The IEEE 1149.1 compliant TAP Controller plays the role of interfacing the ATE with the wrapper, and hence with the BIST module. The instructions supported by the TAP controller are reported in Table 1.

Instructions	TAP Instructions description
RESET	Puts the BIST module into the reset state
RUNBIST	Puts the BIST module into the test mode and starts the March algorithm from the beginning
LOADSTEPS	Loads the number of operations to be executed by the March algorithm; by default this number is equal to the number of operations required by a single and complete element of the adopted March algorithm
READSTATUS	Reads the status register and verifies whether the March algorithm finished its task (either because it reached its end, or because it executed the specified number of operations)
READRESULT	Reads the register containing the information about the last error detected by the March algorithm
READERROR	Reads the register containing the number of errors detected by the March algorithm

Table 1: TAP instructions.

3. The Diagnosis Test Program

The tester independency has been another aspect that we considered to be essential in our development. From the electrical performances point of view, the embedded test and diagnosis solutions helped in a determinant way. The required resources are limited to few, low performances, pin electronics, with some data capturing ability. A high performance (clock only) channel has been used to test the circuit *at-speed*. The other crucial point was the test program. Since the individuated partitioning strategy of the diagnosis resources, between the ATE and the DUT, we had to individuate a tester independent way of developing the test program. This has been achieved using the meta-language instructions implemented in the TAP, see Table 1, allowing describing the diagnosis scheme at a sufficiently high abstraction level.

The scheme in Fig. 2 shows how the ATE interacts with the DBIST module through the TAP and wrapper. The diagnostic control module drives the DBIST via the TAP meta-instructions, in collecting failures information. The fault classification module, analyze the failure data, classifying the fault type.

In section 4, we will describe how techniques inherited from image processing are useful to classify faults, using a combination of fault dictionary and voting techniques built over the principle of an adaptive Hough transform. This solution allowed us to achieve fault classification, and localization, without the need of generating the complete failure bitmap. The DBIST module allows us to extract a quantity of information proportional to the testing time.

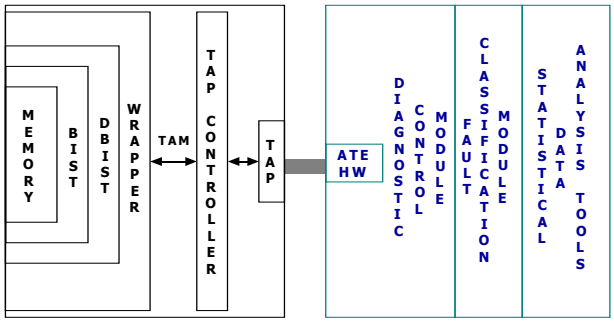


Figure 2: DUT and ATE diagnosis resources scheme.

4. Fault Classification

The adopted fault classification strategy uses image-processing techniques, to identify the most probable failure pattern among the set of possible ones provided through a fault dictionary. The adopted fault dictionary includes, for the considered memory under test, the most relevant failure patterns. Fig. 3 shows an example of the dictionary, with four different failure patterns.

In these conditions, while a subset of errors is captured from the BIST execution, a Hough transformation (HT) is applied [13-14], in the space indicated by the models of the fault dictionary.

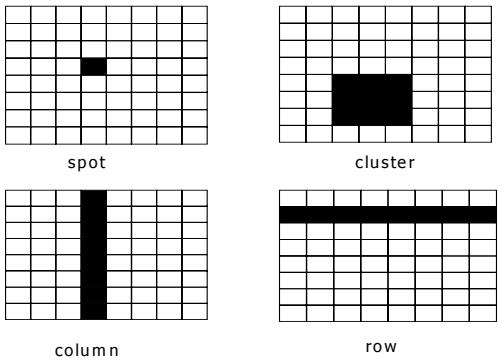


Figure 3: An example of a fault dictionary.

Hough transformation allows mapping the features recognized in an image space, to a set of points into a parameter space. In the classical image processing case the images are captured from a camera, and a filtering is required to extract the interesting features before the HT application. Furthermore, it is necessary to cope with the presence of noise, which reduces the ability to identify features. The straight-line representation is a classical example, indeed very useful also in our case. In the image space, a line is represented by a collection of (x,y) coordinates. In the HT, these points can be mapped into a polar representation (ρ,θ).

For each fault described into the dictionary, a specific HT should be computed, determining an array representing the specific Hough transformed space.

The analysis of the peaks into the arrays, also called *voting technique*, allows identifying the most probable fault candidate.

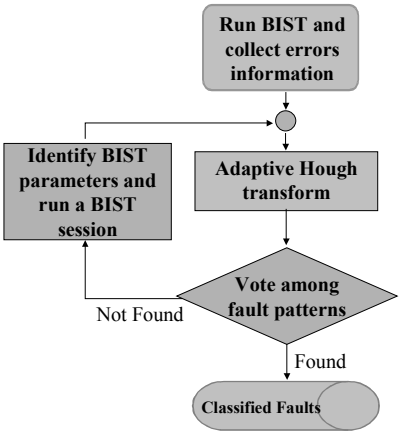


Figure 4. Flow chart of the fault classification process

The analysis starts considering a subset of the failures recognized by the BIST execution, mapped into our *virtual* image, that is a partial bitmap representation. HT and the voting technique are applied. This coarse classification of the fault is then refined, by an inverse mapping to the data point that may belong to the fault shape. A successive BIST execution, configuring opportunely the BIST controller, may allow confirming the first hypothesis. The approach also consists in matching one (or more) of the candidates described in the dictionary, with the available failure location [15].

From an image processing theory point of view, the technique can be classified under the problem of recognizing predefined patterns of occluded objects into binary images. The orthogonal nature of the pseudo-images and the intrinsic absence of capturing noise offered the ability to achieve high efficiency in the classification.

Furthermore, the effectiveness of the methodology is improved, accumulating the results over a large population, and analyzing the statistical computed data.

Figure 4 reports a block diagram showing the overall process of fault classification. Figure 5, offers a more detailed diagram of the Hough transformation process, applied to the fault dictionary we considered within this work. The HT in the "*spot*" space is actually the space itself. In this case the classification aims in recognizing presence of isolated faulty cells in the subset of localized faults. The following HT, for *rows* and *columns* space, determines two vectors of size [row,1] and [1, column], whose analysis leads to identifying the concerned faulty shapes. HT in the *cluster* space is obtained starting from the row and column space transform.

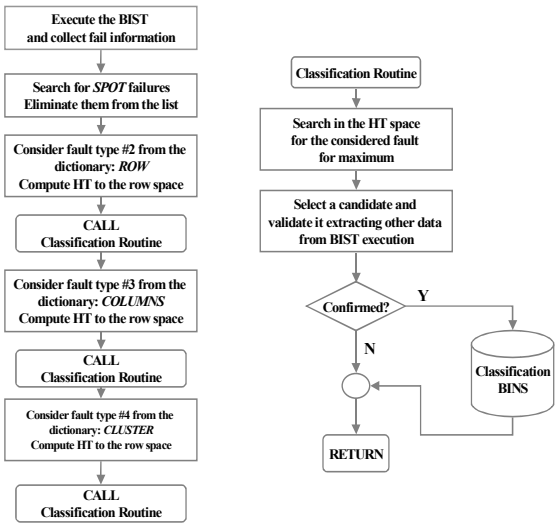
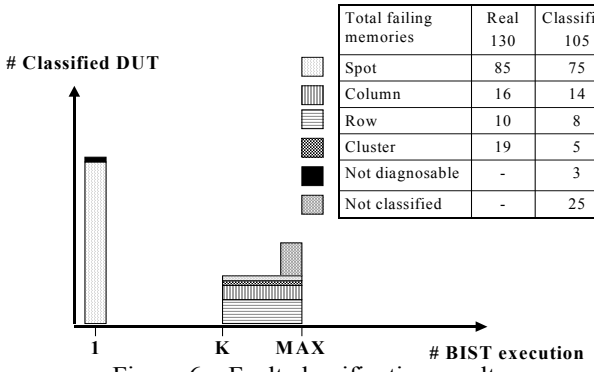


Figure 5. Detailed flow chart for the FC process

5. Experimental Results

The paper presented the results obtained in a project developed and still undergoing at STMicroelectronics. The architecture is currently being evaluated on a sample chip including two 16 K × 16 bits embedded memory. The chip is currently being manufactured using a mixed/power BCD 0.35 μm library. The area overhead introduced by the BIST and DBIST modules amounts to less than 2% of the memory area, that of the wrapper to about 1%, that for the TAP controller to less than 1%.

The bi-dimensional adaptive Hough transform, represented an effective solution able to classify the faults starting from a limited failure collection. This allowed us a significant testing time reduction, limiting the number of BIST runs. The effects of memory size on the efficiency of the solution are also limited. One goal of the project was to find the better trade-off between testing time and area overhead (mainly due to failure buffering).



Total failing memories	Real 130	Classified 105
Spot	85	75
Column	16	14
Row	10	8
Cluster	19	5
Not diagnosable	-	3
Not classified	-	25

Figure 6 – Fault classification results.

Figure 6 reports the results obtained with our experiments. Our analysis has been done considering 130 failing

parts, whose diagnosis has been performed first in an exhaustive way and then, using our algorithm. Figure 6 shows that one single run of the BIST is enough to classify 75 failing parts. This also depended by the fact that in our test case, spot failures showed up with isolated and few fails in the memory, and therefore were not aliased with other failure patterns. Furthermore, 25 memories have also been considered not usefully classifiable, since the elevated number of faulty cells detected by the first BIST run. Running the BIST for further K times, the algorithm can start the classification over a larger number of failures, topologically mapped. The value MAX - K represents the number of iterations required to confirm the fault pattern selected by the voting scheme. The values K and MAX should be chosen considering technological information, memory topology, test time constraint and size of the failure buffer available on board. The efficiency of the approach showed to be quite high for row and columns failures. Some aliasing effects between spot and cluster have also been observed and depend on the failure collection scheme implemented in our DBIST engine. Furthermore the overall efficiency of the algorithm, dropped down in presence of multiple failure types occurring simultaneously in the memory.

6. Conclusions

The presented approach is able to achieve fault classification and localization for embedded memories tested through BIST, within a test-manufacturing environment. In compliance with the initial hypothesis, we built a scheme allowing a scalable and modular extraction of fault diagnosis information in the available testing time. Full deterministic failure bit-map can be extracted, with some penalties for the testing time, but is a solution suitable for the engineering phase. Both in the manufacturing mode and in the engineering one, we only used low-cost resources on board of ATE. A diagnosis resource partitioning approach has been used, between the DUT and the ATE computer, where we implemented our diagnostic flow, using a combination of TAP instructions. Fault classification has been achieved using techniques, like the two-dimension adaptive Hough transform inherited from image processing.

Future works will address advances in design for diagnosis techniques. The improvement of DBIST will incorporate some of the fault classification structures, like some failure buffer, aimed to improve the overall efficiency and effectiveness of the approach. We also intend to focus in improving the classification efficiency of clusters.

7. References

[1] E. Nelson, J. Dreibeblbis, R. McConnell, Test and Repair of Large Embedded DRAMS: part 2, IEEE International Test Conference, 2001, pp. 173-181

[2] D. Appello, F. Corno, M. Giovinetto, M. Rebaudengo, M. Sonza Reorda, A P1500 compliant architecture for BIST-based Diagnosis of embedded RAMs, IEEE Asian Test Symposium, 2001, pp. 97-102

[3] C.-T. Huang, J.-R. Huang, C.-F. Wu, C.-W. Wu, T.-Y. Chang, A programmable BIST core for embedded DRAM, IEEE Design and Test of Computers, Vol. 16, No. 1, pp. 59-70, 1999

[4] R. David, A. Fuentes, Fault Diagnosis of RAM's from Random Testing Experiments, IEEE Transactions on Computer, 1990, Vol. 39, no. 2, pp. 220-229

[5] R. Treuer, V. Agarwal, Built-In Self Diagnosis for Repairable Embedded RAMs, IEEE Design and Test of Computers, 1993, Vol. 10, no. 2, pp. 24-33

[6] L. Shen, B. Cockburn, An Optimal March Test for Locating Faults in DRAMs, IEEE International Workshop on Memory Testing, 1993, pp. 61-66

[7] V.N. Yarmolik, Y.V. Klimets, A.J. van de Goor, S.N. Demidenko, RAM diagnostic tests, IEEE International Workshop on Memory Technology, Design and Testing, 1996, pp. 100-102

[8] T. Bergfeld, D. Niggemeyer, E. Rudnick, Diagnostic testing of embedded memories using BIST, IEEE Conference on Design, Automation and Test in Europe 2000, pp. 305-309

[9] C.-F. Wu, C.-T. Huang, C.W. Wang, K.-L. Cheng, C.-W. Wu, Error Catch and Analysis for Semiconductor Memories Using March Tests, IEEE/ACM International Conference on Computer Aided Design 2000, pp. 468-471

[10] J.-F. Lu, C.-W. Wu, Memory Fault Diagnosis by Syndrome Compression, IEEE Design, Automation and Test in Europe, 2001, pp. 97-101

[11] H.Kim, Y. Zorian, G. Komoriya, H. Pham, F. Higgins, J. Lewandowski, Built-In Self-Repair for Embedded High Density SRAM, IEEE International Test Conference, 1998, pp. 1112-1119

[12] E.J. Marinissen, Y. Zorian, R. Kapur, T. Taylor, L. Whetsel, Towards a Standard for Embedded Core Test: An Example, IEEE International Test Conference, 1999, pp. 616-627

[13] J. Illingworth, J. Kittler, The Adaptive Hough Transform, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-9, No. 5, September 1987, pp. 690-698

[14] C. Derek, W. Pao, Hon F. Li, Shapes Recognition Using Straight Line Hough Transform: Theory and Generalization, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14, No. 11, November 1992, pp. 1076-1089

[15] M. Gharavi-Alkhansari, A Fast Globally Optimal Algorithm for Template Matching Using Low-Resolution Pruning, IEEE Transactions on Image Processing, Vol. 10, No. 4, April 2001, pp. 526-533