Application of Memory Scrambling Aware Multi-Level Diagnosis Flow

Suren Martirosyan

Yerevan State University
Yerevan, Armenia
e-mail: suren.martirosyan.92@gmail.com

ABSTRACT

To overcome the issues of safety, reliability and efficiency in nanoscale designs, advanced methods of fault detection and diagnosis were developed. Multi-level model based methods of fault detection were proposed using a hierarchy of detection and diagnosis methods and dynamic models, since previous approaches do not give a deeper insight and mainly limit or trend checking of some measurable output variables, which usually makes it impossible to do fault diagnosis. The new developed methods generate several symptoms indicating the difference between nominal and faulty statuses. Based on different symptoms, fault diagnosis procedures follow, determining the fault by applying the developed classification scheme. In this paper, the validity of a memory scrambling aware multi-level fault diagnosis flow is shown by experiments and different case scenarios.

Keywords

memory faults, multi-level diagnosis, fault classification, fault localization.

1. INTRODUCTION

Fault detection and diagnosis become increasingly important for the improvement of reliability, safety and efficiency in nanoscale designs. Memory scrambling aware multi-level diagnosis flow is proposed which provides end-to-end diagnosis information about the fault including its type, the corresponding defect classification, physical location of the fault as well as aggressor cell information in case of coupling faults. The proposed flow uses memory scrambling information to accurately calculate the physical address of the failed cells as well as its corresponding (X, Y) coordinates. All alternative solutions that do not consider memory scrambling information lead to an inefficient memory test since in that case exhaustive logical data patterns need to be applied to the memory for achieving the desired fault coverage. In [1], it is stated that lack of memory scrambling information can lead up to 35% test escapes.

Build-in self-test (BIST) systems usually build a test and repair infrastructure for memory devices which provides test mechanisms and background patterns for running test sequences and algorithms [2].

March test algorithms [3] are used to provide classification for static and dynamic faults using the proposed fault diagnosis flow.

In order to prove the validity of abovementioned flow, different case scenarios need to be considered, including multiple faults of different types in a single memory.

The rest of the paper is organized as follows. Section 2 describes the multi-level fault diagnosis flow. The experiments and results are shown in Section 3. Section 4 brings real-life case scenarios of the flow application. Section 5 concludes the paper.

2. MULTI-LEVEL FAULT DIAGNOSIS FLOW

In this section, a multi-level fault diagnosis flow is described which provides comprehensive information about the fault including its type, the corresponding defect classification, physical location of the fault as well as aggressor cell information in case of coupling faults.

Depending how many steps of diagnosis flow are needed to be achieved, it might be required to apply additional test algorithms or use memory structural information. First the flow identifies whether the memory instance has a fault or not. For this purpose, usually built-in self-test (BIST) is being run on system of chip. In case of faults, their logical addresses are identified in this level. Logical address of the fault is usually obtained by running test system in diagnostic mode. Next the physical address (i.e., physical row and column position) of faults and (X, Y) coordinates of failing cells in the memory are identified. This can be achieved by using memory scrambling information. After obtaining these data, the defect classification can be done to understand the distribution of defects and have initial view on defect types and causes. Defect distribution can be (see Fig. 1):

a. single bit;
b. vertical paired bits;
c. horizontal paired bits;
d. quadro bits;
e. column fail;
f. row fail.

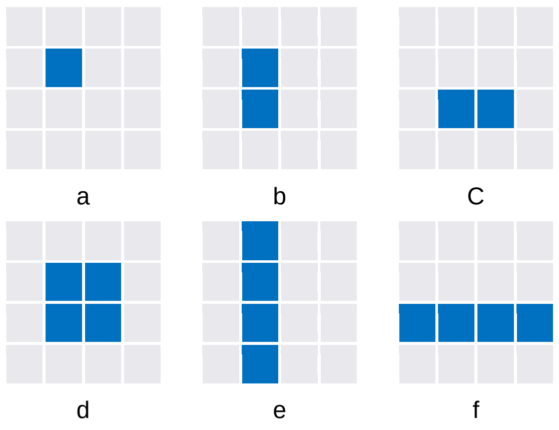

Fig. 1. Defect classification types

The fault classification is needed to identify the fault type (i.e., stuck-at, transition, coupling, etc.). At this step, special March test algorithms are applied, which are able to classify the fault types. In case of coupling faults, the logical and physical addresses of aggressor cells are identified. For that purpose, special March-like test algorithms are applied to localize the aggressor cell of the fault.
The flow is generic and can be applied also with other test algorithms targeting another set of faults.

3. EXPERIMENTAL RESULTS

We have used a BIST solution in our experiment. The solution allows to create test patterns which can be applied to the chip in the automatic test equipment (ATE) environment. It can process tester output files providing fault detection, classification and localization.
The experiment has been done on FPGA board which contains two 16nm memories: one single-port and one 2-port SRAMs (Static Random Access Memory).
In the first memory, we have injected the following faults:

- Stuck-at 0 in a cell with physical address (Row = 2, Column = 19).
- Coupling fault with victim cell physical address (Row = 32, Column = 46) and aggressor cell physical address (Row = 31, Column = 46).
- Transition 0 fault in a cell with physical address (Row = 39, Column = 10).
- Deceptive read destructive 0 fault in a cell with physical address (Row = 80, Column = 32).

The following faults have been injected in the second memory:

- Stuck-at 1 in a cell with physical address (Row = 10, Column = 15).
- Read destructive 0 fault in a cell with physical address (Row = 100, Column = 44).
- Write destructive 1 fault in a cell with physical address (Row = 71, Column = 55).

The test pattern which contains testing algorithms runs on both memories. As a result, memory test data have been generated for them. After processing test data log, the fault classification is done. Table 1 shows the obtained fault classification results.
All faulty cells have been identified, fault types and their signatures have been generated. The faults have been detected in the first phase of fault diagnosis flow. The BIST solution uses structural information of the memories. Therefore, it is possible to identify the corresponding logical (Address, Bit) and physical (Row and Column) addresses as well as physical coordinates (X, Y) of the faults.
All faults except of coupling fault in the first memory, have been detected and diagnosed in the first phase of diagnosis flow.
In order to identify the aggressor cell of coupling fault, additionally fault localization step needs to be applied. After running fault localization pattern on the first memory, the corresponding test data log is generated as an output. After processing the test data log, the fault localization (i.e., identification of aggressor cell information) is performed. Table 2 shows the obtained fault localization result. Since memory structural information is available then logical and physical addresses as well as physical coordinates of aggressor cell are also identified. The fault diagnosis for coupling fault has been completed.
The multi-level fault diagnosis flow has been applied to different chips of 45nm, 28nm and 16nm technology and enabled to do successful PFA. Section 4 describes several real-life case scenarios where the proposed flow has been applied.

4. REAL-LIFE CASE SCENARIOS

In the Scenario 1 the fault diagnostic pattern has been run on real memory. The test data log showed that 4 faulty cells were found during the first step. The logical addresses of faults were detected. With the help of memory scrambling data, the BIST solution was able to detect the physical addresses and coordinates of these faults. Defect classification was done from the obtained data. The results showed that there was a quadro bit in memory. The obtained information made possible doing physical failure analysis (PFA) to identify the cause of failure. Fig. 2 shows the

Table 1. Classification of Faults

#	BIST	Memory	Address	Bit	Row	Column	X	Y	FFM	FP
1	BIST1	RAM_1p	1	8	2	19	31.978um	14.6178um	SF	<1/0/->
2	BIST1	RAM_1p	60	22	32	46	24.418um	34.7193um	CFtr	<1; 0W1/0/->
3	BIST1	RAM_1p	74	4	39	10	22.654um	7.91725um	TF	<0W1/0/->
4	BIST1	RAM_1p	156	15	80	32	12.322um	24.2963um	DRDF	<R1/0/1>
5	BIST2	RAM_2p	17	6	10	15	29.962um	11.6398um	SF	<0/1/->
6	BIST2	RAM_2p	196	21	100	44	7.282um	33.2303um	RDF	<R0/1/1>
7	BIST2	RAM_2p	139	26	71	55	14.59um	41.4198um	WDF	<1W1/0/->

Table 2. Localization of Aggressor Cell

#	BIST	Memory	Victim Row	Victim Column	Aggressor Row	Aggressor Column	Aggressor Address	Aggressor Bit	Aggressor X	Aggressor Y
1	BIST1	RAM_1p	32	46	31	46	58	22	24.67um	34.7193um

Scenario 1.

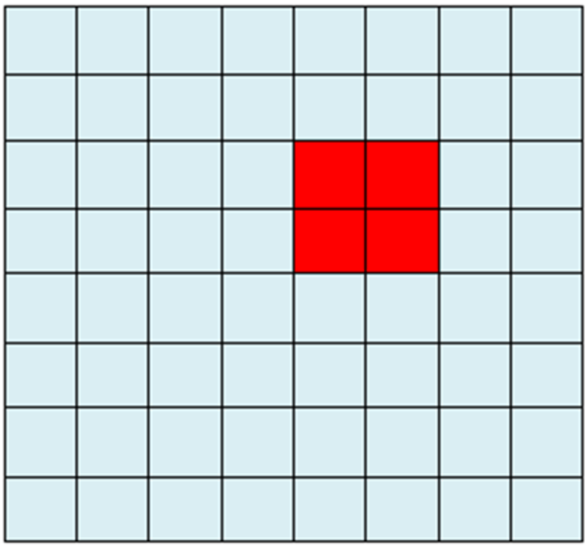
Fig. 2. Quadro bit defect

In the next scenario, the flow has been applied to a single port memory. Running fault diagnostic pattern and processing the generated data log the logical and physical addresses of faulty cells were identified. In the next step of multi-level fault diagnosis flow the defect classification have been done. The results showed that a whole column of memory was faulty and each fault found in the first step was on that column. As it turned out, the cause of failure was broken bit-line. The Scenario 2 is shown in Fig. 3.

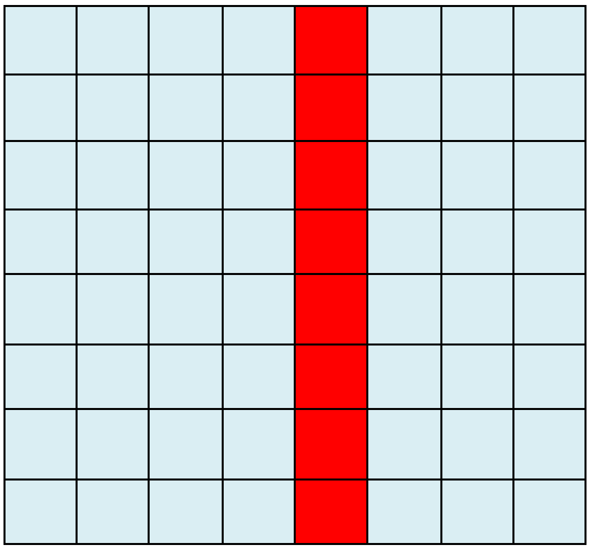
Fig. 3. Broken bit line

Fig. 4 shows the Scenario 3 where after applying fault diagnosis flow it turned out that 2 coupling faults exist in memory.
After applying fault localization pattern, we have found that there was only a single aggressor cell for both faults. Processing the data log of localization pattern, the exact cell coordinate and address have been identified. 2 weak cells have been impacted by a single aggressor cell. The final view of Scenario 3 is provided in Fig. 5.
In Scenario 4, after applying fault diagnosis flow to a memory, the obtained results showed that some of border cells are damaged. A solution to this problem is to put dummy (extra) rows and columns at the borders of memory to protect functional cells from damaging. Fig. 6 shows the locations of faulty cells in Scenario 4.

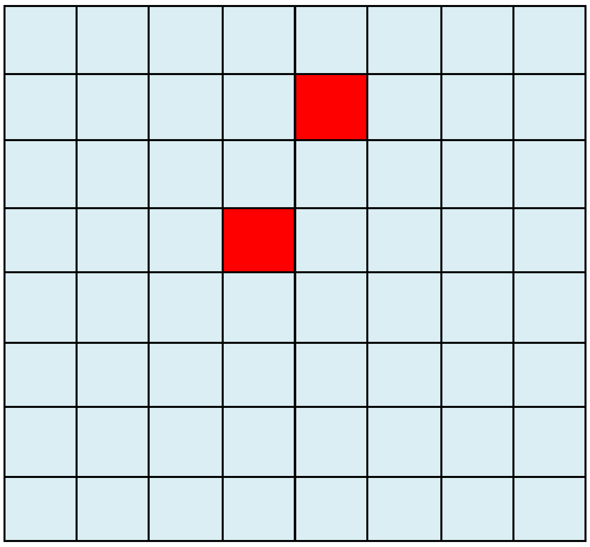
Fig. 4. Coupling faults

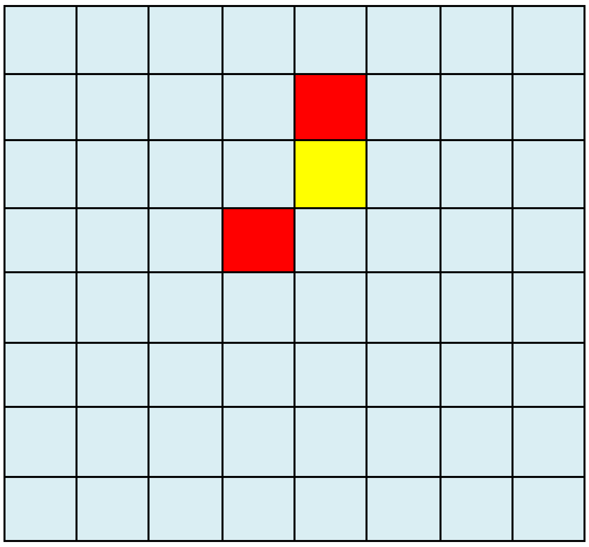
Fig. 5. Aggressor cell position marked yellow

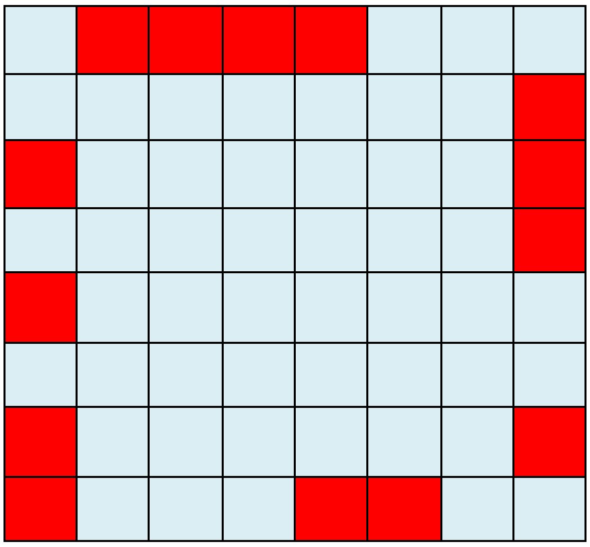
Fig. 6. Cells at border are damaged

5. CONCLUSION

In this paper, a memory scrambling aware multi-level fault diagnosis flow is shown. The flow is validated on 16nm FPGA board as well as it has been applied to numerous chips of 45nm, 28nm and 16nm technologies enabling successful physical failure analysis (PFA). Some real-life case scenarios of the flow application are presented at the end of the paper.

REFERENCES

[1] A.J. van de Goor, I. Schanstra, "Address and Data Scrambling: Causes and Impact on Memory Tests", *IEEE International Workshop on Electronic Design*, pp. 128-137, 2002.

[2] Y. Zorian, S. Shoukourian, "Embedded-Memory Test and Repair: Infrastructure IP for SoC Yield", *IEEE Design and Test of Computers*, pp. 58-66, 2003.

[3] A.J. van de Goor, “Testing semiconductor memories: Theory and Practice”, *John Wiley & Sons*, 1991.

[4] K. Darbinyan, G. Harutyunyan, S. Shoukourian, V. Vardanian, Y. Zorian, “A Robust Solution for Embedded Memory Test and Repair”, *IEEE Asian Test Symposium*, pp. 461-462, 2011.

[5] S. Hamdioui, A.J. van de Goor, M. Rodgers, “March SS: a test for all static simple faults”, *IEEE Workshop MTDT*, pp. 95-100, 2002.

[6] S. Hamdioui, G. N. Gaydadjiev A.J.van de Goor, “A Fault Primitive Based Analysis of Dynamic Memory Faults”, *IEEE Workshop on Circuits*, pp 84-89, 2003.

[7] M. Jurczak, N. Collaert, A. Veloso, T. Hoffmann, S. Biesemans, “Review of FinFET Technology”, *IEEE International SOI Conference*, pp. 1-4, 2009.

[8] H. W. Cheng, Y. Li, “16-nm Multigate and Multifin MOSFET Device and SRAM Circuits”, *International Symposium on Next-Generation Electronics*, pp. 32-35, 2010.

[9] G. Harutyunyan, G. Tshagharyan, V. Vardanian, Y. Zorian, “Fault Modeling and Test Algorithm Creation Strategy for FinFET-Based Memories”, *IEEE VLSI Test Symposium*, pp. 49-54, 2014.

[10] G. Harutunyan, V. Vardanian, “A March Test for Full Diagnosis of All Simple Static Faults in Random Access Memories”, *IEEE East-West Design and Test Workshop*, pp. 68-71, 2006.

[11] G. Harutyunyan, D. Melkumyan, “Fault Location and Diagnosis Algorithm for Static and Dynamic Faults in SRAMs”, *Proceedings of the National Academy of Sciences of Armenia and the State Engineering University of Armenia*, pp. 280-287, 2010.

[12] G. Harutyunyan, A. Hakhumyan, S. Shoukourian, V. Vardanian, Y. Zorian, “Symmetry Measure for Memory Test and Its Application in BIST Optimization”, *Journal of Electronic Testing: Theory and Applications*, pp. 753-766, 2011.

[13] J.-F. Li, K.-L. Cheng, C.-T. Huang, and C.-W. Wu, “March based RAM diagnostic algorithms for stuck-at and coupling faults”, *IEEE ITC*, pp. 758-767, 2001.