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. 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):

- single bit;
- vertical paired bits;
- horizontal paired bits;
- quadro bits;
- column fail;
- 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 SCENариOS

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

| Scenario 1. |

---

### Table 1. Classification of Faults

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

### Table 2. Localization of Aggressor Cell

<table>
<thead>
<tr>
<th>#</th>
<th>BIST</th>
<th>Memory</th>
<th>Victim</th>
<th>Victim</th>
<th>Aggressor</th>
<th>Aggressor</th>
<th>Aggressor</th>
<th>Aggressor</th>
<th>Aggressor</th>
<th>Aggressor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BIST1</td>
<td>RAM_1p</td>
<td>32</td>
<td>46</td>
<td>31</td>
<td>46</td>
<td>58</td>
<td>22</td>
<td>24.67um</td>
<td>34.7193um</td>
</tr>
</tbody>
</table>

---
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.

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. 2. Quadro bit defect
Fig. 3. Broken bit line
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