An Area-efficient Built-in Redundancy Analysis for Embedded Memories with Optimal Repair Rate using 2-D Redundancy

Joohwan Lee, Kihyun Park, and Sungho Kang
Dept. of Electrical & Electronic Engineering
Yonsei University
Seoul, Korea
{eldsich, cimabear, shkang}@yonsei.ac.kr

Abstract—A novel built-in redundancy analysis (BIRA) is proposed for embedded memories. The proposed BIRA algorithm has two phases. In the first phase, detected faults are collected to area-efficient fault storing content addressable memory (CAM). In order to determine a correct repair solution, spare memories are allocated in the second phase using linear feedback shift register (LFSR) with fast analyzing speed. Experimental results show that the proposed BIRA algorithm achieves optimal repair rate and very low area overhead.

Keywords—built-in redundancy analysis; optimal repair rate; embedded memory; built-in self-test

I. INTRODUCTION

With the increasing complexity of VLSI devices, high-density and high-capacity embedded memories are often implemented in a system chip. Since embedded memories occupy the majority of the die area, the yield of the embedded memories will dominate the yield of system-on-a-chip (SoC). Most SoCs adopt a built-in self-test and built-in redundancy analysis to test and repair their embedded memories instead of using external ATE because this method is more cost-effective.

The approaches using a 2-D spare architecture have both spare rows and columns, and these are better in terms of repair efficiency than a 1-D spare architecture (with spare rows or columns). Various redundancy analysis (RA) algorithms for 2-D redundancy have been developed [1–7]. Among these algorithms, repair-most (RM) algorithm [1], CRESTA [2], ESP [3], and IntelligentSolveFirst (ISF) [6] are the most popular RA algorithms for built-in redundancy analysis (BIRA). These BIRA algorithms have weakness as well as strength. RM is a greedy algorithm, and though its repair rate is high, it is not optimal. CRESTA achieves 100% detection ability of the repairable chips with fast analyzing speed. However, its area overhead increases with an increasing number of redundancies. ESP has low area overhead with simple RA algorithms, but the repair rates of ESP are not optimal because of their excessive omission of faulty information. ISF mainly focuses on high quality results with less hardware and shorter analysis time. However, the RA speed of ISF in cases of complex fault distributions is still not fast enough for commercial production purposes. Reference [7] has the smallest area overhead but does not have optimal repair rate.

In order to overcome the weakness of previous algorithms, an area-efficient BIRA is proposed for embedded memories with optimal repair rate using 2-D redundancy. The rest of this paper is organized as follows. In Section 2, a new BIRA algorithm will be proposed with two phases. Experimental results show that the proposed BIRA algorithm achieves optimal repair rate and very low area overhead in Section 3. Section 4 concludes this paper.

II. PROPOSED BIRA ALGORITHM

A. Overview of the proposed BIRA

We propose a BIRA algorithm with two phases. The first phase collects faults. This fault collection (FC) phase only once records detected faults to parent and child address CAM with low area overhead. Whole new fault information is stored into parent address CAM. However, other faults sharing fault information with parent address CAM are stored into child address CAM. After finishing FC phase, spare allocation (SA) phase begins. During the SA phase, repair strategies are provided using LFSR [8] and a repair solution is determined correctly with fast analyzing speed.

B. Fault collection

In order to minimize the area overhead, a novel fault storing CAM is proposed for an M by N memory block using 2-D spare architecture with Rs spare rows and Cs spare columns as shown in Table I and Table II. Table I shows the fields of parent address CAM and the maximum entry number of parent address CAM is Rs+Cs. Table II shows the fields of child address CAM and the maximum entry number of child address CAM is Rs(Cs-1)+Cs(Rs-1). In Table I and Table II, the meaning of acronyms and the sizes of fields for the CAM

The authors would like to appreciate CAD tool support from IC Design Education Center (IDEC).
The proposed BIRA algorithm achieves very low area overhead; because previous RA algorithms have the rooms for final repair solution separately but parent address CAM in Table I include the space for final repair solution.

The fault collection flow chart of the proposed BIRA is shown in Fig. 2. In Fig. 2, both early termination conditions and must-repair conditions to speed up the BIRA analysis speed were already introduced in [3]. In the case of the early termination conditions, the process of FC is terminated immediately. When the must-repair conditions meet, if available spare elements do not exist, the process of FC is also ended. Whenever carrying out the must-repair process, one spare element becomes unavailable. When there is no fault storing CAM as well as previous two cases, the FC process is aborted.

In Fig. 2, a newly detected leading fault is stored into parent address CAM. A leading fault does not share its row and column address with already stored faults. Other faults are stored into child address CAM. Since all the detected faults in reparable memory chip can be recorded in the proposed area-efficient fault storing CAM without losing failure information, optimal repair rate can be obtained.

C. Spare allocation

In order to solve the NP-complete spare allocation problem, we use exhaustive repair strategies using LFSR [8]. To determine repair strategy using LFSR has the advantage of area overhead. In the strategy indicated by the contents of the LFSR, a '1' denotes a spare row and '0' denotes a spare column. For example, consider the 2R2C (two spare rows and two spare columns) case shown in Fig. 3. The initial strategy is 'row, column, column, row' or 1001 as shown in the underlined number of Fig. 3. The next repair strategy for this case will be '1100', i.e. 'row, row, column, column' after the LFSR is updated. After 5 updates, the LFSR generates all 6 repair strategies: 1001, 1100, 0110, 0101, 1010, and 0011. A correct repair solution can be determined by using the fields (FFC, MR, RR, RFC, MC, RC, and CFC) of parent address CAM.

The spare allocation flow chart of the proposed BIRA is shown in Fig. 4. In Fig. 4, repair strategy is provided in order by LFSR. SA phase are continued until all repair strategies are provided or memory under test is fixed.

### Table I. Parent Address CAM

<table>
<thead>
<tr>
<th>Acronym</th>
<th>CAM content</th>
<th>CAM size</th>
</tr>
</thead>
<tbody>
<tr>
<td>PE</td>
<td>Parent enable flag</td>
<td>1</td>
</tr>
<tr>
<td>FFC</td>
<td>Fault from child flag</td>
<td>1</td>
</tr>
<tr>
<td>MR</td>
<td>Must row enable flag</td>
<td>1</td>
</tr>
<tr>
<td>RR</td>
<td>Repaired by spare row flag</td>
<td>1</td>
</tr>
<tr>
<td>RA</td>
<td>Row address</td>
<td>[\log_2 M]</td>
</tr>
<tr>
<td>RFC</td>
<td>Row repeat fail count</td>
<td>[\log_2 CS]</td>
</tr>
<tr>
<td>MC</td>
<td>Must column enable flag</td>
<td>1</td>
</tr>
<tr>
<td>RC</td>
<td>Repaired by spare column flag</td>
<td>1</td>
</tr>
<tr>
<td>CA</td>
<td>Column address</td>
<td>[\log_2 N]</td>
</tr>
<tr>
<td>CFC</td>
<td>Column repeat fail count</td>
<td>[\log_2 RS]</td>
</tr>
</tbody>
</table>

### Table II. Child Address CAM

<table>
<thead>
<tr>
<th>Acronym</th>
<th>CAM content</th>
<th>CAM size</th>
</tr>
</thead>
<tbody>
<tr>
<td>CE</td>
<td>Child enable flag</td>
<td>1</td>
</tr>
<tr>
<td>SF</td>
<td>Row/column select flag</td>
<td>1</td>
</tr>
<tr>
<td>PCP</td>
<td>Parent address CAM pointer</td>
<td>[\log_2 (RS+CS)]</td>
</tr>
<tr>
<td>RCA</td>
<td>Row/column address</td>
<td>[\log_2 (\max(MN))]</td>
</tr>
</tbody>
</table>
III. EXPERIMENTAL RESULTS

The performance of a BIRA algorithm can be measured by repair rate and area overhead. For the comparison of repair rate, we assumed a 512×512 memory array and the number of spare row RS and spare column CS were also assumed 2 and 4, respectively. To guarantee the effectiveness of the proposed algorithm on various processes, each experiment was repeated 10,000 times with 6 to 20 faults, respectively. Injected faults were scattered in the 12x12 zone so as to introduce various fault types (single faulty cell, faulty row, faulty column, cluster fault). Fig. 5 shows the average repair efficiency for each scheme according to the number of faults. The proposed algorithm guarantees optimal repair rate.

Generally, most BIRAs include storages to restore failure information. The area of storage requirements is not exactly the same as the area of the whole BIRA but the area of storage cells dominates the BIRA. For the comparison of area overhead, equations to calculate storage requirements for various BIRA are shown below. Aspare_register, ACRESTA, AESP, and APROPOSED from (1) to (5) represent the number of bits required for spare register and each algorithm.

\[
A_{\text{sparse\_register}} = ([\log_2 M] + 1) \times RS + ([\log_2 N] + 1) \times CS + 1
\]

(1)

\[
A_{\text{CRESTA}} = A_{\text{sparse\_register}} \times (RS + CS)! / (RS! \times CS!)
\]

(2)

\[
A_{\text{ESP}} = (RS + CS) \times ([\log_2 M] + [\log_2 N] + 2) + A_{\text{sparse\_register}}
\]

(3)

\[
A_{\text{ISP}} = 2RS\times CS \times ([\log_2 M] + [\log_2 N] + 1) + 2RS\times CS \times ([\log_2 RS] + [\log_2 CS])
\]

(4)

\[
A_{\text{PROPOSED}} = (RS + CS) \times ([\log_2 M] + [\log_2 CS] + 3) + (RS + CS) \times ([\log_2 N] + [\log_2 RS] + 3) + (RS \times (CS - 1) + CS \times (RS - 1)) \times ([\log_2 RS] + [\log_2 CS] + 2)
\]

(5)

According to the area estimation, the proposed BIRA algorithm has the smallest storage cells shown in Fig. 6 except ESP. However, ESP cannot achieve optimal repair rate.

In conclusion, Table III compares the proposed BIRA algorithm with the popular algorithms [1–3, 6] and the other previous algorithm [7] for repair rate and area overhead. Reference [7] is excluded in Fig. 5 and Fig. 6, because the results of [7] are similar to those of ESP [3].

IV. CONCLUSION

A novel BIRA algorithm is proposed for embedded memories with optimal repair rate and very low area overhead. The proposed BIRA algorithm consists of fault collection phase and spare allocation phase. In the first phase, detected
faults are collected to area-efficient fault storing CAM. In order to determine correct repair solution, spare memories are allocated in the second phase using LFSR. Experimental results show that the proposed BIRA algorithm achieves optimal repair rate and very low area overhead.

![Area estimation for RA algorithms](image)

**TABLE III. PERFORMANCE COMPARISON OF BIRAS**

<table>
<thead>
<tr>
<th>BIRA algorithm</th>
<th>Repair rate</th>
<th>Area overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>RM [1]</td>
<td>High</td>
<td>Very high</td>
</tr>
<tr>
<td>CRESTA [2]</td>
<td>Optimal</td>
<td>High</td>
</tr>
<tr>
<td>ESP [3]</td>
<td>Low</td>
<td>Extremely low</td>
</tr>
<tr>
<td>ISF [6]</td>
<td>Optimal</td>
<td>Low</td>
</tr>
<tr>
<td>[7]</td>
<td>Low</td>
<td>Extremely low</td>
</tr>
<tr>
<td>PROPOSED</td>
<td>Optimal</td>
<td>Very low</td>
</tr>
</tbody>
</table>

**REFERENCES**


