A New BIST Scheme with Encoding Logic to Achieve Complete Fault Coverage

Ling Zhang, Junjin Mei, Guan-zhong Wang, Tonghan Li
School of Computer,
Hubei Polytechnic University,
Huangshi, 435003, China

Abstract—Built-in Self Test (BIST) has been proved as one of the effective design for testability techniques, where on-chip test architectures are designed to test the digital circuits themselves. To reduce test application time and improve fault coverage, A deterministic Built-in Self Test (BIST) technique that can get complete fault coverage without using any storage device is proposed in this paper. The test architecture contains a novel on chip encoding logic that generates all required test vectors in real time. Experimental results show that the proposed technique requires much less test application time to achieve complete fault coverage for all testable stuck-at faults with reasonable hardware cost.

Keywords—Built-in Self Test (BIST); test application time; fault coverage

I. INTRODUCTION

Logic BIST architecture commonly uses linear feedback shift registers (LFSRs) as test generator for pseudo-random testing. The architecture is simple and effective. However, because complex circuits often contain some hard-to-detect faults that random-pattern resistant, the conventional BIST often requires long test application time, and hard to achieve high fault coverage. To solve these problems, many of techniques have been proposed. Test point insertion [5-8], weighted pattern testing [9-11], and reseeding [12] are three most popular logic BIST schemes.

The test point insertion scheme inserts test points into circuits under test (CUT) intrusively to improve random testability. Timing sequence of the CUT is affected at the same time, which is not desired for high performance or critical designs. For biggest circuits, there are lots of unspecified bits in the corresponding test cubes. Reseeding techniques take advantage of this fact, which use a linear feedback state register (LFSR) to encode for the specified bits for a test cubes by solving the linear equation, and gain the appropriate seed. To generate more effective random patterns, the weighted random pattern testing uses a weighted bit stream to bias the weight of the bit stream to inputs of circuit under testing. For achieving the target fault coverage, many weights must be stored in an ROM then resulting in large are overhead.

There are other BIST schemes also to reduce test data volume, test power or test application time. Paper [12] proposes a input vector monitoring scheme for concurrent BIST for on-line testing. To reduce hardware cost of paper [12], a new Multilevel Decoding Logic is proposed in paper [13]. Paper [14] presents a novel BIST architecture for compute the number of switch of the circuits to reduce the test power. Paper [15] illustrates a BIST architecture for fault diagnosis that can be used to identify permanent failures in embedded ROM. A Complete Logic BIST Technology with No Storage Requirement was presented in paper [16]. Though these BIST schemes reduce the size of test set efficiently, and some need not storage memory, the long test application time and the high hardware cost are also the difficulties of the BIST.

In this paper, a novel deterministic BIST is proposed to reduce test application time and improve fault coverage. A pre-generated test set is encoded as a combinational circuit called as Encoding Logic Unit (ELU). It can generates all required test vectors without use storage device, and corresponding test application time is reduce more compared with the previous BIST schemes with reasonable hardware cost.

The rest of the paper is organized as follows. The proposed BIST scheme is described in Section 2. In Section 3, The simplify of module of ELU is presented. Experimental results are given in Section 4. Finally, we conclude our paper in Section 5.

![Figure 1. the architecture of the deterministic BIST](image)

II. PROPOSED BIST SCHEME

The schematic of the proposed deterministic BIST is given in Figure 1. It contains a Encoding Logic Unit (ELU), a counter and the response monitor. All required test vectors are generated by the ELU along with the counter. Test responses are captured by the response monitor. The deterministic test sets are encoded as the unit of the ELU, then just a simple counter can generate all vectors for the CUT. The CUT’s primary inputs and scan cells are connected to the Encoding Logic Unit directly, and the ELU is designed as a simple combinational circuit that contains $m$ outputs and $\log_2 n$ inputs, where $m$ equals to the number of inputs of the CUT and $n$ equals to the number of vectors. During test process, each value of the counter corresponds to a test vector. Thus, all required test vectors can be generated through the operation of the counter until achieve 100% fault coverage.
The critical point of the proposed architecture is the ELU design. The number of inputs of the CUT is assumed as m, and the number of pre-generated deterministic vectors is n. The design of ELU can be described as follows: to encode these test vectors, the number of inputs for the ELU is designed as $n = \log_2 n = 2$, and the number of outputs is designed as m that for applied these vectors to the CUT. All minterms of the inputs make up the inputs value of the ELU, and all outputs could be calculated by the logic operation of these minterms.

To clearly present the ELU design, an example of well-known ISCAS’85 C17 benchmark test set \cite{10} is given in Table II. The test set consists of four test vectors that denoted as **Input Test matrix** in the table, and the corresponding responses that presented as **Output Response matrix** are also given in the table. The number of vectors n=4, and the number of inputs m=5. The number of the ELU’s inputs is computed as $n = \log_2 4 = 2$, the number of outputs is equal to the number of inputs of CUT that is 4. To design the ELU for this test set of C17 benchmark circuit, the truth table of the ELU is listed in Table II. As shown the table, all minterms of the inputs formed all values of the ELU’s inputs, so each output of the ELU can be computed easily from the minterms of these inputs. The outputs computation results for Table I are shown as expression(1) expression(2), expression(3) and expression(4).

\[
\begin{align*}
y_1 &= \sum (M1, M2) = x_1x_2 + x_1x_2 \\
y_2 &= \sum (M1, M4) = x_1x_2 + x_1x_2 \\
y_3 &= \sum (M1, M3) = x_1x_2 + x_1x_2 \\
y_4 &= \sum (M2, M4) = x_1x_2 + x_1x_2 + x_1 = x_1 + x_2 \end{align*}
\]

### Table II. The Truth Table of ELU For Example C17

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>x2</td>
</tr>
<tr>
<td>y1</td>
<td>y2</td>
</tr>
<tr>
<td>M1</td>
<td>0</td>
</tr>
<tr>
<td>M2</td>
<td>0</td>
</tr>
<tr>
<td>M3</td>
<td>1</td>
</tr>
<tr>
<td>M4</td>
<td>1</td>
</tr>
</tbody>
</table>

The ELU architecture of the proposed scheme for C17 is given in Figure 2. As shown in the figure, corresponding to truth table of Table II, the unit contains two inputs x1 and x2 that are actuated by a simple counter which can share with the on-chip counters, five outputs y1 to y5 connect directly to the inputs of CUT. All minterms are also given in the figure, which are denoted as M1, M2, M3 and M4. The operation process can be described as follows: the counter generates each minterm of the inputs, which will generates a determined test vector through the ELU architecture. Until all minterms generated, then the determined test sets are all applied to the CUT with 100% fault coverage. To reduce the hardware cost, the architecture could be optimized by simplify schemes, and several simplify schemes are given in next section.

**III. SIMPLIFY ALGORITHM FOR ELU UNIT**

The intuitively scheme to reduce the number of logic gates is merging the compatible columns of the test set. As shown in Table I, the logic value of y3 and y5 are always same, so they could be merged to as one output broadcast to two inputs input 3 and input 5 of CUT. As shown in Figure 3, the number of output reduced to 4, and the number of logic gate are decreased to 10. If the combinational ELU module with n inputs and m outputs is considered, it encodes for a pre-computed test set of CUT with 100% fault coverage. The test set contains x columns, which contains j compatible columns. Then the
number of outputs are reduced to \( x - j + 1 \), at the same time, the logic gate will be reduced by \( j - 1 \).

The second simplify scheme to reduce hardware cost of ELU is the Boolean algebra simplifying methods. By doing this, some redundancy variable will disappear from the logic expression of the outputs, and the corresponding architecture sure becomes simple. Take C17 for example too, the expression of \( y_1 \) presented as:

\[
y_1 = \overline{x_1}x_2 + x_1\overline{x}_2
\]

Using the rule of \( (\overline{A} + A) = 1 \) \( (A \text{ is a Boolean variable}) \)

\[
y_1 = \overline{x_1}x_2 + x_1\overline{x}_2 = (\overline{x}_1 + x_1)x_2 = x_2
\]

where variable of \( x_1 \) disappear from the expression by Boolean algebra simplifying scheme. The other outputs of the ELU are simplified in the same way, the results are given as expression(5), expression(6) and expression(7). The last architecture is given in Figure 4, the output is reduced as 4, and the number of logic gates is reduced to 6.

\[
y_2 = x_1x_2 + x_1x_2
\]

\[
y_3 = y_2 = \overline{x_1}x_2 + x_1\overline{x}_2 = \overline{x_1}(\overline{x}_2 + x_2) = \overline{x}_1
\]

\[
y_i = \sum(M1,M2,M4) = \overline{x}_1x_2 + \overline{x}_1x_2 + x_1x_2 = \overline{x}_1 + \overline{x}_1 = x_1 + 1
\]

**IV. EXPERIMENTAL RESULTS**

In this section, the experimental results for the proposed BIST for ISCAS’85 and ISCAS’89 benchmarks are implemented. Test sets generated by ATALANTA\cite{17} is used in the experiments. The simplified schemes for ELU module are achieved by using GUN C programming language. The experimental results are given in Table III.

Table III gives the experimental results of test application time, the test application time and the hardware costs are both given in the table. The first three columns give the circuit name(CN), the number of ports(#P), the number of testable faults, the number of cycles required is denoted as #Cycles. The test application time is compared with related BIST solutions without storage memories, as shown in the table, they contains paper\cite{2}, paper\cite{9}, paper\cite{11} and paper\cite{16}. The last column is the test application time and the hardware cost of ours where the hardware cost is measured by the number of gate they contain. It is easy to see that the proposed scheme needs much less test application time with reasonable.

<table>
<thead>
<tr>
<th>CN</th>
<th>#P</th>
<th>#TF</th>
<th>Test application time</th>
<th>#Cycles</th>
<th>proposal</th>
<th>#Cycles</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>C432</td>
<td>36</td>
<td>523</td>
<td>1024</td>
<td>-</td>
<td>-</td>
<td>279</td>
<td>51</td>
</tr>
<tr>
<td>C499</td>
<td>41</td>
<td>750</td>
<td>1024</td>
<td>-</td>
<td>-</td>
<td>298</td>
<td>54</td>
</tr>
<tr>
<td>C880</td>
<td>60</td>
<td>942</td>
<td>2048</td>
<td>1K</td>
<td>980</td>
<td>278</td>
<td>52</td>
</tr>
<tr>
<td>C1355</td>
<td>41</td>
<td>1566</td>
<td>4096</td>
<td>2K</td>
<td>1046</td>
<td>595</td>
<td>85</td>
</tr>
<tr>
<td>C1908</td>
<td>33</td>
<td>1870</td>
<td>8192</td>
<td>4K</td>
<td>3327</td>
<td>900</td>
<td>110</td>
</tr>
<tr>
<td>C2670</td>
<td>233</td>
<td>2630</td>
<td>6144</td>
<td>5K</td>
<td>1002</td>
<td>343</td>
<td>110</td>
</tr>
<tr>
<td>C3540</td>
<td>50</td>
<td>3291</td>
<td>32768</td>
<td>4.5K</td>
<td>-</td>
<td>919</td>
<td>152</td>
</tr>
<tr>
<td>C5315</td>
<td>178</td>
<td>5293</td>
<td>4096</td>
<td>-</td>
<td>-</td>
<td>507</td>
<td>122</td>
</tr>
<tr>
<td>C6288</td>
<td>32</td>
<td>7710</td>
<td>1024</td>
<td>-</td>
<td>-</td>
<td>40</td>
<td>32</td>
</tr>
<tr>
<td>C7552</td>
<td>207</td>
<td>7419</td>
<td>32768</td>
<td>8K</td>
<td>3958</td>
<td>737</td>
<td>209</td>
</tr>
</tbody>
</table>

As can be seen from Figure 1, Figure 2 and Table I, The optimal results are very close. But the CPU time of Approach 1 is far less than that of Approach 2. It can be seen that the proposed algorithm.

**V. CONCLUSIONS**

The paper proposes a new novel deterministic BIST architecture without using any storage memory. It contains a counter which can shared with the counter on-chip, ELU module that used to generate all test vectors for
achieve 100% fault coverage. To reduce the area cost of the ELU, two simplify schemes are also given in the paper. The experimental results also proved the effectiveness of the proposed scheme.

VI. ACKNOWLEDGEMENTS

The research work was supported by Natural Science Foundation of Hubei Provincial under Grant (2011CDC107), Educational Commission of Hubei Province (B2013061), Hubei Polytechnic university (No. 13xjz05c) and Outstanding Youth Science and Technology Innovation Team Project of Hubei Polytechnic University (13xtz10).

REFERENCES


