Deterministic and Low Power BIST Based on Scan Slice Overlapping

Ji Li  Yinhe Han  Xiaowei Li
Institute of Computing Technology, CAS, Beijing, 100080, China
Graduate School of the CAS, Beijing, 100039, China
Leeji1212@ict.ac.cn

Abstract
This paper presents a new deterministic pattern generation structure that can be used in conjunction with any LFSR reseeding scheme. The proposed scheme utilizes Scan slices overlapping to reduce the number of specified bits and the number of transitions at the same time. Thus, it can significantly reduce test power and even further reduce test storage. A decoder is used to generate control signals. Experimental results indicate that the proposed method significantly reduces the switching activity by 80% and only needs relatively small test data storage.

1. Introduction
When the process of very large-scale integrated circuits (VLSI) scales down into deep sub-micron, the complexity of circuit designs has greatly increased. Testing of a chip has posed some new challenges, especially with the emergence of System-on-chip (SOC). Traditional test methods based on the automatic test equipment (ATEs) are becoming unacceptable. Built-in self-test(BIST) offers an attractive solution to these challenges.

In recent years, research on deterministic BIST has become a hot topic, which can provide very high test coverage and relatively short test time. The architecture of deterministic BIST mainly consists of three parts: test data storage, stimulus decoder, and response compactor. Many kinds of decoders have been proposed to reduce test data storage, such as LFSR[1-3], decoder of encoding compression[4-8], XOR network[9] and the other customized counters[10]. Among these decoder schemes, LFSR reseeding[1-3] generate deterministic test cubes by expanding seeds, providing significant reduction of test data storage and bandwidth. Therefore LFSR reseeding technique has been studied extensively, and supported by many commercial tools including TestKompres by Mentor Graphics[11] and SmartBIST by Cadence[12].

While LFSR reseeding is powerful for test data compression, it is not so good for power consumption. BIST implementation randomly fills don’t care bits in the test cubes, which can result in excessive switching activities when they are shifted into a scan chain. Observing the X-bit distribution in test cubes can help us to find a method to handle this problem. Mintest patterns are examined. We get an observations from the result: The density of X-bit in circuit under test is high. To S13207, the X-bit occupies up to 93.2% of total test bits. “High density” of X-bit can make two consecutive scan slices in test patterns have a high probability to be overlapping. Scan slice is defined to be the set of inputs applied to the scan chain inputs at a scan cycle. Scan slices overlapping means that scan slices are same. See the example in Figure 1, the pattern contains 10 scan slices. Each slice consists of 4 bits. The first and second slice in the original pattern are: {{1 1 X X}, {X X 1 1}}. If the second and third bits in the first slice and the first and second bits in the second slice are assigned to 1, they will be {{1 1 1}, {1 1 1}} and overlapping.

![Figure 1. Slices overlapping in test patterns](https://example.com/slices.png)

2. Scan Slices Overlapping
There are many don’t care bits in test cubes. Conventionally, in BIST environment, the don’t care bits are filled with random values, which result in excessive switching activity when they are shifted into a scan chain. Observing the X-bit distribution in test cubes can help us to find a method to handle this problem. Mintest patterns are examined. We get an observations from the result: The density of X-bit in circuit under test is high. To S13207, the X-bit occupies up to 93.2% of total test bits. “High density” of X-bit can make two consecutive scan slices in test patterns have a high probability to be overlapping. Scan slice is defined to be the set of inputs applied to the scan chain inputs at a scan cycle. Scan slices overlapping means that scan slices are same. See the example in Figure 1, the pattern contains 10 scan slices. Each slice consists of 4 bits. The first and second slice in the original pattern are: {{1 1 X X}, {X X 1 1}}. If the second and third bits in the first slice and the first and second bits in the second slice are assigned to 1, they will be {{1 1 1}, {1 1 1}} and overlapping.

![Figure 1. Slices overlapping in test patterns](https://example.com/slices.png)

3. Test Pattern Generation
Rather than using LFSR reseeding to directly encode the specified bits as in conventional LFSR reseeding, we can divide the test cube into blocks and slices of a block overlap each other. We call this kind of block as overlapping block. Then, for each test cube, when calculating the seed of this cube by solving equations, we can only take the first slice in every block into account.

![Figure 1. Slices overlapping in test patterns](https://example.com/slices.png)
That is, when the LFSR expands a seed into a test pattern to randomly fill the scan chains, only the value of the first slice in every block is generated from the LFSR, and the value of the rest slices can be got by directly shifting the value of the first slices of blocks. After the scan chains have been filled, the test pattern is applied to the circuit under test and the response is shifted out to MISR. The process is then repeated to generate the next test pattern.

4. Pattern partition

According to the proposed architecture, the pattern is needed to be partition into some overlapping slice sets. Obviously, in order to utilize the overlapping of consecutive scan slices to reduce the specified bits and transitions furthest, we should minimize the number of blocks. The partitions which can lead to the minimal number of blocks are not unique, and different partitions schemes may result in different results. A simple example is shown. See Figure 3, the original pattern contains 7 scan slices. Because of incompatibility between the third scan slice and the fifth scan slice, the original pattern should be divided into two overlapping blocks and the incompatible two scan slices should be contained in these two blocks respectively. Note that the forth scan slice is compatible with the two incompatible scan slice at the same time, which results in two different partition schemes as shown in Figure 3. We can see, after combination of compatible scan slices, total number of specified bits in the first partition scheme is 7, more than that in the second partition scheme which is reduced to 6.

4. Pattern partition

An example of the pattern partition procedure is shown in Figure 4. Figure 4-(1) shows the original pattern. In Figure 4-(2), the original pattern is divided into overlapping blocks by the algorithm 1. In Figure 4-(3), in every overlapping block, the specified bits are combined into the first slice and the assignments of all slices are equal. Thus, the LFSR only need to generate the specified bits in the first slice of every block after partition and combination, and don’t care bits in this block are specified.

This approach reduces the number of transitions in the scan chains and also reduces the total number of specified bits that must be generated by the LFSR as compared with conventional LFSR reseeding. It can be seen that the total number of specified bits in the original pattern is 16, and the total number is reduced to 10 after partition in this example.
Besides the reduction of number of specified bits, the total number of transitions is also greatly reduced in this way. Now we make a simple analysis about reduction of transitions. Because the don’t care bits in each test cube get filled with random values by the LFSR, the transitions may happen to every neighbor scan cell in original pattern. When the pattern is divided into overlapping blocks, the transitions only happen between blocks. Obviously, the total number of transitions can be reduced after pattern partition.

It must be noticed that the control signals become the one part of total test storage. Therefore in order to reduce test storage, the control signals should be compressed. This can be done using fixed run-length encoding.

5. Experimental results

To validate the efficiency of the proposed method, experiments were performed on the largest ISCAS 89 benchmark circuits. Test sets used in this experiment were obtained using Mintest dynamic compaction [15].

Table 1 presents the results. Each test cube is divided into overlapping blocks, and the number of blocks is shown in each case. The number of specified bits and the number of transitions required for the proposed scheme is compared with the original test cubes. The number of transitions is calculated as described in [16]. Column NP, NC and NB respectively show number of patterns, the number of scan chains and the corresponding number of overlapping blocks. Column NCB shows the total number of control signals before compression. Column T gives the size of the original test sets. The total number of specified bits after our method is applied to the test cube is shown in column partNSB. The last column of storage shows the change of test data compression and the power consumption is relatively very small to the power generation is not included in the power computing list in [4,13] and the proposed scheme is given in Table3. To the each scheme, the first column shows the required storage and the second column shows the power reduction. Note that the power consumption resulting from the test pattern generation is not included in the power computing list in the table (the same to [4] and [13]), since this part power consumption is relatively very small to the power.

![Figure5. Circuit s13207: Comp% vs. Pred%](image)

Table 2 provides a test data compression comparison between the proposed scheme and the best previously published results[5-8], whose compression ratios are all relative to the size of the dynamic compacted test set from MinTest. Column T gives the size of precomputed test set. With the exception of one case when proposed scheme has lower compression rates than VIHC (s35932), the table clearly shows that the proposed method leads to better compression ratios than previous approaches.

Finally a comparison of the previous approaches [4,13] and the proposed scheme is given in Table3. To the each scheme, the first column shows the required storage and the second column shows the power reduction. Note that the power consumption resulting from the test pattern generation is not included in the power computing list in the table (the same to [4] and [13]), since this part power consumption is relatively very small to the power.

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Data Storage</th>
<th>Test Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>NP</td>
<td>NC</td>
</tr>
<tr>
<td>s9234</td>
<td>160</td>
<td>8</td>
</tr>
<tr>
<td>s13207</td>
<td>236</td>
<td>8</td>
</tr>
<tr>
<td>s15850</td>
<td>126</td>
<td>8</td>
</tr>
<tr>
<td>s38417</td>
<td>100</td>
<td>8</td>
</tr>
<tr>
<td>s38584</td>
<td>137</td>
<td>8</td>
</tr>
</tbody>
</table>

![Table1. Experimental Result for Proposed Scheme](image)
resulting from the transitions. When compared to dual-LFSR reseeding proposed in [13], the proposed scheme is much more effective and up to 60% better power reduction (s13207). Because dual-LFSR reseeding scheme uses different test sets, the optimality of the ATPG and compaction procedures used to obtain the test sets strongly affects the results, and 1000 pseudo-random patterns are applied first. Even though all of this, our method needs less storage in s9234 and s38417. When compared to alternating run-length code in [4], the reduction of test power for the proposed scheme is not as much as the method proposed in [4] which is based on run-length encoding and there is still 20% gap, but the

5. Conclusions

LFSR reseeding is an attractive approach for compressing test data. We have shown that the proposed deterministic BIST scheme provides a way to reduce test power for LFSR reseeding. By utilizing Scan slices overlapping combined with LFSR reseeding, it can reduce transitions during test(up to 90%) and specified bits in test cubes. Experimental results indicate that the proposed method significantly reduces test power and even further reduces test storage. The proposed technique can be combined with other techniques, such as partial reseeding and seed encoding, and thus the better results can be obtained.

Reference