A New Pseudo-Exhaustive Test Method

Jhing-Fa Wang, Wei-Lun Wang, and Tzyy-Kuen Tien

Department of Electrical Engineering
National Cheng Kung University Tainan, Taiwan, R.O.C.

Abstract

An efficient and systematic pseudo-exhaustive test pattern generation algorithm has been proposed in this paper to solve the problem of test generation in BIST. Our algorithm has the following advantages: (1) it requires a minimum number of test signals for testing a circuit (2) it executes quickly with polynomial time complexity (3) it requires fewer test patterns and low hardware cost compared to five previous proposed methods.

I. Introduction

In built-in self-test (BIST) scheme, the flip-flops of circuit under test (CUT) can be controlled and observed by using the scan design methodology such as level sensitive scan design (LSSD) method etc., to reduce the testing complexity of CUT to the combinational circuits. In this paper, we assume the CUTs are purely combinational or the CUTs can be tested as combinational by means of a scan technique such as LSSD. Among the recent developments in VLSI testing, one of the most powerful testing methods is the exhaustive test method. In the exhaustive test method, an n input combinational network can be tested thoroughly by applying all $2^n$ distinct input combination signals and verifying that the correct output is obtained for each combination. However exhaustive test requires too much test time as the number of inputs is high (more than 20). With the lower complexity, the pseudo-exhaustive test retains almost all benefits of an exhaustive test.

Several pseudo-exhaustive test pattern generation methods, such as syndrome driver counter (SDC) [1], constant weight counter (CWC) [2], condensed linear feedback shift register (condensed LFSR) [3], the combination of LFSR and exclusive-or gate (LFSR/XOR) [4], and the other type of LFSR based on cyclic code (LFSR with cyclic code) [5] etc. For the same CUT, Table 1 shows their comparison of required test length and hardware cost. To obtain both short test length and small hardware cost simultaneously, we propose an efficient pseudo-exhaustive test method, Efficient Pseudo-Exhaustive Test (EPET) algorithm, in this paper.

<table>
<thead>
<tr>
<th>Test Technique</th>
<th>Hardware</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDC</td>
<td>low</td>
<td>high</td>
</tr>
<tr>
<td>CWC</td>
<td>high</td>
<td>lowest</td>
</tr>
<tr>
<td>Condensed LFSR</td>
<td>low</td>
<td>medium</td>
</tr>
<tr>
<td>LFSR/XOR</td>
<td>high</td>
<td>low</td>
</tr>
<tr>
<td>Cyclic LFSR</td>
<td>low</td>
<td>low</td>
</tr>
<tr>
<td>EPET</td>
<td>lowest</td>
<td>lowest</td>
</tr>
</tbody>
</table>

II. Notation and Terminology

Before describing the concept of EPET algorithm, some terminologies and definitions are introduced as follows:

It is assumed that the test patterns to a CUT are from the test signals (which are constructed by a counter or LFSR) or the linear combination of any pair of test signals.

Definition 1 [2]: Let $D(N)$ be defined as a dependency matrix with $m$ rows (primary outputs) by $n$ columns (primary inputs) and it shows the structure connection relationship between the primary inputs and outputs of the combinational network N. Each row and column represent one of the primary outputs and
inputs, respectively. The value of each entry is either 1 or 0. An entry is 1 if and only if the corresponding input is one Boolean variable of the combinational logic function of the corresponding output.

We use Figure 1 as an example to illustrate this definition and other terminologies there after. In Figure 1, its dependency matrix \( D(N) \) is shown as follows:

\[
\begin{align*}
X_1 & \quad X_2 & \quad X_3 & \quad X_4 & \quad X_5 & \quad X_6 & \quad X_7 & \quad X_8 \\
\hline
f_1 & \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \\
\hline
f_2 & \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \\
\hline
f_3 & \begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix} \\
\hline
f_4 & \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \end{bmatrix} \\
\hline
f_5 & \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}
\end{align*}
\]

Since a primary output \( f_1 \) is the function of \( X_1, X_2, X_3, \) and \( X_4 \), the entry value of the first row in \( D(N) \) is 11110000.

**Definition 2 [6]:** Let \( P(N) \) be defined as an assignment matrix with \( n \) rows by \( q \) columns, where \( q \) is the test signal number. The value of each entry is either 1 or 0. Initially, the test signal number equals the maximum row weight of \( D(N) \) (i.e., the maximum value of the sum of nonzero entry of each row). An entry \( P(i,j) \) is 1 if and only if the \( i \)-th primary input is assigned to the \( j \)-th test signal (i.e., the \( j \)-th test signal is connected to the \( i \)-th primary input such that the test signal will not produce the dependent function in the primary output of \( N \)).

There may be 1 or 2 nonzero entries with the same row. If there exist 2 nonzero entries for the \( i \)-th row, this implies that the \( i \)-th primary input is assigned to a linear combination of one pair of test signals.

In Figure 1, its assignment matrix \( P(N) \) is shown in the following:

\[
P(N) =
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

In the 6th row of \( P(N) \), it shows that the 6th primary input \( X_6 \) is assigned to a linear combination of one pair of test signals \( t_2 \) and \( t_4 \).

**Definition 3:** Let \( R \) be defined as a vector with \( n \) components and its component \( T(i) \) shows the potential assignment number that the unassigned \( i \)-th primary input can be assigned to the test signals.

The value of each component is 0 or any other positive integer which is less than or equal to the number of test signals. If the \( i \)-th primary input has already been assigned or it can not be assigned to any existed test signal, the value of \( T(i) \) must be set to 0.

After the four primary inputs, \( X_1, X_2, X_3, \) and \( X_4 \) have already been assigned, we regard \( X_5 \) as a dependent function of \( X_3 \) and \( X_4 \) and an independent function of \( X_1 \) and \( X_2 \) by \( D(N) \), hence \( X_5 \) can be assigned to the same test signal with \( X_1 \)'s test signal (i.e., \( t_1 \)) or \( X_2 \)'s (i.e., \( t_2 \)) and the value of \( T(5) \) is 2, which is shown in the following:

\[
T(1) T(2) T(3) T(4) T(5) T(6) T(7) T(8)
\begin{bmatrix}
0 & 0 & 0 & 0 & 2 & 2 & 1 & 2
\end{bmatrix}
\]

In summary, the combination of two vectors, \( R \) and \( T \), indicates three operations of the assignment between the primary inputs and test signals and these operations are shown as follows:

<table>
<thead>
<tr>
<th>( R(i) )</th>
<th>( T(1) )</th>
<th>( T(2) )</th>
<th>( T(3) )</th>
<th>( T(4) )</th>
<th>( T(5) )</th>
<th>( T(6) )</th>
<th>( T(7) )</th>
<th>( T(8) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Comments:

(1) The \( i \)-th primary input \( X_i \) has already been assigned to a appropriate test signal.
(2) The i-th primary input Xi has not been found out its appropriate test signal so far and the positive integer k (which is less than or equal to the number of test signals) shows the potential assignment number that the unassigned input can be assigned to the appropriate test signal. If there are more than one primary input as above mentioned then take the assignment of the primary input with the least value of k.

(3) The i-th primary input Xi can not be assigned to any existed test signal. For this case, the input is assigned to a linear combination of two appropriate test signals or a new test signal.

Definition 5: Let W be defined as a vector with n components and its components W(i) shows the number of ones in the i-th column (primary input) of D(N).

In Figure 1, each component value of W is shown as follows:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

III. The Design Procedure of Efficient Pseudo-Exhaustive Test (EPET) Algorithm

Based on the assumption that any outputs does not depend on all inputs, an n-input, m-output combinational network N is considered.

EPET Algorithm:

Step 1 (Construct a dependency matrix):
Input the entries of a dependency matrix D(N).

Step 2 (Find out the maximum row weight of D(N) to determine the test signal number):
Determine the maximim weight row of D(N) (i.e., one row of the maximum number of nonzero entry in each row).

Step 3 (Construct one matrix P(N), and three vectors, R, T, and W):
Construct an assignment matrix P(N) with n rows by q columns (this equals the number of test signal, initially). Construct three n-tuple vectors R, T, and W.

Step 4 (Find out a unassigned primary input):
Find the i-th primary input, if R(i) is 0 and T(i) is minimum (but not equals 0). If there exists more than one primary inputs as above mentioned, then select the least numbered primary input.

Step 5 (Take the assignment to a unassigned primary input):
For the unassigned primary input Xi, the value of T(i) shows the potential assignment number that the unassigned input Xi can be assigned to the appropriate test signal. If T(i) equals 1 then the unassigned input Xi can be assigned to its unique test signal, else (i.e., T(i) is larger than 1) assign the test signal of the smallest order (where the unassigned input Xi can be assigned) to the i-th primary input.

Step 6 (Update P(N), R and T):
Update the assignment matrix P(N), vector R and vector T, respectively. If a primary input input Xi is assigned to a test signal tj then the entry P(i,j) is set to 1, R(i) is set to 1, and T(i) is set to 0. The other components of R for the unassigned input are all set to 0. The other components of T for the unassigned inputs are set to a positive integer (which is less than or equal to the number of test signals), respectively, by the independence between the unassigned inputs and the assigned inputs.

Step 7 (Determine whether all the unassigned primary inputs have the nonzero potential assignment number):
Check all the components of T, if any component of T is nonzero, go to Step 4.

Step 8 (On the condition that all the components of T are 0, determine whether all the unassigned primary inputs have been assigned):
Check all the components of R, if all components of R are equal to 1 (this implies that all the primary inputs have already been assigned), then go to Step 15, else continue.

Step 9 (For a unassigned primary input Xi with R(i) = 0 and T(i) = 0, assign Xi to a linear combination of two appropriate test signals or a new test signal):
Find the i-th primary input, if R(i) equals 0 and W(i) is maximum. If there exists more than one primary inputs as above mentioned then select the least numbered appropriate primary input.

Step 10 (Determine whether Xi can be assigned to a linear combination of two appropriate test signals):
Check whether the i-th primary input can be assigned to the linear combination of any pair of test signals such that this pair of test signals are independent of their linear combination for observing every primary output of the combinational network N. If such one pair of test signals tj and tr are found, then assign their
linear combination to the i-th primary input and update both P and R, i.e., P(i,j), P(i,r), and R(i) are set to 1. Whatever such one pair of test signals has been found or not, continue next step.

**Step 11 (Check whether all unassigned primary inputs have been tried to do the assignment):**
Go to Step 9, until all unassigned primary inputs have been tried to do the assignment of a linear combination of two appropriate test signals.

**Step 12:**
If all the components of R are equal to 1 (this implies that all the primary inputs have been assigned), go to Step 15.

**Step 13 (Assign the unassigned primary inputs to a new test signal):**
If R(i) equals 0 and W(i) is minimum, then assign a new test signal t_u (where u = q+1) to the unassigned i-th primary input. Increase the value of q by 1 (i.e., the number of test signal is increased by 1) and update P, R (i.e., P(i,u) and R(i) are set to 1).

**Step 14:**
Check all the components of R, if any component of R equals 0 (this means that a primary input has not yet been assigned) then go to Step 4.

**Step 15:**
Print out the entry values of P to build a test pattern generator and stop this algorithm.

To demonstrate the computational efficiency of the EPET algorithm, several circuits, C880, C2670, C5315, and C7552, have been proposed as benchmarks for this algorithm and these are shown in Table 2. The results of applying the EPET algorithm to four non-MTC circuit examples are shown in Figures 1-4. Table 3 compares the required test length for these four examples using this algorithm with the required test length using five previously proposed test generation methods, syndrome driver counter (SDC) [1], constant weight counter (CWC) [2], condensed linear feedback shift register (condensed LFSR) [3], the combination of LFSR and exclusive-or gate (LFSR/XOR) [4], and a LFSR based on cyclic code (LFSR with cyclic code) [5].
From Table 3, it can be seen that the EPET algorithm requires fewer test patterns in nearly all cases. The only case that the required test patterns of EPET algorithm is higher than the required test patterns of CWC technique applied to Example 4 by 1. However, the CWC method has the disadvantage of requiring a large hardware cost for generating test patterns.

IV. Summary

An efficient algorithm for designing a pseudo-exhaustive test pattern generator where none of primary outputs depends on all of the primary inputs has been presented in this paper. Our algorithm has the following advantages: (1) it requires a minimum number of test signals for testing a circuit (2) it executes quickly with polynomial time complexity (3) it requires fewer test patterns and low hardware cost compared to five previous proposed methods.

REFERENCES