TEST THE BENCHMARK CIRCUIT BY USING BUILT-IN SELF TEST AND TEST PATTERN GENERATION IN FPGA TECHNOLOGY

B. Hariprasad
PG Scholar in VLSI System design,

A. Chandra Babu
M.Tech, Asst. Professor, ECE Department,

1hariprasadsret@gmail.com,

chandraece.413@gmail.com.

ABSTRACT:

In the proposed method we are test the S27 sequential circuit by using Built in Self Test. This paper describes an on-chip test generation method for functional broadside tests. The hardware was base on the application of primary input sequences initial from a well-known reachable state, therefore using the circuit to produce additional reachable states. Random primary enter sequences were changed to avoid repeated synchronization and thus defer varied sets of reachable states. Functional broadside tests are two-pattern scan based tests that avoid over testing by ensuring that a circuit traverses only reachable states in the functional clock cycles of a check. These consist of the input vectors and the equivalent responses. They check for proper operation of a verified design by testing the internal chip nodes. Useful tests cover a very high percentage of modeled faults in logic circuits and their generation is the main topic of this method. Often, functional vectors are understood as verification vectors, these are used to verify whether the hardware actually matches its specification. Though, in the ATE world, any one vectors applied are understood to be functional fault coverage vectors applied during developing test. This paper show the on chip test Generation for a bench mark circuit using simple fixed hardware design with small no of parameters altered in the design for the generation of no of patterns.

Key words: error correcting codes, hamming distance, data comparison, parity matrix.

I. INTRODUCTION

Testing a circuit is necessary to make sure that all design errors have been fixed. Over testing occurs due to nonfunctional operation conditions created by unreachable scan in states. Tests applied under nonfunctional operation conditions, which are made possible by scanning in an unreachable state, may lead to unnecessary yield loss. Slow paths that cannot be sensitized may cause circuit to fail and when current demands higher than possible cause voltage drop leading to the circuit failure.

Functional broadside tests allow only reachable states as scan-in states. As broadside tests are two pattern tests, the circuit undergoes state transitions that are possible during functional operations. Delay faults can effect functional operation and current demands don’t exceed the limit. This avoids over testing. Power dissipation also don’t exceed that possible. Functional and pseudo functional scan based tests compute reachable states offline. Pseudo functional tests use functional constraints
to avoid arbitrary states. These tests are not sufficient for avoiding unreachable states.

Over testing due to the application of two-patterns scan-based tests was described in [2]–[4]. Slow paths that cannot be sensitized during functional operation may cause the circuit to fail [2]. In addition, current demands that are higher than those possible during functional operation may cause voltage drops that will slow the circuit and cause it to fail [3], [4].

In both cases, the circuit will operate correctly during functional operation. Functional broadside tests [5] ensure that the scan-in state is a state that the circuit can enter during functional operation, or a reachable state. As broadside tests [6], they operate the circuit in functional mode for two clock cycles after an initial state is scanned in. Four primary parameters must be considered in developing a BIST methodology for embedded systems; these correspond with the design parameters for on-line testing techniques discussed.

- Fault coverage: This is the fraction of faults of interest that can be exposed by the test patterns produced by pattern generator and detected by output response monitor. In presence of input bit stream errors there is a chance that the computed signature matches the golden signature, and the circuit is reported as fault free. This undesirable property is called masking or aliasing.
- Test set size: This is the number of test patterns produced by the test generator, and is closely linked to fault coverage: generally, large test set simply high fault coverage.
- Hardware overhead: The extra hardware required for BIST is considered to be overhead. In most embedded systems, high hardware overhead is not acceptable.
- Performance overhead: This refers to the impact of BIST hardware on normal circuit performance such as its worst-case (critical) path delays. Overhead of this type is sometimes more important than hardware overhead.

II. BLOCK DIAGRAM OF BIST

Figure 1 Block diagram of BIST

BIST can be used for non-concurrent, on-line testing of the logic and memory parts of a system. It can readily be configured for event-triggered testing, in which case, the BIST control can be tied to the system reset so that testing occurs during system start-up or shut down. BIST can also be designed for periodic testing with low fault latency. This requires in incorporating a testing process into the CUT that guarantees the detection of all target faults within a fixed time. On-line BIST is usually implemented with the twin goals of complete fault coverage and low fault latency. Hence, the test generation (TG) and response monitor (RM) are generally designed to guarantee coverage of specific fault models, minimum hardware overhead, and reasonable set size. These goals are met by different techniques in different parts of the system. TG and RM are often implemented by easy, counterlike circuits, especially linear-feedback shift registers (LFSRs). The LFSR is simply a shift register formed from standard flip-flops, with the outputs of selected flip-flops being fed back to the shift register’s inputs. When used as a TG, an LFSR is set to cycle rapidly through a large number of its states. These states, whose choice and order depend on the design parameters of the LFSR, define the test patterns. In this mode of operation, an LFSR is seen as a source of (pseudo) random tests that are, in principle, applicable to any fault and circuit types.

LFSR DESIGN

The hardware used for generating a primary input sequence consists of a Linear Feedback Shift Register and a small number of gates. Gates are used to modify the random sequence in order to avoid repeated synchronization that is the sequence takes the circuit to repeat the same or similar states. In addition to this, a single gate is used for determining the tests to
be applied based on primary input sequence.

FIG 2. On-chip generation of A

The fixed hardware structure needs to be tailored to the given circuit through these parameters.
1) The number of LFSR bits.
2) Seeds for the LFSR to generate different primary input sequences and several subsets of tests.
3) The length of primary input sequence.
4) The specific gates used to modify LFSR sequence into primary input sequence A.
5) Specific gates for selecting the type of functional broadside tests that will be applied to the circuit based on primary input sequence A.

TABLE 1

<table>
<thead>
<tr>
<th>u</th>
<th>LFSR(u)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>101</td>
</tr>
<tr>
<td>1</td>
<td>010</td>
</tr>
<tr>
<td>2</td>
<td>001</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
</tr>
<tr>
<td>4</td>
<td>010</td>
</tr>
<tr>
<td>5</td>
<td>001</td>
</tr>
<tr>
<td>6</td>
<td>100</td>
</tr>
<tr>
<td>7</td>
<td>110</td>
</tr>
<tr>
<td>8</td>
<td>011</td>
</tr>
<tr>
<td>9</td>
<td>001</td>
</tr>
<tr>
<td>10</td>
<td>100</td>
</tr>
<tr>
<td>11</td>
<td>010</td>
</tr>
<tr>
<td>12</td>
<td>101</td>
</tr>
<tr>
<td>13</td>
<td>011</td>
</tr>
<tr>
<td>14</td>
<td>101</td>
</tr>
<tr>
<td>15</td>
<td>010</td>
</tr>
</tbody>
</table>

The on-chip test generation hardware is based on the one described in [7]. It differs from this paper hardware in following ways.
1) In [7], a circuit with n primary inputs and a parameter mod, the LFSR used for producing A has n + mod bits. n left most bits are used for driving the primary input sequence and mod right most bits are used for modifying random sequence in order to avoid repeated synchronization. The same mod right most bits are used for modifying all primary inputs and this may lead to modifying the primary inputs to same values and some primary inputs receive shifted or by initializing sequence. S(u) is state of circuit at time u for.
2) In [7] for selecting the tests that applied to the circuit based on A, a large multiplexer and a significant number of gate is used. In this paper a single gate is used for selecting the tests by fixing the gate in advance and make sure that all primary inputs fit with the preselected gate.
3) In [7] the lengths of primary inputs are varied to control the number of tests. In our design primary inputs are of uniform length, this make the application process uniform.

This paper design is independent of the number of sequences used. Sequences differ only in seeds of the LFSR. The seeds can be stored on-chip, or a seed can be scanned in together with the initial state of the circuit before the application of every primary input sequence. This paper solely focuses on application on input sequence, so for output testing an output compactor such as Multiple Input Shift Register (MISR) is assumed.

If a circuit under test is embedded in a larger design, the primary inputs may drive other logic blocks in same design and addition with external inputs, they may drive circuit under test. The primary outputs may drive some logic blocks or primary output is the output of the design. To avoid these complexities we assume that the primary sequence is of any combination of values. Functional constraints on primary input sequence are accommodated in following ways.
1) The logic used for generating primary input sequence is extended to impose some functional constraints.
2) A separate logic block is used for modifying A to satisfy functional constraints.
3) Placing the on-chip test generation hardware for a logic block on the inputs of the logic blocks driving it can create some of the functional constraints for the block.

III. Proposed Method for On-Chip Generation of Functional Broadside Tests

In this paper the circuit is initialized to a known state before functional operation is assumed. Initialization is achieved by applying a synchronizing sequence or by hardware reset or by combination of both. The initial state is denoted by Sr. As Sr is
the initial state of functional operation, it is a reachable state. In the set of reachable states produced by Sr consists of every state Si, there exist a primary input sequence that takes the circuit from Sr to Si. Since Si is generated from Sr, Si is a reachable state.

On-chip generation for reachable states is also done by using Sr and primary input sequence A= a(0)a(1)......a(L-1) of length L. Circuit is initialized using scan in operation. 0≤u≤L and S(0)= Sr.

Every state S(u) can be used as initial state for functional broadside test (S(u), a1, a2) where S(u) is a scan in state and a1 and a2 are primary input vectors applied to S(u) using a slow and fast clocks respectively. Every sequence of length two of A defines a functional broadside test, t(u)=(S(u),a(u),a(u+1)). By using these vectors a different source for producing primary vectors is avoided.

For producing primary input sequence ISCAS 89 S27 benchmark circuit is used. For initial state Sr =000 Primary input sequence is obtained. For every time unit u there is S(u) and primary input vector is generated.

Table 1 shows S(u) and a(u) for every u. Functional broadside tests t(0)=(000,1001,1110),t(1) =(010,1110,0010)……………….t(14) =(101,1111,1110).

For detecting faults circuit is placed on initial state and tests are generated using primary input sequence.

The faults are detected in following ways

1) An output vector z(u+1) in response to s(u+1) is calculated and if it is different from the expected output vector.

2) If final state s(u+2) is different from expected fault free vector. z(u+1) and s(u+2) is captured by output compactor, MISR. But the tests t(u) and t(u+1) may overlap because for t(u) s(u+1) is generated and for t(u+1) also s(u+1) is generated. For application of both tests we need special hardware. Thus to avoid the hardware we use non overlapping tests such that in the form {t(u0),t(u1),……..t(uk-1)} where u0+1<u+1and 0≤i<k-1.

The source for generating primary input sequence is LFSR. But random primary input sequence may limit the number of reachable states. A cube is considered to avoid repeated synchronization. A cube c synchronizes subset of state variables S(c) in some conditions. Let be applied to the primary inputs when the circuit is in the all unspecified present-state. Suppose that this results in a next-state s. The state variables whose values are specified in s are included in S(c).In the example of shown in Fig 2, the primary input cube I0I1I2I3=0XXX applied in present-state y0y1y2= XXX results in the next-state Y0Y1Y2=0XX, synchronizing state variable y 0. In addition, the primary input cube I0I1I2I3=XX1X applied in present-state y0y1y2= XXX results in the next-state Y0Y1Y2=XX0, synchronizing state variable y2.

A primary cube c with small specified values prevents the circuit from entering various reachable
states. Single cubes are considered to avoid repeated synchronization. For considering single cube using a software procedure by computing primary input cubes that synchronize one or more state variables. For a circuit with primary inputs, I0, I1, I2 …IN-1, the procedure considers every primary input cube Cj,v where primary input Ij assumes the value and the other primary inputs are unspecified. For Cj,v, where 0≤j<n and v {0,1} the procedure computes the set of synchronized state variables s(cj,v). It then combines all the primary input cubes into a single cube that is useful for avoiding repeated synchronization as follows.

Let F be the target faults and test set T(U)={t(ui) : uiU}, and the conditions required for test selection are:
1) To ensure that tests are non-overlapping.
2) It is possible to produce U on on-chip.
3) Test should detect as many as possible faults.
4) U should be as small as possible.

A counter CNT is used for test selection. Counter sequence is CNT= cnt(0)cnt(1)…….cnt(M-1) where M=log2L and sel=2m.

IV. RESULTS

The functional broadside tests for various benchmark circuits are evaluated.

<table>
<thead>
<tr>
<th>circuit</th>
<th>mod</th>
<th>d</th>
<th>L</th>
<th>seq</th>
<th>eff</th>
<th>f.c.</th>
<th>determ</th>
</tr>
</thead>
<tbody>
<tr>
<td>s644</td>
<td>4</td>
<td>4</td>
<td>1024</td>
<td>5</td>
<td>824</td>
<td>86.51</td>
<td>86.51</td>
</tr>
<tr>
<td>s1423</td>
<td>3</td>
<td>4</td>
<td>1024</td>
<td>43</td>
<td>1016</td>
<td>86.27</td>
<td>93.33</td>
</tr>
<tr>
<td>s3578</td>
<td>3</td>
<td>4</td>
<td>1024</td>
<td>30</td>
<td>1003</td>
<td>75.15</td>
<td>79.06</td>
</tr>
<tr>
<td>s3532</td>
<td>3</td>
<td>4</td>
<td>1024</td>
<td>2</td>
<td>599</td>
<td>89.78</td>
<td>89.78</td>
</tr>
<tr>
<td>s38584</td>
<td>3</td>
<td>4</td>
<td>1024</td>
<td>91</td>
<td>1023</td>
<td>16.49</td>
<td>-</td>
</tr>
</tbody>
</table>

From above table it is known that the proposed method of functional broadside tests achieve high fault coverage than functional test sequences.

CONCLUSION

A fixed and simple hardware structure is implemented to conduct functional broadside tests on-chip. The hardware is implemented using the primary input sequence to a circuit with known reachable state to achieve additional reachable states. Random sequences are chosen to avoid repeated synchronization. Two pattern tests are done to achieve higher fault coverage. The parameters that are tailored the circuit under test to conduct functional broadside tests on-chip are length of LFSR, seeds for LFSR, length of primary sequence, gates for modifying the sequence and gates for selecting tests based on primary input sequence.

REFERENCES


