A Novel Built-in Self Test Algorithm for functional Broadside tests

G. Kiran Kumar\(^1\)  R. Suresh Kumar\(^2\)  B. Poornaiah\(^3\)

\(^1\)PG Student (M.Tech), \(^2\)Assistant Professor, \(^3\)H.O.D & Associate Professor,
Dept. Of ECE, Universal College of Engineering & Technology, Guntur

Abstract— In this paper, we propose a method for on-chip generation of weighted test sequences for synchronous sequential circuits. For combinational circuits, three weights, 0, 0.5 and 1, are sufficient to achieve complete coverage of stuck-at faults, since these weights are sufficient to reproduce any specific test pattern. For sequential circuits, the weights we use are defined based on subsequences of a deterministic test sequence. Such weights allow us to reproduce parts of the test sequence, and help ensure that complete fault coverage would be obtained by the weighted test sequences generated. This accumulator-based 3-weight test pattern generation scheme is presented copes with the inherent drawbacks of the scheme proposed more precisely. The weighted random test pattern generation represents a significant departure from classical methods of generating test sequences for complex large scale integration packages. The virtue of this technique is its simplicity and the fact that test-generation time is virtually independent of or gates in the logic package to be tested. This technique can be used both in a conventional tester and in a tester where the weighted random test pattern generation is implemented in hardware.

Keywords— Functional tests, Broadside test, Built in Self Test.

I. INTRODUCTION

Very-large-scale integration (VLSI) is the process of creating integrated circuits by combining thousands of transistor-based circuits into a single chip. VLSI began in the 1970s when complex semiconductor and communication technologies were being developed. The microprocessor is a VLSI device. The term is no longer as common as it once was, as chips have increased in complexity into the hundreds of millions of transistors.

As of early 2008, billion-transistor processors are commercially available, an example of which is Intel's Montecito Itanium chip. This is expected to become more commonplace as semiconductor fabrication moves from the current generation of 65 nm processes to the next 45 nm generations (while experiencing new challenges such as increased variation across process corners). Another notable example is NVIDIA’s 280 series GPU.

This microprocessor is unique in the fact that its 1.4 Billion transistor count, capable of a teraflop of performance, is almost entirely dedicated to logic (Itanium’s transistor count is largely due to the 24MB L3 cache). Current designs, as opposed to the earliest devices, use extensive design automation and automated logic synthesis to lay out the transistors, enabling higher levels of complexity in the resulting logic functionality.

This method is used for on-chip generation of weighted test sequences for synchronous sequential circuits. The weights we used were defined based on subsequences of a deterministic test sequence. The use of a deterministic test sequence to define the weights allowed us to reproduce parts of the test sequence, and helped ensure that complete fault coverage would be obtained. It described a procedure for defining a set of weights from which weight assignments can be constructed, a procedure for selecting weight assignments so as to detect target faults, and presented experimental results to demonstrate that complete fault coverage can be achieved by this method. It also investigated the tradeoffs between the number of weight assignments and the number of observation points required to achieve complete fault coverage. The use of pure-random sequences as part of the weight scheme, followed by the synthesis of the on-chip test generation hardware, is the subject of future work.

In this paper we describe a method for on-chip generation of weighted test sequences for synchronous sequential circuits. For combinational circuits, three weights, 0, 0.5 and 1, are sufficient to achieve complete
coverage of stuck-at faults, since these weights are sufficient to reproduce any specific test pattern. For sequential circuits, the weights we use are defined based on subsequences of a deterministic test sequence. Such weights allow us to reproduce parts of the test sequence, and help ensure that complete fault coverage would be obtained by the weighted test sequences generated. BIST is a design-for-testability technique that places the testing functions physically with the circuit under test (CUT). The basic BIST architecture requires the addition of three hardware blocks to a digital circuit: a test pattern generator, a response analyzer, and a test controller. The test pattern generator generates the test patterns for the CUT.

Some of pattern generators are a ROM with stored patterns, a counter, and a linear feedback shift register (LFSR). A typical response analyser is a comparator with stored responses or an LFSR used as a signature analyser. It compacts and analyses the test responses to determine correctness of the CUT. A test control block is necessary to activate the test and analyze the responses. However, in general, several test-related functions can be executed through a test controller circuit. A technique for weighted pseudo-random built-in-self-test of VLSI circuits is proposed, which uses special scan cells and a new weight selection algorithm to achieve low power dissipation. Certain types of circuit faults are undetectable using the correlated bit streams produced by the conventional linear-feedback-shift-register. Weighted sets comprising three weights, namely 0, 1, and 0.5 have been successfully utilized so far for test pattern generation, since they result in both low testing time and low consumed power.

Since accumulators are commonly found in current VLSI chips, this scheme can be efficiently utilized to drive down the hardware of BIST pattern generation. Proposed system is advantageous as it can be implemented using any adder design and it don’t require any modification of the adder. Also the operating speed of the adder is same.

II. Weighted Generation BIST

The utilization of accumulator modules for output data compaction in symmetric transparent BIST for RAMs is proposed. It is widely accepted by the test community that the utilization of modules that typically exist in the circuit, e.g., accumulators or arithmetic logic units, for BIST test pattern generation and/or response verification possesses advantages. It is shown that in this way the hardware overhead, the complexity of the controller, and the aliasing probability are considerably reduced.

Lower hardware overhead and elimination of the need for multiplexers in the circuit path; furthermore, the modules are exercised. Therefore, faults existing in them can be discovered. The comparison will be performed with respect to the hardware overhead and no elimination of the need for multiplexers in the circuit path.

Test-pattern generators are based on arithmetic operations, are becoming cost-effective built-in self-test solutions for circuits with embedded processors. Similar to pseudorandom Test-pattern generators, arithmetic Test-pattern generators use reseeding to reach high levels of fault Coverage.

Which improves the computation time and the optimization includes the seed and the increment of the arithmetic test pattern generation. Which is suitable for scan-path structures, the problem of embedding the test vectors is not considered. A methodology to generate a reseeding test for build in self test based on accumulated pattern generation is presented. This methodology required the generator to run the test.

A digital system is tested and diagnosed during its lifetime on numerous occasions. Such a test and diagnosis should be quick and have very high fault coverage. One way to ensure this is to specify such a testing to as one of the system functions, so now it is called Built In Self Test (BIST). With properly designed BIST, the cost of added test hardware will be more than balanced by the benefits in terms of reliability and reduced maintenance cost.

For BIST, we would require that the test patterns be generated on the system/chip itself. However, this should be done keeping in mind that the additional hardware is minimized. One extreme is to use exhaustive testing using a counter and storing the results for each fault simulation at a place on the chip (like ROM). An n input circuit would then require 2^n combinations which can be very tiresome.
III. ACCUMULATOR BASED 3-WEIGHT PATTERN GENERATION

A new weighted random pattern design for testability is described where the shift register latches distributed throughout the chip are modified so that they can generate biased pseudo-random patterns upon demand. A two-bit code is transmitted to each weighted random pattern shift register latches to determine its specific weight. The weighted random pattern test is then divided into groups, where each group is activated with a different set of weights. The weights are dynamically adjusted during the course of the test to "go after" the remaining untested faults.

An accumulator-based 3-weight test pattern generation scheme is presented; the proposed scheme generates set of patterns with weights 0, 0.5, and 1. Since accumulators are commonly found in current VLSI chips, this scheme can be efficiently utilized to drive down the hardware of built in self test pattern generation, as well. Comparisons with previously presented schemes indicate that the proposed scheme compares favourably with respect to the required hardware. Generally, the accumulator-based compaction technique uses an accumulator to generate a composite fault signature for a circuit under test. The error coverage for this method has been previously analyzed. We describe an alternative technique for calculating the error coverage of accumulator-based compaction using the asymmetric error model. This technique relies on the central limit theorem of statistics and can be applied to other count-based compaction schemes. The data paths of most contemporary general and special purpose processors include registers, adders and other arithmetic circuits. If these circuits are also used for built-in self-test, the extra area required for embedding testing structures can be cut down efficiently. Several schemes based on accumulators, subtracters, multipliers and shift, resistors have been proposed and analysed in the past for parallel test response compaction, whereas some efforts have also been devoted in the bit-serial response compaction case. The utilization of accumulators for time compaction of the responses in built-in self test environments has been studied by various researchers. One of the well-known problems of time compactors is aliasing, i.e. the event that a series of responses containing errors result in a signature equal to that of an error-free response sequence. In this paper we propose a scheme to reduce aliasing in accumulator based compaction environments. With the proposed scheme, the aliasing probability tends to zero, as the number of the patterns of the test set increases.

In general, pseudo random pattern generation requires more patterns than completely deterministic Automatic Test Pattern Generation (ATPG), but obviously, fewer than the exhaustive testing. However, it was found that the stuck-fault coverage rises in a logarithmic fashion towards hundred percentage, but at the cost of enormous numbers of random patterns. On top of it, certain circuits are random pattern resistant circuits in that they do not approach full fault coverage with an unbiased random pattern. Such circuits require extensive insertion of testability hardware or a modification of random pattern generation to ‘weighted pseudo random pattern generation’ in order to obtain an acceptable fault percentage. This desire to achieve higher fault coverage with shorter test lengths and therefore shorter test times led to the invention of the weighted pseudo random pattern generator.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>C_out = C_in</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>C_out = C_in</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>C_out = C_in</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Fig 1: TRUTH TABLE FOR FULL ADDER

The main object of the weighted pattern generation is an accumulator cell. To implement the accumulator in the proposed weighted pattern generation scheme is based on presented in Figure.
In the above figure, we assume that the set and reset are active high signals and at the same time the set and reset are used to without loss of generality. And at the time, the respective cell of another register B[i] is also occurred. For this accumulator cell, one out of three configurations can be utilized, as shown in Fig.

The LFSR are the basic building blocks of the pseudo random test pattern generators. In unbiased pseudo random testing, the outputs from the LFSR is fed directly to the CUT and thus the no. of LFSR stages required is equal to the number of inputs to the CUT. For a weighted pseudo random testing we however require much more LFSR stages than the inputs to the CUT. This is so because each weighted bit usually requires more than one equi-probable bit coming in from an LFSR stage for the generation of its weighted bit.

Now we assume that for both the unbiased and the weighted case we have the total number of LFSR shift registers required for each of the CUTs.

In the last section we see how, depending upon the no. of inputs of the CUT and the associated probabilities we can make an LFSR configuration for both, the unbiased and the weighted pseudo random testing part. We do this by writing a Verilog file with the entire configuration written in it. After that we run the Verilog simulator and get the raw patterns in a file. This reads the patterns generated from an LFSR working for weighted pseudo random testing.

The general configuration of the proposed scheme is presented. The Logic module provides the Set [n-1:0] and Reset [n-1:0] signals that drive the S and R inputs of the Register A and Register B inputs. Note that the signals that drive the S inputs of the flip-flops of Register A, also drive the R inputs of the flip-flops of Register B and vice versa.

The intersection of different values yields an unspecified value (\(x\)), which is translated into a weight of 0.5. The intersection of test subsequence’s yielded weight assignments that were used in a similar way to the ones for combinational circuits. However, for sequential circuits, the intersection of a subset of test subsequence’s of length \(M\) results in \(M\) weight assignment s that have to be used consecutively, and changed at every time unit. The need to change the weight assignment at every time unit is undesirable.

In addition, this method is not applicable when a single test sequence is given for the circuit. It can be equally well applied with both implementations. Therefore, the comparison will be performed with respect to the hardware overhead and the impact on the timing characteristics of the adder of the accumulator. Both schemes require a session counter in order to alter among the different weight sessions; the session counter consists of \(\log_2 K\) bits, where \(K\) is the number of test sessions of the weighted test set. In the 3-weight pattern generation scheme proposed the scan chain is driven by the output of a linear feedback shift register (LFSR).
Logic is inserted between the scan chain and the CUT inputs to fix the outputs to the required weight (0, 0.5, or 1). In order to implement the scheme [5], a scan structure is assumed. Furthermore, an LFSR required to feed the pseudorandom inputs to the scan inputs is implemented; the number of LFSR stages is \( \log_2 n \), where \( n \) is the number of scan cells, as well as a scan counter, common to all scan schemes.

Comparisons with a previously proposed accumulator-based 3-weight pattern generation technique, the hardware overhead of the proposed scheme is lower, while at the same time no redesign of the accumulator is imposed, thus resulting in reduction in test application time.

**IV. RESULTS AND CONCLUSIONS**

Finally, we have presented an accumulator-based 3-weight (0, 0.5, and 1) test-per-clock generation scheme, which can be utilized to efficiently generate weighted patterns without altering the structure of the adder. Comparisons with a previously proposed accumulator-based 3-weight pattern generation technique and it indicates that the hardware overhead of the proposed scheme is lower, while at the same time no redesign of the accumulator is imposed, thus resulting in reduction in test application time. Comparisons with scan based schemes show that the proposed schemes results in lower hardware overhead. Finally, comparisons with the accumulator-based scheme proposed and reveal that the proposed scheme results in significant decrease in hardware overhead.

Comparisons with a previously proposed accumulator-based 3-weight pattern generation technique and it indicates that the hardware overhead of the proposed scheme is lower, while at the same time no redesign of the accumulator is imposed, thus resulting in reduction in test application time. Comparisons with scan based schemes show that the proposed schemes results in lower hardware overhead. Finally, comparisons with the accumulator-based scheme proposed and reveal that the proposed scheme results in significant decrease in hardware overhead.
REFERENCES


Authors Profiles

Kiran Kumar Gopathoti is pursuing his Master degree M.Tech in VLSI Design in UNIVERSAL COLLEGE OF ENGINEERING & TECHNOLOGY, GUNTUR. He completed his Graduation from Chundi Ranganayakulu College of Engineering, Guntur. His Field of Interest includes CAD based VLSI systems.

R. Suresh Kumar is working as an Assistant Professor in Universal College of Engineering & Technology, Guntur. He Completed his Post Graduation
Poornaiah Billa is a Part time Research Scholar in Instrument Technology Department, Andhra University College of Engineering (A), Andhra University, Visakhapatnam, India. He received his M.Tech. in Electronics and Communication Engineering from Andhra University, Visakhapatnam in 2002 and B.Tech in Electronics and Instrumentation Engineering from Nagarjuna University, Guntur in 1998. At present, he is working as Associate Professor and Head in the Electronics and communication Engineering in Universal college of Engineering and Technology, Dokiparru, Guntur, A.P. He has 11 Years of teaching experience in various institutes.