A Novel Approach for BIST Pattern Generation

[1]P. Sivarami Reddy, *PG Student [VLSI] , Dept. of ECE, Dhruva Institute of Engineering and Technology, Hyderabad,A.P,India*

[2]B. Manasa, *Assistant Professor, Dept. of ECE, Dhruva Institute of Engineering and Technology, Hyderabad, A.P,India.*

[3]B. T. Vedanth, *Assistant Professor, Dept. of ECE, MallaReddy Institute of Engineering and Technology, Hyderabad,A.P,India.*

ABSTRACT

The semiconductor industry, driven by ever-increasing demands for higher performance and reliability, as well as greater functionality and speeds, continuously introduces new higher density technologies and new integrated circuits. The increasing complexity of VLSI circuits, in the absence of a corresponding increase in the number of input and output pins, has made structured design for testability (DFT) and built-in self-test (BIST). Weighted pseudorandom built-in self test (BIST) schemes have been utilized in order to drive down the number of vectors to achieve complete fault coverage in BIST applications. This weighted sets comprising in to three weights, namely 0, 1, and 0.5 have been successfully utilized so far for test pattern generation, by using these three weights the result in both low testing time and low power consumption.

In this proposed system here presenting a new scheme for an accumulator-based 3-weight test pattern generation technique is presented; hence by using this proposed scheme we generates set of patterns with weights 0, 0.5, and 1. And as we well know that the accumulators are commonly found in current VLSI chips, this scheme can be efficiently utilized to drive down the hardware of BIST pattern generation, as well. and also here we did Comparisons with previously presented schemes indicate that the proposed scheme have been compares favorably with respect to the required hardware. In this paper we are presenting the comparisons of previous and proposed method and the synthesis and simulation is done by using Xilinx ISE simulator and Model Sim software respectively.

Index Terms—Built-in self test (BIST), test per clock, VLSI testing, weighted test pattern generation**.**

I. INTRODUCTION

Current trends in semiconductor technologies, as well as in design methodologies, readily indicate that the ever-increasing degree of integration of devices on a single substrate continuously demands more efforts in achieving zero-defect designs. The process of transforming the idea into a detailed circuit description in terms of the elementary circuit components constitutes design description. Clearly, this ultimate quality goal cannot be met without including testability as a design objective. Test pattern generation for BIST applications has to conform to two basic requirements: a high fault coverage and ease of incorporating a hardware necessary to produce test data onto a chip.

In this pseudorandom built-in self test (BIST) generators have been widely utilized to test integrated circuits and systems. Which include linear feedback shift registers (LFSRs) , cellular automata, and accumulators driven by a constant value. For circuits with hard-to-detect faults, a large number of random patterns have to be generated before high fault coverage is achieved. Therefore, weighted pseudorandom techniques have been proposed where inputs are biased by changing the probability of a "0" or a "1" on a given input from 0.5 (for pure pseudorandom tests) to some other value. although the weights are computed to be suitable for most faults, some faults may require long test sequences to be detected with these weight assignments if they do not match their activation and propagation requirements. Multiple weight assignments have been suggested for the case that different faults require different biases of the input combinations applied to the circuit, to ensure that a relatively small number of patterns can detect all faults. Approaches to derive weight assignments for given deterministic tests are attractive since they have the potential to allow complete coverage with a significantly smaller number of test patterns.

In order to minimize the hardware implementation cost, there we already have other schemes based on multiple weight assignments utilized weights 0, 1, and 0.5. This approach takes to decrease keeping some outputs of the generator steady (to either 0 or 1) and letting the remaining outputs change values (pseudo-) randomly (weight 0.5). This approach, helps apart from reducing the hardware overhead has beneficial effect on the consumed power, since some of the circuit under test (CUT) inputs(0 or 1) remain steady during the specific test session. One system proposed a 3-weight pattern generation scheme relying on weights

0, 1, and 0.5. The choice of weights 0, 1, and 0.5 was done in order to minimize the hardware implementation cost. another proposed system a 3-weight random pattern generator based on scan chains utilizing weights 0, 1, and 0.5, in a way similar to. Recently renovated the interest in the 3-weight pattern generation schemes, proposing an efficient compaction scheme for the 3-weight patterns 0, 1, and 0.5. From the above we can conclude that 3-weight pattern generation based on weights 0, 1, and 0.5 has practical interest since it combines low implementation cost with low test time. Current VLSI circuits, e.g., data path architectures, or digital signal processing chips commonly contain arithmetic modules [accumulators or arithmetic logic units (ALUs)]. This has fired the idea of arithmetic BIST (ABIST) . The basic idea of ABIST is to utilize accumulators for built-in testing (compression of the CUT responses, or generation of test patterns) and has been shown to result in low hardware overhead and low impact on the circuit normal operating speed. Manich *et al.* presented an accumulator-based test pattern generation scheme that compares favorably to previously proposed schemes. it was proved that the test vectors generated by an accumulator whose inputs are driven by a constant pattern can have acceptable pseudorandom characteristics, if the input pattern is properly selected. However, modules containing hard-to-detect faults still require extra test hardware either by inserting test points into the mission logic or by storing additional deterministic test patterns.

In this paper, a novel scheme for accumulator-based 3-weight generation is presented. The proposed scheme copes with the inherent drawbacks of the scheme proposed in [11]. More precisely: 1) it does not impose any requirements about the design of the adder (i.e., it can be implemented using any adder design); 2) it does not require any modification of the adder; and hence, 3) does not affect the operating speed of the adder. Furthermore, the proposed scheme compares favorably to the scheme proposed previously in terms of the required hardware overhead.

This paper is organized as follows. In Section II, the idea underlying the accumulator-based 3-weight generation is presented. In Section III, the design methodology to generate the 3-weight patterns utilizing an accumulator is presented. In Section IV, the proposed scheme is compared to the previously proposed ones. Finally, Section V, shows the simulation results and section VI concludes this paper.

II. Existing system

To improve the fault coverage is to use a "mixed-mode" approach where deterministic patterns are used to detect the faults that the pseudorandom patterns miss. Storing deterministic patterns in a read-only memory (ROM) requires a large amount of hardware overhead. Koenemann proposed a technique based on reseeding an LFSR that reduces the storage requirements. The LFSR that is used for generating the pseudorandom patterns is also used to generate deterministic *test cubes* (test patterns with unspecified inputs) by loading it with a computed seed. The number of bits that need to be stored is reduced by storing a set of seeds instead of a set of deterministic patterns.

Current VLSI circuits, e.g., data path architectures, or digital signal processing chips commonly contain arithmetic modules [accumulators or arithmetic logic units (ALUs)]. This has fired the idea of arithmetic BIST (ABIST) . The basic idea of ABIST is to utilize accumulators for built-in testing (compression of the CUT responses, or generation of test patterns) and has been shown to result in low hardware overhead and low impact on the circuit normal operating speed. In this, it was presented an accumulator-based test pattern generation scheme that compares favorably to previously proposed schemes and it was proved that the test vectors generated by an accumulator whose inputs are driven by a constant pattern can have acceptable pseudorandom characteristics, if the input pattern is properly selected. However, modules containing hard-to-detect faults still require extra test hardware either by inserting test points into the mission logic or by storing additional deterministic test patterns. 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 tradeoff between the number of weight assignments and the number of observation points required to achieve complete fault coverage. In order to overcome this problem, an accumulator-based weighted pattern generation scheme was proposed in. The scheme generates test patterns having one of three weights, namely 0, 1, and 0.5 therefore it can be utilized to drastically reduce the test application time in accumulator-based test pattern generation.

However, this system possesses three major drawbacks which is followed as: 1) it can be utilized only in the case that the adder of the accumulator is a ripple carry adder; 2)it requires redesigning the accumulator; this modification, apart from being costly, requires redesign of the core of the data path,

a practice that is generally discouraged in current BIST schemes; and 3) it increases delay, since it affects the normal operating speed of the adder.

III. Proposed System

To overcome the draw backs presented in the existing systems here we shall illustrate the idea of an accumulator-based 3-weight pattern generation by means of an example. Let us consider the test set for the c17 ISCAS benchmark given in Table I. Starting from this deterministic test set, in order to apply the 3-weight pattern generation scheme, one of the schemes proposed in and can be utilized. According to these schemes, a typical weight assignment procedure would involve separating the test set into two subsets, S1 and S2 as follows:

S1={T1,T4}

S2={T2,T3}

The weight assignments for these subsets is

W(S1)={--,--,1--,1}

W(S2)={--,--,0,1,0}

where a "_" denotes a weight assignment of 0.5, a "1" indicates that the input is constantly driven by the logic "1" value, and "0" indicates that the input is driven by the logic "0" value. 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. In the first assignment, inputs A[2] and A[0] are constantly driven by "1", while inputs A[4], A[3], A[1] are pseudo randomly generated (i.e., have weights 0.5). Similarly, in the second weight assignment (subset S2), inputs A[2] and A[0] are constantly driven by "0", input A[1] is driven by "1" and inputs A[4] and A[3] are pseudo randomly generated.

The above reasoning calls for a configuration of the accumulator, where the following conditions are met: 1) an accumulator output can be constantly driven by "1" or "0" and 2) an accumulator cell with its output constantly driven to "1" or "0" allows the carry input of the stage to transfer to its carry output unchanged. This latter condition is required in order to effectively generate pseudorandom patterns in the accumulator outputs whose weight assignment is "_".

Test vector	Inputs A[4:0]
T1	00101
T2	01010
T3	10010
T4	11111

Table I : Test Set For The C17 Benchmark

The implementation of the weighted-pattern generation scheme is based on the full adder truth table, presented in Table II. From this we can see that in lines #2, #3, #6, and #7 of the truth table Cout=Cin. Therefore, in order to transfer the carry input to the carry output, it is enough to set A[i] = NOT B[i]. The proposed scheme is based on this observation.

#	Cin	A[i]	B[i]	S[i]	C_{out}	Comment
1	0	0	0	0	0	
2	**0**	0	1	1	**0**	$C_{out}=C_{in}$
3	**0**	1	0	1	**0**	$C_{out}=C_{in}$
4	0	1	1	0	1	
5	1	0	0	1	0	
6	**1**	0	1	0	**1**	$C_{out}=C_{in}$
7	**1**	1	0	0	**1**	$C_{out}=C_{in}$
8	1	1	1	1	1	

Table II: Truth Table For Full Adder

The implementation of the proposed weighted pattern generation scheme is based on the accumulator cell presented in Fig, which consists of a Full Adder (FA) cell and a D-type flip flop with asynchronous set and reset inputs whose output is also driven to one of the full adder inputs. In Fig, we assume, without loss of generality, that the set and reset are active high signals. In the same figure the respective cell of the driving register B[i] is also shown.

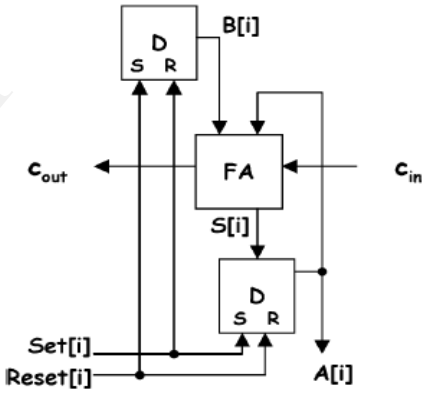


Fig. 1. Accumulator cell.

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. A preselected number of patterns N is applied under every weight assignment. A weight of 0.5 assigned to an input i by a weight assignment w implies that pseudo-random patterns are applied to input i while N test patterns are applied to the circuit; a weight of 0 assigned to input i implies that input i is held at 0 constantly for the N test patterns and a weight of 1 assigned to input i implies that input i is held at 1 constantly for the N test patterns. The weight assignments are based on a deterministic test set. Each weight assignment is obtained by intersecting a subset of deterministic test

patterns. The intersection of identical values, 0 or 1, yields a weight of 0 or 1, respectively.

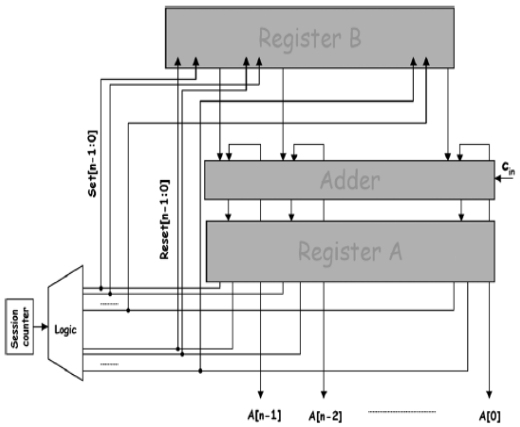


Fig2: Block diagram for proposed scheme

IV. COMPARISON WITH SCAN-BASED SCHEMES

Since the test application algorithms that have been invented and applied by [5], [8], and [9] can be equally well applied with the proposed scheme, test application time is similar to that reported there. Therefore, the comparison will be performed with respect to hardware overhead. In the 3-weight pattern generation scheme proposed by Pomeranz and Reddy in [5] 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 as well as a scan counter, common to all scan schemes. A number of 3-gate modules is required for every required weighted input (in [the hardware overhead is calculated for the ISCAS'85 benchmarks). we have presented arithmetic results for some of the ISCAS'85 benchmarks. For the calculations in Table III, we have assumed that schemes [5] and [8] are applied to a circuit with scan capability; therefore, the hardware overhead to transform the latches into scan latches is not taken into account. For the scheme proposed in [5], the total hardware overhead comprises the hardware for the weighting gates (second column), the scan counter and the LFSR implementation.

In order to calculate the numbers in the second column, we utilized the data found in [5]. For the scheme in [8], the LFSR, the scan counter and the decoding logic are required; the hardware overhead for the decoding logic has been quoted from [8, Table I].

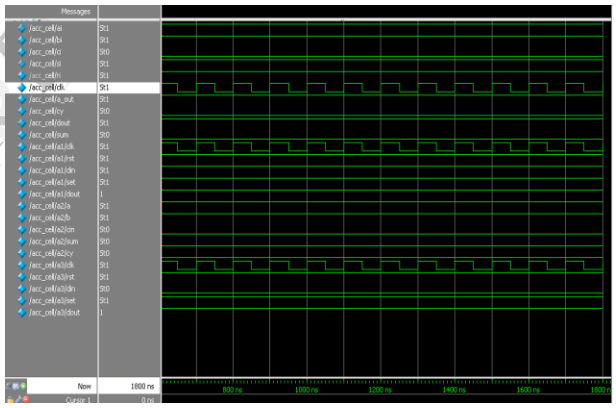
CUT	Pomeranz [5]				Wang [8]				Proposed
	weighting gates + scan counter + LFSR = Total				LFSR + decoding logic + scan counter = Total				
c880	5	47	47	**99**	47	6	47	**100**	**27**
c1355	1	43	43	**87**	43	6	43	**92**	**38**
c1908	0	40	40	**80**	40	2	40	**82**	**23**
c2670	542	63	63	**668**	63	39	63	**165**	**101**
c3540	2	45	45	**92**	45	24	45	**114**	**73**
c5315	0	60	60	**120**	60	7	60	**127**	**39**
c7552	1134	62	62	**1258**	62	66	62	**190**	**139**

Table III: Comparisons with the scan schemes proposed in [5] and [8]

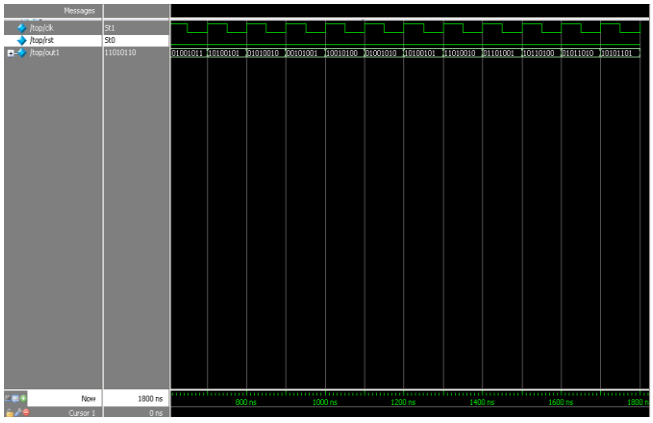
V. Simulation results

VERILOG is frequently used for two different goals: simulation of electronic designs and synthesis of such designs. Synthesis is a process where a VERILOG is compiled and mapped into an implementation technology such as an FPGA or an ASIC. Many FPGA vendors have free tools to synthesize VERILOG for use with their chips, where ASIC tools are often very expensive. Here shows the simulation and synthesis results for proposed design is to be presented as follows

ACCUMULATOR



TOP MODULE



VI. CONCLUSION

Finally, In this paper we have presented a modified version of the existing system of 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.

REFERENCES

[1] P. Bardell, W. McAnney, and J. Savir, Built-In Test For VLSI: Pseudorandom Techniques. New York: Wiley, 1987.
[2] P. Hortensius, R. McLeod, W. Pries, M. Miller, and H. Card, "Cellular automata-based pseudorandom generators for built-in self test," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 8, no. 8, pp. 842–859, Aug. 1989.
[3] A. Stroele, "A self test approach using accumulators as test pattern generators," in Proc. Int. Symp. Circuits Syst., 1995, pp. 2120–2123.
[4] H. J. Wunderlich, "Multiple distributions for biased random test patterns," in Proc. IEEE Int. Test Conf., 1988, pp. 236–244.
[5] I. Pomeranz and S. M. Reddy, "3 weight pseudo-random test generation based on a deterministic test set for combinational and sequential circuits," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.12, no. 7, pp. 1050–1058, Jul. 1993.
[6] K. Radecka, J. Rajski, and J. Tyszer, "Arithmetic built-in self-test for DSP cores," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 16, no. 11, pp. 1358–1369, Nov. 1997.
[7] J. Rajski and J. Tyszer, Arithmetic Built-In Self Test For Embedded Systems. Upper Saddle River, NJ: Prentice Hall PTR, 1998.
[8] S. Wang, "Low hardware overhead scan based 3-weight weighted random BIST," in Proc. IEEE Int. Test Conf., 2001, pp. 868–877.
[9] S. Zhang, S. C. Seth, and B. B. Bhattacharya, "Efficient test compaction for pseudo-random testing," in Proc. 14th Asian Test Symp., 2005, pp. 337–342.
[10] J. Savir, "Distributed generation of weighted random patterns," IEEE Trans. Comput., vol. 48, no. 12, pp. 1364–1368, Dec. 1999.
[11] I. Voyiatzis, D. Gizopoulos, and A. Paschalis, "Accumulator-based weighted pattern generation," presented at the IEEE Int. Line Test Symp., Saint Raphael, French Riviera, France, Jul. 2005.