RESEARCH ARTICLE OPEN ACCESS

Performance of Responsibility Discovery in VLSI Circuit Using Accumulator

Md. Azmathullah, K. Nanda Kumar,Syed Reshma
M.Tech [Vlsisd], Department Of Ece,Chadalawada Ramanamma Engineering College [1]
M.Tech, Assistant Professor, Department Of Ece, Chadalawada Ramanamma Engineering College[2]
M.Tech, Assistant Professor, Department Of Cse, Chadalawada Ramanamma Engineering College[3]

ABSTRACT
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. 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. In this paper 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 BIST pattern generation, as well. The output of this pattern generator is applied to a circuit under test where the errors are detected by using MISR circuit.
Index Terms— VLSI Testing, Weighted Test Pattern Generation, Built-In Self-Test (BIST), Test Per Clock,

I. INTRODUCTION

In recent years, Pseudorandom built-in self test (BIST) generators have been widely utilized to test integrated circuits and systems. The arsenal of pseudorandom generators includes, among others, linear feedback shift registers (LFSRs) [1], cellular automata [2], and accumulators driven by a constant value [3]. 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 ofa "0" or a "1" on a given input from 0.5 (for pure pseudorandom tests)to some other value [10], [15].Weighted random pattern generation methods relying on a single weight assignment usually fail to achieve complete fault coverage using a reasonable number of test patterns since, 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 [4]. Approaches to derive weight assignments forgiven deterministic tests are attractive since they have the potential to allow complete coverage with a significantly smaller number of test patterns [10].

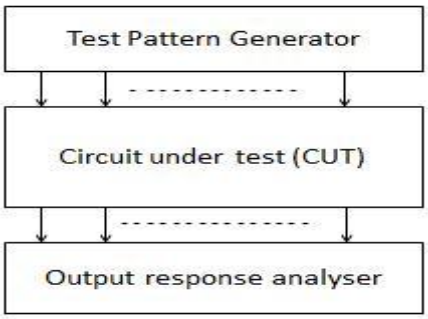

Fig 0 . Block Diagram for BIST

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) [6]. 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. In [7], 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 order to overcome this problem, an accumulator-based weighted pattern generation

scheme was proposed in [11]. 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, the scheme proposed in [11] possesses three major drawbacks:

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.

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 in [11] and [22] in terms of the required hardware overhead.

TABLE I
TEST SET FOR THE C17 BENCHMARK

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

TABLE II
TRUTH TABLE OF THE FULL ADDER

#	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	

II. ACCUMULATOR-BASED 3-WEIGHT PATTERN GENERATION

A. Weighted Pattern Testing

Weighted pattern testing is performed by weighting the signal probability (probability that the signal is a _1_) for each input to the CUT. Two issues in weighted pattern testing are what set of weights to use and how to generate the weighted signals. For computing weight sets, many techniques have been proposed. It has been shown that for most circuits, multiple weight sets are required to achieve sufficient fault coverage. For BIST, the weight sets must be stored on-chip and control logic is needed to switch between them which can result in a lot of overhead. In order to reduce the BIST overhead for weighted pattern testing, researchers have looked for efficient methods for on-chip generation of weighted patterns.

B. *3-Weight Pattern Generation*

The implementation of the weighted-pattern generation scheme is based on the full adder truth table, presented in Table I. From Table I, we can see that in lines #2, #3, #6, and #7 of the truth table, $C_{out} = C_{in}$. So, 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.

The implementation of the proposed weighted pattern generation scheme is based on the accumulator cell presented, 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. we assume, without loss of generality, that the set and reset are active high signals. In the same Fig.2, the respective cell of the driving register B[i] is also shown. For this accumulator cell, one out of three configurations can be utilize.

III. DESIGN METHODOLOGY

The implementation of the weighted-pattern generation scheme is based on the full adder truth table, presented in Table II. From Table II we can see that in lines #2, #3, #6, and #7 of the truth table, $C_{OUT}=Cin$. 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. The implementation of the proposed weighted pattern generation scheme is based on the accumulator cell presented in Fig. 1, 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., 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. For this accumulator cell, one out of three configurations can be utilized, as shown in Fig. .

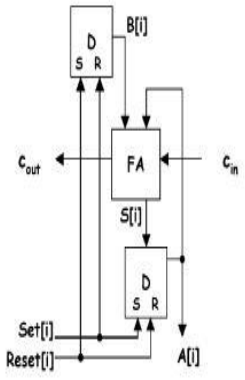

Fig. 1. Accumulator cell.

In Fig. we present the configuration that drives the CUT inputs when is required. A[i]=1 and Set[i]=1 and Reset[i]=0 hence A[i]=1 and B[i]=0. Then the output is equal to 1, and C_{in} is transferred to *Cout.*

In Fig., we present the configuration that drives the CUT inputs when A[i] ="_"is required. Set[i]=0 and Reset[i]=1and hence A[i]=1 and B[i]=0 Then, the output is equal to 0 and C_{in} is transferred to C_{out}.

In Fig, we present the configuration that drives the CUT inputs when A[i] = "_" is required. Set[i]=0 and Reset[i]=0 The D input of the flip-flop of register B is driven by either 1 or 0, depending on the value that will be added to the accumulator inputs in order to generate satisfactorily random patterns to the inputs of the CUT.

In Fig, 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.

IV. COMPARISONS

In this section, we shall perform comparisons in three directions. In Section 4, we shall compare the proposed scheme with the accumulator- based 3-weight generation scheme that has been proposed in [11].

IV.1. Comparisons with [11]

The number of test patterns applied by [11] and the proposed scheme is the same, since the test application algorithms that have been invented and applied by previous researchers, e.g., [5], [8], [9] can be equally well applied with both implementations. Therefore, the comparison will be performed with respect to:

1) the hardware overhead And
2) 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 the number of test sessions (i.e., weight assignments) of the weighted test set. The scheme proposed in [11] requires the redesign of the adder; more precisely, two NAND gates are inserted in each cell of the ripple-carry adder. In order to Provide the inputs to the set and reset inputs of the flip flops; decoding logic is implemented, similar to that in [8]. For the proposed scheme, no modification is imposed on the adder of the accumulator. Therefore, there is no impact on the data path timing characteristics.

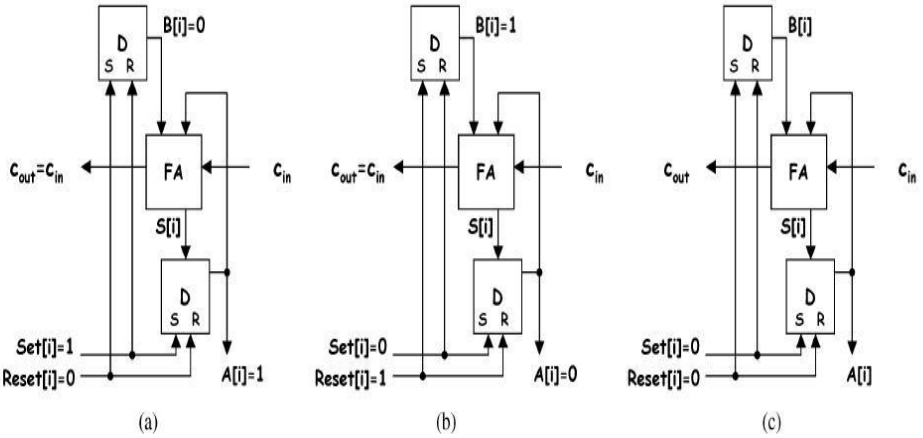

Fig. 2. Configurations of the accumulator cell of Fig. 1.

V. Fault & Fault Models

A fault is the representation of a defect reflecting a physical condition that causes a circuit to fail to perform in a required manner. A failure is a deviation in the performance of a circuit or system from its specified behavior and represents an irreversible state of a component such that it must be repaired in order for it to provide its intended design function. A circuit error is a wrong output signal produced by a defective circuit. A circuit defect may lead to a fault, and it can result in a system failure. The reduction in feature size increases the probability that a manufacturing defect in the IC will result in a faulty chip. FPGA's are no exception from this. A very small defect can easily found when the feature size of the device is less than 100 nm. Furthermore, it takes only one faulty CLB or wire to make the entire FPGA fail to function properly. Yet, defects created during the manufacturing process are unavoidable and, as a result, some number of FPGA's is expected

to be faulty; therefore, testing is required to guarantee fault free FPGA's.

Faults can be divided into two categories:
1. Permanent Faults
2. Transient Faults

Fabrication faults and design faults are among the Permanent Faults. Transient Faults, commonly called single event upsets (SEUs), are brief incorrect values resulting from external forces (terrestrial radiation, particles from solar flares, cosmic rays, and radiation from other space phenomena) altering the balance or locations of electrons, usually in a small area of the system.

Tolerating permanent faults is critical to maximizing device and system yields to decrease costs, and to increasing the lifespan of deployed devices. To effectively evaluate the quality of a set of tests for a product, as well as to evaluate the effectiveness of a BIST approach in its application to that product, fault models are required for emulation of faults or defects in a simulation environment. Because of the diversity of defects, it is difficult to generate tests for real defects. Fault models are necessary for generating and evaluating a set of test vectors.
Fault models can be divided into three types

1. Interconnect Fault Model
2. Logic Block Fault Model
3. Delay Fault

Logic Block fault model: Logical faults represent the effect of physical faults on the behavior of the system

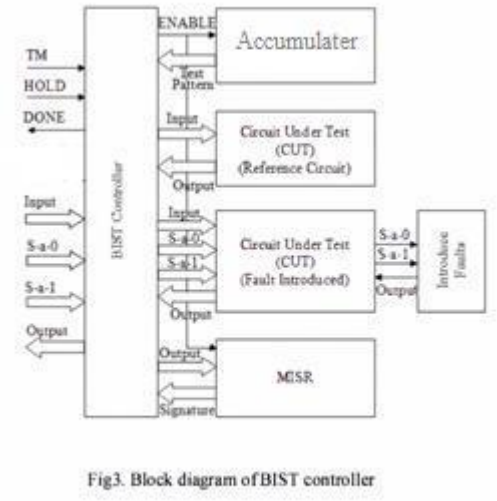

Fig3. Block diagram of BIST controller

VI. Simulation Results

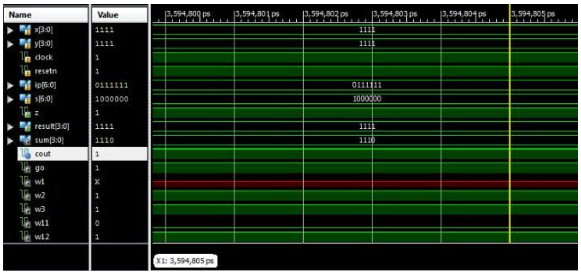

Fig 4 .Simulation results for Fault Detection in the circuit.

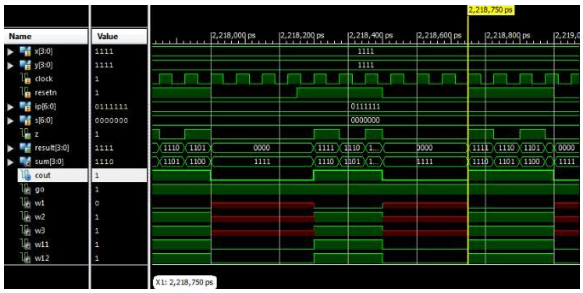

Fig 5 .Error output due to fault in the circuit

VII. Conclusion

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 [11] indicate that the hardware overhead of the proposed scheme is lower (75%), while at the same time no redesign of the accumulator is imposed, thus resulting in reduction of 20%–95% in test application time. Comparisons with scan based schemes [5], [8] show that the proposed schemes results in lower hardware overhead. Finally, comparisons with the accumulator-based scheme proposed in [22] reveal that the proposed scheme results in significant decrease (98%) 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 fbiased 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. S. Zhang, S. C. Seth, and B. B. Bhattacharya,

[9] S. Wang, "Low hardware overhead scan based 3-weight weightedrandom BIST architectures," U.S. Patent 6 886 124, Apr. 26, 2005.

[10] S. Manich, L. Garcia, L. Balado, J. Rius, R. Rodrøguez, and J. Figueras, "Improving the efficiency of arithmetic bist by combining targeted andgeneral purpose patterns," presented at the Des. Circuits Integr. Syst.(DCIS), Bordeaux, France, 2004.

[11] A. D. Singh, M. Seuring, M. Gossel, and E. S.Sogomonyan, "Multimode scan: Test per clock BIST for IP cores," *ACM Trans. Design*Autom. Electr. Syst., vol. 8, no. 4, pp. 491–505, Oct. 2003.

[12] S. Manich, L. Garcia, L. Balado, E. Lupon, J.Rius, R. Rodriguez, and J.Figueras, "On the selection of efficient arithmetic additive test pattern generators," in Proc. Eur.Test Workshop, 2003.

[13] A. Stroele, "A self testapproach using accumulators as test patterngenerators," in Proc. Int. Symp. Circuits Syst., 1995.