Design and Testing of Weighted Pattern Generation of an Accumulator

Nagaraju Sangepu
Associate Professor , CSE
Kakatiya Institute of Technology &Sciences
Warangal, India.
Email: nag.sangepu@gmail.com

Sravani Kummera
M.Tech(VLSI & ES)
Kakatiya Institute of Technology &Sciences
Warangal, India.
Email:kummerasravani@gmail.com

Abstract— **Pseudorandom built-in self test (BIST) generators have been widely utilized to test integrated circuits and systems. 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. Weighted pattern generation of an accumulator is achieved by using weights 1, 0, and 0.5(for pure pseudorandom tests). 3-Weight Pattern Generation on semi custom Design, the main goal for designing it is to maintain speed, area.**

Keywords- Built-in self test (BIST), test per clock, VLSI testing, weighted test pattern generation.

I. INTRODUCTION *(HEADING 1)*

To test integrated circuits and systems, Pseudorandom built-in self test (BIST) generators have been widely utilized. The arsenal of pseudorandom generators includes, accumulators driven by a constant value [3]. A large number of random patterns have to be generated to achieve high fault coverage for circuits with hard to detect faults. 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 [10], [15].

Some faults may require long test sequences to be detected, complete fault coverage usually can't be achieved by the weighted random pattern generation methods that depends on a single weight assignment.Although the weights are computed to be suitable for most faults, by using a reasonable number of test patterns.If their activation and propagation requirements do not match. 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], Multiple weight assignments have been suggested. 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[10].

For the minimization of the hardware implementation cost, other schemes were introduced based on multiple weight assignments by utilization of weights 0, 1, and 0.5. By this approach some outputs of the generator remains steady (to either 0 or 1) and letting the remaining outputs change values (pseudo-) randomly (weight 0.5). This approach, not only reduces the hardware overhead but also has a beneficial effect on the consumed power, since some of the circuit under test (CUT) inputs (those having weight 0 or 1) remain steady during the specific test session [30]. Pomeranz and Reddy [5] 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. Wang [8], [13] proposed a 3-weight random pattern generator based on scan chains utilizing weights 0, 1, and 0.5, in a way similar to [5]. Recently, Zhang *et al.* [9] 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)]. Thus the idea of arithmetic BIST (ABIST) [6] arises. 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 [22]–[27]. In [22], 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

Identify applicable sponsor/s here. *(sponsors)*

test points into the mission logic or by storing additional deterministic test patterns [24], [25].

In order to overcome this problem, accumulator 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 datapath, 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.

A scheme for accumulator-based 3-weight generation is presented in [33]. This 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. But it has its drawbacks too, By using the adder in this scheme the complexity of the hardware used increases.

In this paper a novel scheme for weighted pattern generation of an accumulator is presented. Replacing the adder by Xor gate we can cope with that previous scheme[33].thus reducing the hardware, complexity for the test pattern generation.

This paper is organized as follows. In Section II, the idea underlying the weighted pattern generation of an accumulator 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, concludes this paper.

II. WEIGHTED PATTERN GENERATION OF AN ACCUMULATOR

The idea of an weighted pattern generation of an accumulator was illustrated by means of an example. Let us consider the test set for the c17 ISCAS benchmark [12], [31] given in Table I.

In order to apply the 3-weighted pattern generation scheme, one of the schemes proposed in [5], [8], and [9] 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} and S2={T2,T3}. The weight assignments for these subsets is W(S1)={-,-,1,-,1}, and 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. 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.

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	

These are the 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 "_".

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, $C_{out}=C_{in}$.

Therefore, in order to transfer the carry input to the carry output, it is enough to set A[i]=NOT(Bi). The proposed scheme is based on this observation.

The implementation of the 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. In Fig. 1, we assume that

the set and reset are active high signals. For this accumulator cell, one out of three configurations can be utilized, as shown in Fig. 2.

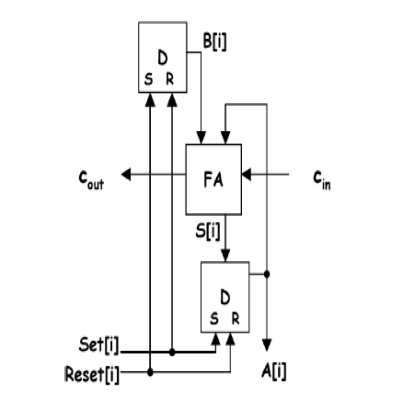

Fig.1. Accumulator Cell

In Fig. 2(a) we present the configuration that drives the CUT inputs when A[i]=1 is required. Set[i] = 1and Reset[i] = 0 and hence A[i] = 1 and B[i] = 0. Then the output is equal to 1, and C_{in} is transferred to C_{out}.

In Fig. 2(b), we present the configuration that drives the CUT inputs when A[i] = 0 is required. Set [i] = 0 and Reset[i] = 1 and hence A[i] = 0 and B[i] = 1. Then, the output is equal to 0 and C_{in} is transferred to C_{out}.

In Fig. 2(c), 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.3. Replacement of adder by an Xor gate is done. Generally A full adder can be implemented in many different ways such as with a custom transistor-level circuit or composed of other gates. In this implementation, the final OR gate before the carry-out output may be replaced by an XOR gate without altering the resulting logic. Using only two types of gates is convenient if the circuit is being implemented using simple IC chips which contain only one gate type per chip.

In Fig. 4, 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.

III. COMPARISONS

In Section IV-A, we shall compare the proposed scheme with the accumulator- based 3-weight generation scheme that has been proposed in [11].

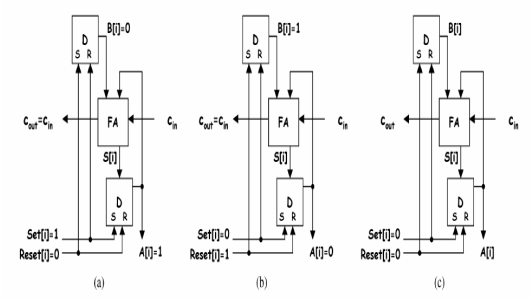

Fig.2. Configurations of Accumulator Cell

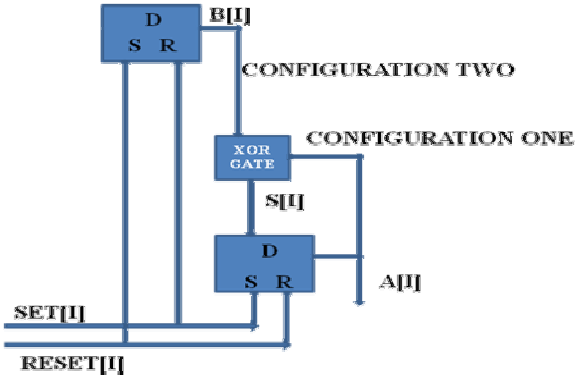

Fig.3. Implementation of weighted pattern generation of an accumulator by using XOR gate.

. In Section IV-B, we shall compare the proposed scheme with the 3-weight scan schemes that have been proposed in [5] and [8]. In Section IV-C, we shall compare the proposed scheme with accumulator based 3-weight pattern generation that has been proposed in [33].

A. Comparisons With [11]

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 based on the number of test patterns applied by [11]

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 $log_2{}^{k}$ its, where k is the number of test session 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].

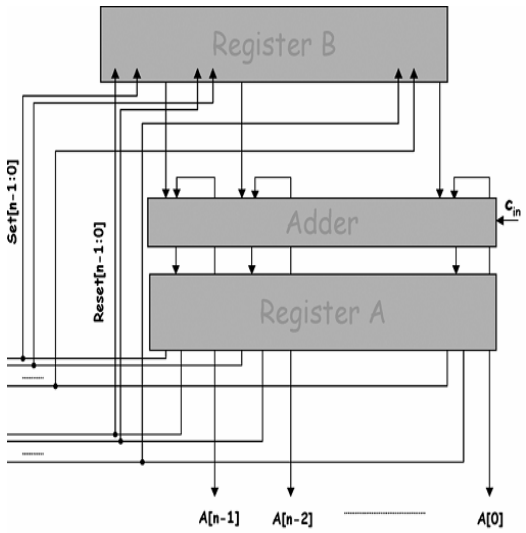

Fig.4. Block Diagram of an weighted pattern generation of an accumulator.

.

For the proposed scheme, no modification is imposed on the adder of the accumulator. Therefore, there is no impact on the datapath timing characteristics.

In Table III, the hardware overheads are calculated in gate equivalents, where an n-input NAND or NOR accounts for 0.5 n gates and an inverter accounts for 0.5 gates, as proposed in [8]. For the calculation of the delay in the adder operation we have considered both ripple carry and prefix adder implementations. For the comparisons of the ripple carry adder implementations, the adder cell utilized in [11] is considered; in the cell presented in [11], initially the delay from the C_{in} to C_{out} of the adder *cell* is two NAND gates and one XOR gate; in the modified cell proposed in [11], the delay is increased to three NAND and one XOR gate; we have considered that the delay of a NAND gate is one gate equivalent, while the delay of an XOR gate is two gate equivalents. Since the implementation of the proposed scheme does not rely on a specific adder design, the utilization of a prefix adder can result in impressive results. For the calculation of the delay of prefix adders, the formula obtained by [29] is utilized, where the delay is of the order $4 \times log_2 n$, where n is the number of the adder stages. From Table III, we can see that the proposed scheme results in 57%–90% decrease in hardware overhead, while at the same time achieving a decrease in operational delay overhead that ranges from 84% to 97% for the considered benchmarks.

In Table III we present comparison results for some of the ISCAS'85 benchmarks. In the first column of Table III, we present the benchmark name; in the second and third columns we present the hardware overhead of the accumulator-based scheme proposed in [11] and in this work, respectively; in the fourth column we present the decrease of the proposed scheme over [11] fourth column we present the decrease of the proposed scheme over [11]. In the fifth through the seventh columns, we present the delay of the adder in terms of number of gates that the carry signal has to traverse, from the C_{in} input of the adder to the C_{out} output, and the respective decrease obtained by the proposed scheme.

TABLE III
COMPARISONS WITH [11]

circuit	hardware overhead			delay from c_{in} to c_{out}				
					(ripple)		(prefix)	
name	[11]	pr op.	de cr.	[11]	pr op.	de cr.	pr op	de cr.
c880	41%	8%	81%	240	180		24	90%
c1355	28%	7%	74%	164	123		22	87%
c1908	13%	3%	77%	132	99		21	84%
c2670	34%	8%	75%	932	699	20%	32	97%
c3540	11%	5%	57%	200	150		23	89%
c5315	17%	2%	90%	712	534		30	96%
c7552	17%	4%	75%	828	621		31	96%

B. Comparisons With Scan-Based Schemes [5], [8]

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 input 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. A number of 3-gate modules is required for every required weighted input Wang [8] proposed a low-overhead 3-weight random BIST scheme, again based on scan chains. He proposed two schemes, namely *serial fixing BIST* and *parallel fixing BIST*. Serial fixing scheme is shown to be more costly [8]; therefore we shall concentrate our comparisons to the parallel fixing BIST scheme. The hardware overhead of the decoding logic for some of the ISCAS benchmarks is calculated in [8, Table I]. All schemes require the application of the *session counter*, required to alter among the different weight sessions. Schemes proposed in [5] and [8] are test per scan schemes, and, of course, assume the existence of scan capability of the latches of the design.

In Table IV, we have presented arithmetic results for some of the ISCAS'85 benchmarks. For the calculations in Table IV, 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 in [5, Table V]. 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].

TABLE IV

COMPARISONS WITH THE SCHEME PROPOSED IN [22]

benchmark			[22]		proposed	
circuit	h/w	#inp	tests	h/w	#tests	h/w
c880	383	60	1112	36	768	27
c1355	546	41	1409	26	1024	38
c1908	880	33	3198	23	1536	23
c2670	1193	157	1962	1386	4096	101
c3540	1669	50	2167	31	1536	73
c5315	2307	178	1453	189	2048	39
c7552	3512	206	2918	1090	4608	139
s5378	1004	214	2078	791	5120	47
s9234	2027	247	14803	4763	11264	181
s13207	2573	700	14476	7497	12288	61
s15850	3448	611	14902	18438	21504	159
s38584	11448	1464	8449	26235	16384	82
Average values			5744	5042	6848	81

From Table V it is trivial to see that the proposed scheme presents an important decrease in the hardware overhead, while the number of tests is comparable, while in some cases it also outperforms the scheme in [22]. It is interesting to note that the hardware overhead (with respect to the hardware overhead of the benchmarks) is practical, in contrast to [22] which sometimes exceeds the benchmark hardware (c2670, s5378, s9234, s13207, s15850, s38584). It is also interesting to note the average values presented in the last line of the Table. The average increase in the number of tests is 19%, while the average decrease in hardware overhead is 98%.

IV. SIMULATION

Here Simulation of results are classified as two types.1.Top design module simulation. 2. Top design module Stimulus simulation The top design module depends on manually functional verification. The simulation depending on design specifications (inputs words length or inputs bit length). If the design specifications are scalar then it is easy for simulation otherwise Top design module verification is difficult.

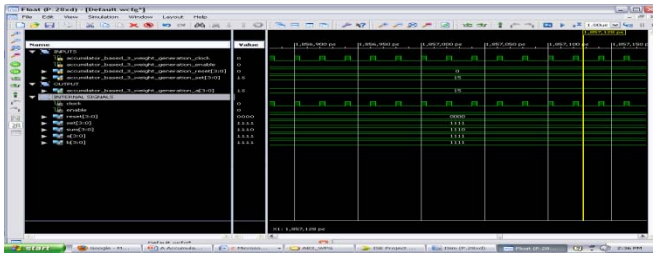

Fig.[5]. Top design module Stimulus simulation with reset is enabled.

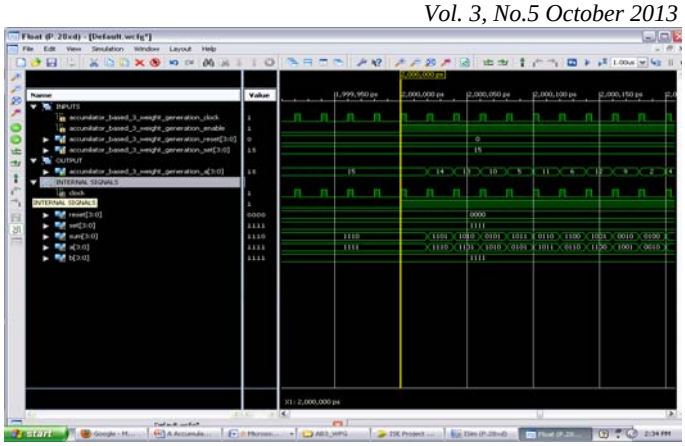

Fig.[6] Top design module Stimulus simulation with reset is disabled

V. CONCLUSION

We have tested the weighted pattern generation of an accumulator 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, 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 in hardware overhead. The main goal for designing it is to maintain **speed** and **area** which are successfully achieved.Further verify this Project on Static Timing analysis (STA) and System on chip verification (SOC) for Practical Debugging and lay out design.

VI. VI .ACKNOWLEDGEMENTS

Sangepu Nagaraju attained his B.Tech in Computer Science and Engineering from Amaravati University, Andhra Pradesh, India in 1987 and M.Tech in Computer Science and engineering from JNTU, Hyderabad, India in 2007. He is Pursuing Ph.D in Semantic Web JNTU, Hyderabad, India. He is now working as Associate Professor in the Dept. of Computer Science and Engineering in Kakatiya Institute of Technology and Science in Warangal, India since August 1998.His research interests include Data mining, Genetic Algorithm, Evolutionary Algorithms and Semantic Web Mining.

Sravani Kummera was born in Nagarjuna Sagar, Andhra Pradesh, India in 1990.She received her B.Tech degree in Electronics Communication and Engineering from JNTU, Hyderabad,India in 2011.She is pursuing her Masters degree in VLSI&ES in Kakatiya Institute of Technology and Science, Warangal, India. Her research interests include Low Power VLSI Design and Digital Communications.

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.

[12] F. Brglez and H. Fujiwara, "A neutral netlist of 10 combinational benchmarks circuits and a target translator in FORTRAN," presented at the Int. Symp. Circuits Syst., Kyoto, Japan, 1985.

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

[14] C. Hamacher, Z. Vranesic, and S. Zaky, Computer Organization. New York: McGraw Hill, 2002.

[15] F. Brglez, C. Gloster, and G. Kedem, "Hardware-based weighted random pattern generation for boundary scan," in Proc. IEEE Int. Test Conf. (ITC), 1989, pp. 264–274.

[16] H.-J. Wunderlich, "Self test using un equi probable random patterns," in *Proc. 17th Int. Symp.* Fault-Tolerant Comput.(FTCS), 1987, pp. 258–263.

[17] O. Novák, Z. Pløva, J. Nosek, A. Hlawiczka, T. Garbolino, and K.Gucwa, "Test-per-clock logic BIST with semi-deterministic test patterns and zero-aliasing compactor," J. Electron. Testing : Theor. Appl., vol. 20, no. 1, pp. 109–122, Feb. 2004.

[18] Y. Son, J. Chong, and G. Russell, "E-BIST: Enhanced test-per-clock BIST architecture," IEE Proc.—Comput. Digit. Techn., vol. 149, pp. 9–15, Jan 2002.

[19] K. Yamaguchi, M. Inoue, and H. Fujiwara, "Hierarchical BIST: Test per- clock BIST with low overhead," Electron. Commun. Japan (Part II: Electron.), vol. 90, no. 6, pp. 47–58, Jun. 2007.

[20] E. Kalligeros, X. Kavousianos, D. Bakalis, and D. Nikolos, "An efficient seeds selection method FOR lfsr-based test-per-clock BIST," in Proc. Int. Symp. Quality Electron. Des., 2002, p. 261.

[21] 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.

[22] S. Manich, L. Garcia-Deiros, and J. Figueras, "Minimizing test time in arithmetic test-pattern generators with constrained memory resources,"IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 26, no. 11, pp. 2046–2058, Nov. 2007.

[23] S. Manich, L. Garcia, and J. Figueras, "Arithmetic test pattern generation: A bit level formulation of the optimization problem," presented at the Des. Circuits Integr. Syst. (DCIS), Lisbon, Portugal, 2005.

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

[25] 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, pp. 9–14.

[26] I. Voyiatzis, "An ALU based BIST scheme for word-organized rams," IEEE Trans. Comput., vol. 57, no. 8, pp. 1012–1022, Aug. 2008.

[27] I. Voyiatzis, "An accumulator—based compaction scheme with reduced aliasing for on-line BIST of rams," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 16, no. 9, pp. 1248–1251, Sep. 2008.

[28] A. Rashid Mohamed, Z. Peng, and P. Eles, "A wiring-aware approach to minimizing built-in self-test overhead," J. Comput. Sci. Technol., vol. 20, no. 2, pp. 216–223.

[29] R. Zimmerman, "Binary Adder Architectures for Cell-Based VLSI and their Synthesis," Ph.D. dissertation, Swiss Federal Inst. Technol. (ETH), Zurich, Switzerland, 1998 [Online].Available:http://www.iis.ee.ethz.ch/~zimmi/publications/comp_arith_notes/s/gz

[30] M. B. Santos, I. C. Teixeira, J. P. Teixeira, S. Manich, R. Rodriguez, and J. Figueras, "RTL level preparation of high-quality/low-energy/ low-power BIST," in Proc. Int. Test Conf., 2002, pp. 814–823.

[31] C. Albrecht, "IWLS 2005 Benchmarks," Lake Arrowhead, CA, 2005 [Online]. Available: http://www.iwls.org/ iwls2005/benchmarks.html

[32] L. R. Huang, J. Y. Jou, and S. Y. Kuo, "Gauss-elimination-based generation of multiple seed-polynomial pairs for LFSR," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 16, no. 9, pp.1015–1024, Sep. 1997.