Low Power BIST Based on Scan Partitioning

Jinkyu Lee and Nur A. Touba

Computer Engineering Research Center
Department of Electrical and Computer Engineering
University of Texas, Austin, TX 78712
{jlee2, touba}@ece.utexas.edu

Abstract

A built-in self-test (BIST) scheme is presented which both reduces overhead for detecting random-pattern-resistant (r.p.r.) faults as well as reduces power consumption during test. 3-valued weights are employed to detect the r.p.r. faults. The key idea is to use a new scan partitioning technique and decoding methodology that exploits correlations in the weight sets to greatly reduce the hardware overhead for multiple weight sets and reduce the number of transitions during scan shifting. The proposed scheme is simple to implement and only constrains the partitioning of scan elements into scan chains and not the scan order thereby having minimal impact on routing. Consequently, the proposed scheme can be easily implemented in standard design flows used in industry. Experiments indicate the scheme can achieve 100% fault coverage and % to 9% scan power reduction with relatively small hardware overhead.

1. Introduction

Built-in self-test (BIST) involves performing on-chip test pattern generation and test response compaction. BIST has many well-known advantages including no need for storing data on a tester, application of large numbers of patterns resulting in improved coverage of non-modeled faults, and the ability to test out in the field. The most economical BIST schemes involve using pseudo-random patterns. However, there are two major drawbacks for pseudo-random BIST. One is that shifting pseudo-random patterns into scan chains creates a large amount of switching activity resulting in high power dissipation. The other problem is that some faults have low detection probabilities and consequently are random pattern resistant (r.p.r.) [1]. In order to detect the r.p.r. faults, additional hardware is required to either weight the patterns, generate deterministic patterns, or insert test points. This paper presents a new weighted pattern BIST scheme that significantly reduces the hardware overhead required for high fault coverage while also significantly reducing the power dissipation.

Weighted pattern testing has been well studied in the literature. Early work in [2] described a weighted random pattern generator in which pseudo-random patterns are biased by changing the probability of each input bit position being a ‘ ’ or a ‘1’. The ”weight” assigned to each input bit position is determined in a way that detects as many r.p.r. faults as possible, thereby increasing the fault coverage. A -valued weight set generation algorithm based on a deterministic test set was presented in [ ]. Each input bit position is weighted to one of three values, , 1, or random, and this can reduce hardware overhead for weighted random test and increase fault coverage compared to the conventional weighted random test. A scheme to further reduce hardware overhead for a -valued weighted random test by scan re-ordering was presented in [ ]. The drawback of scan re-ordering is that it can cause routing congestion which can increase die size and chip complexity. A method to reduce power dissipation during BIST was presented in [5] by using a low power -valued weight set generation algorithm combined with low-transition random test pattern generation (LT-RTPG). The idea in LT-RTPG [6] is to

Proceedings of the 2005 20th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT'05)
0-7695-2464-8/05 $20.00 © 2005 IEEE

reduce the number of transitions in the pseudo-random patterns. This is accomplished by ANDing together *k* outputs from a linear feedback register (LFSR) and using that to drive a toggle flip-flop which then loads the scan chain. This causes the transition probability to be smaller than .5, and it was shown in [6] that this has a relatively small impact on fault coverage. The scheme described in [5] combines LT-RTPG and the -valued weight set generation algorithm in [ ]. However, it has the same disadvantage as [ ] which is that it can result in large routing overhead due to the need to re-order the scan chains.

This paper describes a new BIST scheme that achieves low power without the need to re-order the scan chains. It uses a scan partitioning technique that exploits correlation in the weight sets to both simplify the weight logic as well as reduce the number of transitions in the scan chains. A methodology for designing the decoding logic is described which takes advantage of the scan partitioning. The scan cells in each partition can be ordered in any way desired (e.g., to minimize routing overhead). Consequently, the proposed scheme can be easily implemented in the standard design flow used in industry. Section 2 presents the details of the proposed scheme. Section describes the architecture that is used. Section reports experimental results, and Section 5 concludes this paper.

2. Proposed Scheme

The proposed scheme is composed of two parts: random test for the easy-to-detect faults and weighted random test for r.p.r. faults. The random test is performed by LT-RTPG as proposed in [6] to reduce power consumption. The important aspect of the proposed scheme is the weighted random test since it determines the hardware overhead for the -weight decoder and the total power consumption. The key concept in the proposed scheme is to partition the scan cells in a way that entire scan chains can be weighted to a uniform weight instead of individual scan cells each having their own weight. We define two types of scan chains: "*uniform scan*" and "*non-uniform scan*". A *uniform scan* is defined as a scan chain in which all scan cells have the same weight in each weight set which will be referred to as its "*scan weight*". A *non-uniform scan* is defined as a scan chain in which each scan cell has an individual weight as is the case for conventional weighting. The goal is to maximize the number of the uniform scans so that power consumption and hardware overhead are minimized. Section .1 describes the -valued weight set generation algorithm that was used in the proposed work. The scan chain partitioning algorithm is introduced in Section .2. Section . explains the -weight decoder minimization process.

2.1. 3-valued weight set generation

In a -valued weight scheme, there are only weights, , 1 and *r* (random). '*r*' means that corresponding bit position is not weighted, but determined pseudo-randomly. Because there are only three kinds of weights, the hardware overhead can be reduced significantly compared with conventional weighted random test [ ]. The whole process of weight set generation is shown in Fig. 1. First of all, a certain number of random patterns are applied to a circuit and the detected faults are dropped. The undetected faults after random pattern test are the r.p.r. faults that are targeted by weighted random test. A -valued weight set is then generated based on the deterministic test set as was proposed in [ ]. Consider an example where T = { 11x, xx, x11, x1x } is a set of deterministic test cubes. A weight set is generated so that the largest number of the undetected faults can be detected. In the other words, the weight set that can cover the largest number of the deterministic test cubes is generated. In this example, ' *r*1*r*' is generated based on the deterministic test set. The weight set, ' *r*1*r*' means that the first and the third input

bit position are weighted to and 1 respectively and the second and the fourth input bit position are generated pseudo-randomly. Note that the number of the bit positions that are weighted to '*r*', that is, determined pseudo-randomly, is limited by the number of patterns applied with one weight set. The more patterns that are applied per weight set, the more '*r*' positions there can be since the probability of covering each test cube is increased with more patterns. In this small example, if the maximum number of '*r*' positions was limited to 1 then any single weight set could not cover all of the deterministic test cubes (two '*r*' positions are required for that). Each weight set ' *r*11', ' 11*r*' or ' *r*1 ' can cover only three deterministic test cubes (not all four). After a weight set is generated, fault simulation with the weight set is performed and the detected faults are dropped. The final step is to check if all faults are detected or not. If all of the r.p.r. faults are detected, the -valued weight set generation is completed. Otherwise, another weight set is generated until the set of the undetected faults is empty.

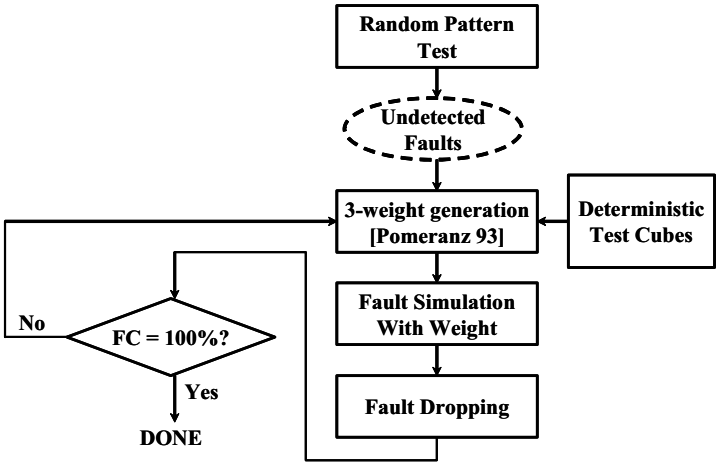

Figure 1. -valued weight set generation

2.2. Scan chain partitioning

In conventional -valued weighted random test, each individual scan cell is weighted to 1, , or *r*. This can result in a large hardware overhead for the -weight decoder. In the proposed work, careful partitioning of the scan chains is done so that all of scan cells in many of the scan chains are uniformly weighted to the same weight. This reduces the hardware overhead for the -weight decoder significantly and also reduces the power consumption during scan shifting. The actual order of the scan cells within each partition doesn't matter and hence they can be ordered in the conventional manner to minimize routing.

The proposed scan partitioning technique is formulated as a minimum clique covering problem. Note that in the remainder of the paper, a "weight cube" is defined as the set of weight assignments for a scan cell where each bit position in the weight cube corresponds to the weight assignment in a particular weight set (, 1, R, or x). An 'x' is a don't care where the weight assignment can be anything with no impact on fault coverage. This is illustrated in Fig. 2 where there are 9 scan cells and weight sets. The weight cube for scan cell *s2* is *x0xx* as the only requirement for *s2* is that it be weighted to in weight set 2. The proposed scan partitioning procedure has four steps which are described below:

Step 1: Prune and group scan cells

The first step involves pruning and grouping the scan cells to simplify the graph constructed in Step 2. All "don't care cells" which are scan cells that have no specified

weight in any weight set are removed from the set of scan cells since they need not be considered. Then the scan cells are grouped together where if all the specified bits of any weight cube for a scan cell is a subset of the specified bits of a weight cube for another scan cell, those two scan cells are put into the same group. Step 1 is illustrated in Fig. 2 where there are weight sets and the number of scan cells is 9. In this example, *s1* and *s4* are don't care cells which are removed from the set of scan cells. Also, the specified bit in the weight cube for *s7*, xxx1, is a subset the specified bits in the weight cube for *s3*, 1xx1, and thus *s3* and *s7* are placed into the same group, *g1*. The end result of Step 1 is a set of scan cell groups (as shown on the right of Fig. 2).

	s1	s2	s3	s4	s5	s6	s7	s8	s9
w1	x	x	1	x	0	1	x	1	0
w2	x	0	x	x	1	x	x	R	x
w3	x	x	x	x	x	1	x	R	1
w4	x	x	1	x	R	x	1	x	x

Weight cube for s2

Group table

G	Weight cube	Scan cells
g1	1xx1	s3 s7
g2	x0xx	s2
g3	1x1x	s6
g4	0x1x	s9
g5	01xR	s5
g6	1RRx	s8

Figure 2. Step 1: Initialization

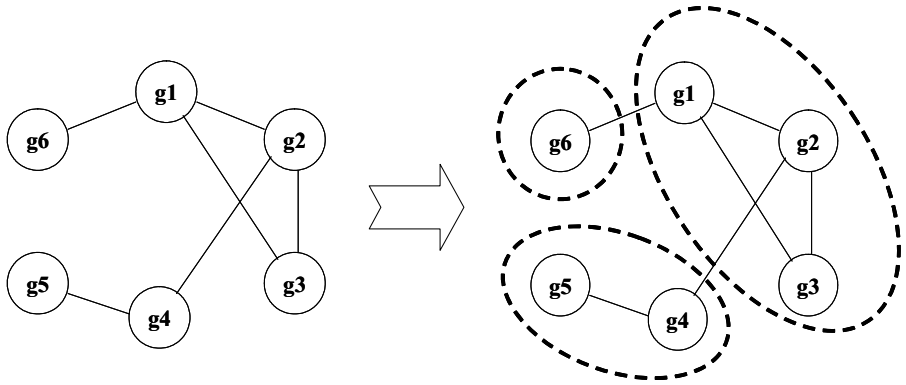

Figure 3. Step 2, : Minimum clique covering

Step 2: Construct compatibility graph

The second step involves constructing a compatibility graph in which each vertex in the graph corresponds to a group formed in Step 1 and an edge is placed between two groups if they are compatible. Two groups are said to be compatible if they do not conflict in any specified bit position (1, , and R). In other words, for each bit position in which both groups are specified, they must match. In the example in Fig. 2, the weight cube for *g1*, 1xx1, is compatible with the weight cube for *g6*, 1rrx. Therefore, an edge exists between *g1* and *g6* in Figure .

Step : Find minimum clique cover

A clique is a complete sub-graph and corresponds to a set of scan cells that can be used to construct a uniform scan chain since their weight assignment in all weight sets are

compatible. In the example in Fig. , *g1*, *g2* and *g3* compose of a complete sub-graph. Therefore, *g1*, *g2* and *g3* can be in one clique. The minimum clique cover for the compatibility graph corresponds to a partitioning of the scan cells that allows the construction of a maximal set of uniform scan chains. The right part of Figure shows the solution of the minimum clique covering problem. In this example, there are cliques that cover all of the vertices in the graph. Clique 1 includes *g1*, *g2* and *g3*, that is, four scan cells, *s2*, *s3*, *s6* and *s7*, clique 2 includes *g4* and *g* , that is, two scan cells, *s* and *s9*, and clique includes *g6*, that is, one scan cell, *s8*. The scan cells in each clique can be weighted by the same weight value in each weight set.

Note that there can be more than one minimum clique cover. Note also that finding a minimum clique cover is an NP-complete problem, however, there are good heuristic algorithms for it (see [7] for more details). A greedy algorithm was used in our experiments.

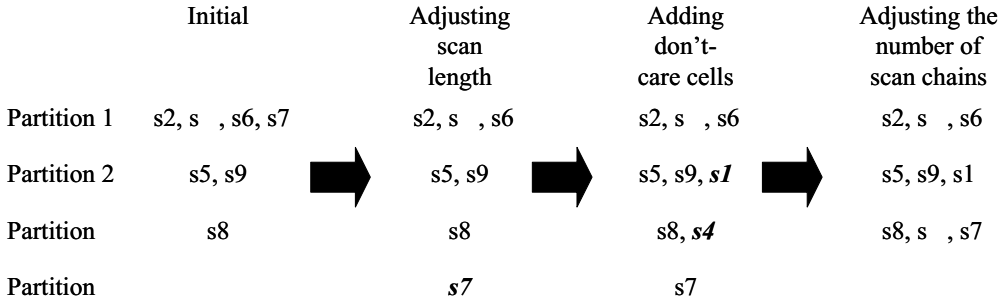

Figure 4. Step : Adjusting scan length and the number of scan chains

Step : Form scan chains

The final step is to form the scan chains from the partitions obtained in the minimum clique cover. Given a certain fixed scan length, the partitions in which the number of scan cells is larger than the scan length are divided. Ideally, it would be best if each scan chain was formed using scan cells from a single partition so that it can be a uniform scan. However, it is unlikely that the number of scan cells in each partition will be a multiple of the scan length. However, there are two degrees of freedom that can be used. The first is that some scan cells may be compatible with more than one partition and thus can be moved between partitions to help balance things out. The second degree of freedom is that the don't care cells which were pruned out in step 1 can now be added back in to any of the partitions (since they are don't cares) to help balance things out. Any scan chains that cannot be made equal to the scan length using these two degrees of freedom must be merged together to form non-uniform scan chains.

Consider the example in Fig. where the scan length is , the number of scan cells in partition 1 is , which is larger than the scan length. Therefore, one of scan cells in the partition 1 should be moved to another partition if possible without destroying the property of a clique. Otherwise, a new partition is added. In Fig. , partition is added, and *s7* is movedto the partition . After this, the number of scan cells that each partition has is less than or equal to the scan length. Any scan chain that has scan cells at this point (which is the same as the scan length) is a uniform scan. Therefore, partition 1 is a uniform scan. Next, the don't care cells are added to make as many additional uniform scans as possible. In the example in Fig. , *s1* is added to partition 2, and *s4* is added to partition , and thus partition 2 is made a uniform scan. Finally, partition is merged into partition to form the last scan chain. Because two different partitions are merged, partition is not a clique anymore and thus forms a non-uniform scan. The end result is

that there are two uniform scans, partition 1 and partition 2, and one non-uniform scan, partition . All the scan cells in each of the uniform scans are weighted by the same scan weight in each weight set. This will reduce power consumption and also minimize the hardware overhead for the -weight decoder. The non-uniform scan is weighted by conventional weighting where each scan cell has its own weight.

2.3. 3-weight decoder minimization

	Scan weight 1	Scan weight 2	Scan weight 3	Scan weight 4	Scan weight 5
w1	1	X	X	1	0
w2	1	0	X	X	0
w3	X	1	0	0	X
w4	X	0	1	1	X
w5	X	1	X	0	1

Figure 5. Decoder minimization

The decoding logic for the -valued weight sets can be minimized further by using compatibility of scan weights. If two scan chains have compatible scan weights across all weight sets, they can share the same decoding logic and weight logic. One way that this commonly happens is when two scan chains are formed from the same original partition. Figure 5 shows an example of decoder minimization. In this example, there are 5 weight sets and 5 uniform scans, that is, 5 scan weights per weight set. *Scan weight 1*, *scan weight 2,* and *scan weight 4* are mutually compatible because there is no conflict in any specified bit position. Therefore, *scan weight 1*, *scan weight 2,* and *scan weight 4* can be generated by the same decoding logic. In the same way, *scan weight 2* and *scan weight* can be generated by the same decoding logic. Therefore, only two separate decoding logic and weight logic (one AND gate and one OR gate) are required to handle all 5 scan chains, thereby reducing hardware overhead for the decoding logic.

3. Architecture

In this section, two different architectures are presented. One is the conventional fixed-length scan architecture, and the other is a variable-length scan architecture that allows better optimization for the proposed scheme as will be described. In the fixed-length scan architecture each scan chain has the same length, whereas in the variable-length scan architecture, the scan chains have variable lengths.

3.1. Fixed-length scan architecture

Figure 6 shows a fixed-length scan architecture. There are uniform scans and 1 non-uniform scan and each scan chain has the same scan length. The uniform scans are weighted by a scan weight decoder. Note that the input of the scan weight decoder is only the weight counter information since all the scan cells in a uniform scan are weighted by the same weight and thus the scan weight decoder does not require bit counter information. The non-uniform scan is weighted by conventional -valued weighted random testing.

Note that hardware overhead is primarily determined by the size of the -weight decoder for non-uniform scans. The size of the -weight decoder for non-uniform scans is much larger than the size of the scan weight decoder for uniform scans. The

conventional -weight decoder requires bit counter information as well as the weight counter information, thereby making the decoding logic much more complex. Therefore, the main objective in scan partitioning is to minimize the number of non-uniform scans.

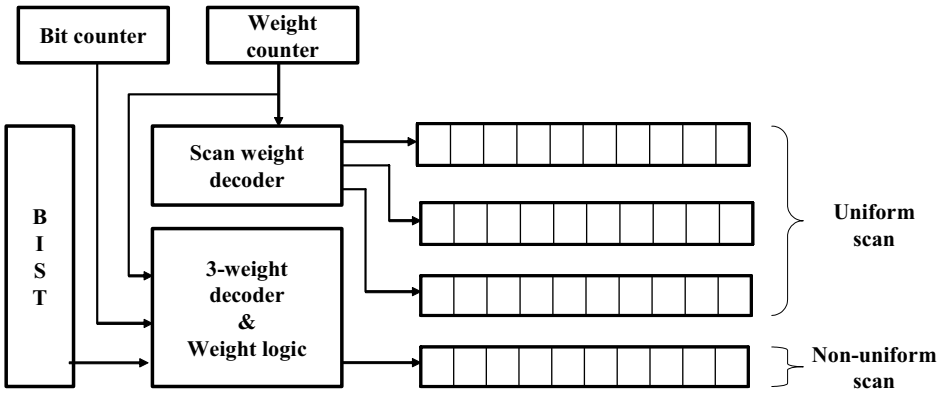

Figure 6. Architecture

3.2. Variable-length scan architecture

A variable-length scan architecture is proposed to reduce the number and length of non-uniform scans since the size of the -weight decoder for non-uniform scans depends on this. The final step, Step , in the scan partitioning technique proposed in Sec. 2.2 can be modified for variable-length scan chains to reduce the number of non-uniform scans and the length of non-uniform scans. The drawback of a variable-length scan architecture is that the test time increases because the number of clock cycles required to shift in each pattern is determined by the longest scan chain, and the length of the longest scan chain increases in the variable-length scan architecture. However, typically in BIST, test time is not as an important issue as hardware overhead and power consumption.

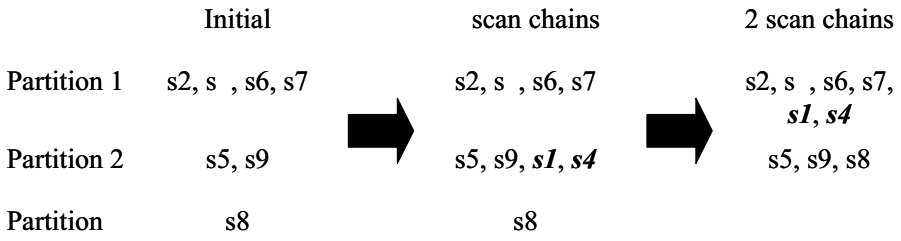

Figure 7. Step : Adjustment in variable-length scan architecture

Step in Sec. 2.2 is modified for variable-length scan in the following way. Instead of a constraint on the length of each scan chain, there is a constraint on the number of scan chains. If the number of partitions is less than the number of scan chains, then the largest partitions can be divided to minimize the maximum scan length. If the number of partitions is larger than the number of scan chains, then the smallest partitions can be merged to create a non-uniform scan. Once the number of partitions is the same as the number of scan chains, then the don't care cells are added to make the lengths of the uniform scans as balanced as possible. In the example in Fig. 7, the don't care cells, *s1* and *s4,* are added to *partition 2*.

4. Experimental Results

The proposed scheme has been applied to the 5 largest ISCAS89 circuits [8]. Table 1 shows the number of random patterns generated by LT-RTPG and the size of LFSR used in the proposed work. The number of weight sets required to achieve 1 % fault coverage for both the proposed method and the method described in [5] are also shown in Table 1. Note that they are similar.

Table 1. Results using LT-RTPG

Circuit	Number of Patterns	Serial-fixing WRBIST [5]	Proposed	
		Num. Weight Sets	LFSR size	Num. Weight Sets
s92	1 2 72	1	2	15
s1 2 7	655 6	2	2	2
s1585	1 1 72	12	2	8
s 8 17	1 1 72	2	6	19
s 858	655 6	12	6	

Table 2. Results for proposed method

Circuit	Num W.S.	Num Scan	Fixed-length scan architecture				Variable-length scan architecture					
			Num U.S.	Num N.S.	Area Overhead	Trans. Reduc.	Num. U.S.	Num. N.S.	Area Overhead	Overhead Reduc.	Trans. Reduc.	Test Time
s92	15	16	11	5	59	68.7%	1		11.5	1 . %	82.2 %	.8%
		2	22	1	7.5	68.7%	27	5	77.5	1 .6%	8 .6%	5%
s1 2 7	2	16	1	2	8 .5	87.5%	15	1	66	21.8%	9 .7%	29.5%
		2	29		67.5	9 .6%	1	1	7.5	29.6%	96.8%	6. %
s1585	8	16	1		2 1.5	81.2%	1	2	2 1.5	16.5%	87.1%	5.8%
		2	26	6	2 2	81.2%	28		179.5	11.1%	87.1%	5. %
s 8 17	19	2	2	9	1175.5	71.8%	26	6	1 2 .5	1 .1%	8 . %	. %
		5	8	12	981.5	76. %	2	8	88	9.9%	79. %	. %
s 858		2		2	197.5	9 .7%		2	1	27.5%	9 . 1%	%
		5	7		1 9	9 . %	8	2	128.5	1 .7%	95. %	%

Table 2 shows the hardware overhead and power consumption resulting from the proposed work. The third column shows the number of scan chains. The fourth and the fifth column, and the eighth and ninth column show the number of uniform scans (U.S.) and the number of non-uniform scans (N.S.) respectively. Note that in all of the circuits, the number of uniform scans is much larger than the number of non-uniform scans, and that means the proposed scan partitioning technique can reduce power consumption and hardware overhead effectively. The hardware overhead shown in the sixth and the tenth column is expressed as gate equivalents which were computed as .5n for an n-input NAND/NOR gate and .5 for an inverter. The -weight decoding logic was obtained by running SIS [9] for a 2-level circuit implementation. A comparison between the hardware overhead in a fixed-length scan architecture and the hardware overhead in a variable-length scan architecture is shown in the eleventh column. Note that the hardware overhead in the variable-length scan architecture is 9.9% ~ 29.6% smaller than the hardware overhead in fixed-length scan architecture. Reduction in the number of transitions in the scan chains are shown in the seventh and twelfth column. The last column shows the drawback of variable-length scan architecture, test time increase. The test time in variable-length scan architecture increased 2 . % ~ 67.5% compared with the test time in the fixed-length scan architecture.

In Table , the experimental results in [5] are compared with the results in the proposed work. In all circuits except for one, *s9234*, both the hardware overhead and the amount of power consumed in the proposed work is smaller than the hardware overhead and the power consumed

in [5]. Note that in the proposed method, no constraints are placed on the scan order, thus the routing complexity for the proposed method is much less. The number of transitions in the scan chains is calculated in the manner described in [5]. 512 consecutive patterns generated by LT-RTPG and the first 128 patterns generated by each weight set are used to count the number of transitions.

Table 3. Comparison with Serial-fixing WRBIST [5]

Circuit Name	Serial-fixing WRBIST [5]			Proposed				
				Fixed-Length Scan			Variable-Length Scan	
	Num W.S.	Overhead (GE)	Transition Reduction	Num W.S.	Overhead (GE)	Transition Reduction	Overhead (GE)	Transition Reduction
s92	1	21 .5	6%	15	7.5	5 .7%	77.5	56.8%
s1 2 7	2	6	27%	2	67.5	58.1%	7.5	59. %
s1585	12	2 7.5	7%	8	2 2	56. %	179.5	57. %
s 8 17	2	1 1.5	7%	19	981.5	55.2%	91	55.8%
s 859	12	671	5%		1 9	58.8%	128.5	59. %

The experimental results described in this section show that the proposed scheme can be an effective solution for two critical problems in BIST, power consumption and overhead to detect r.p.r. faults. The proposed scheme can reduce the hardware overhead to detect r.p.r. faults and the amount of power consumed during test.

5. Conclusions

In this paper, a technique to reduce power consumption and hardware overhead in BIST is proposed. It uses scan partitioning, but does not place any constraints on scan order thereby allowing the scan order to be selected to minimize routing complexity. The proposed method is simple and can be easily incorporated in the standard design flows used in industry.

Acknowledgements

This material is based on work supported in part by the Intel Corporation and in part by the National Science Foundation under Grant No. CCR- 62 8.

References

[1] E.B. Eichelberger, and E. Lindbloom, "Random-Pattern Coverage Enhancement and Diagnosis for LSSD Logic Self-Test," *IBM Journal of Research & Development*, Vol. 27, No. , pp. 265-272, 198

[2] H.D. Shnurnann, E. Lindbloom, and R.G. Carpenter, "The Weighted Random Test-Pattern Generator," *IEEE Tran. On Computers*, pp. 695-7 , 1975

[ ] I. Pomeranz, and S.M. Reddy, " -Weight Pseudo-Random Test Generation Based on a Deterministic Test Set", *Proc. of International Conf. On VLSI Design*, pp. 1 8-15 , 1992

[ ] S. Wang, "Low Hardware Overhead Scan Based -Weight Weighted Random BIST", *Proc, of International Test Conference*, pp. 868-877, 2 1

[5] S. Wang, "Generation of Low Power Dissipation and High Fault Coverage Patterns For Scan-Based BIST", *Proc. of International Test Conference*, pp. 8 -8 , 2 2

[6] S. Wang and S.K. Gupta, "LT-RTPG: A New Test-Per-Scan BIST TPG for Low Heat Dissipation", *Proc. of International Test Conference*, pp. 85-9 , 1999.

[7] T. Cormen, C. Leiserson, and R. Rivet, "Introduction to Algorithms,", *The MIT Press*, 1997

[8] F.D. Brglez, and K. Kozminski, "Combinational Profiles of Sequential Benchmark Circuits," *Proc. Of International Symposium on Circuits and Systems*, pp. 1929-19 , 1989

[9] E.M. Sentovich, K.J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saklanha, H. Savoj, P.R. Stephan, R.K. Brayton, and A. Sangiovanni-Vincentelli, "SIS: A System for Sequential Circuit Synthesis", *University of California-Electronics Research Laboratory Memorandum No. UCB/ERL M92/41*, 1992