A Unified Solution to Scan Test Volume, Time, and Power Minimization

Zhen Chen
Tsinghua University, Beijing, z-chen07@mails.tsinghua.edu.cn

Sharad C. Seth
University of Nebraska - Lincoln, seth@cse.unl.edu

Dong Xiang
Tsinghua University,, dxiang@tsinghua.edu.cn

Bhargab B. Bhattacharya
Indian Statistical Institute, Kolkata, India, bhargab@isical.ac.in

A Unified Solution to Scan Test Volume, Time, and Power Minimization

Zhen Chen
Dept. of Comp. Sci. and Techn.,
Tsinghua University,
Beijing 100084, China
z-chen07@mails.tsinghua.edu.cn

Sharad Seth
Computer Science and Engineering,
University of Nebraska-Lincoln,
Lincoln NE 68588-0115, U.S.A.
seth@cse.unl.edu

Dong Xiang
School of Software,
Tsinghua University
Beijing 100084, P. R. China
dxiang@tsinghua.edu.cn

Bhargab B. Bhattacharya
Indian Statistical Institute,
Kolkata, India
bhargab@isical.ac.in

Abstract —The double-tree scan-path architecture, originally proposed for low test power, is adapted to simultaneously reduce the test application time and test data volume under external testing. Experimental results show significant performance improvements over other existing scan architectures.

Keywords: Test power minimization, Test time reduction, Test data reduction, Nonlinear scan

I. Introduction

Recent trends in manufacturing have spurred research on reduction of test power, test volume, and test-application time. Excessive test power can lead to unreliable operation, yield loss, device burnout, or reduced battery life, whereas increased test volume and test time contribute to the cost of testing.

For STUMPS-like scan architecture [1], with multiple parallel scan chains, it can be shown that as the design size is increased, the test power per cycle grows linearly, while test energy, test volume, and test time all grow at an even higher rate [2]. As the traditional scan schemes do not scale up well for today's complex chips, containing these show-stopper trends through other means has been an active area of research in recent years.

A large body of literature exists on reducing test data volume. Since lower test data volume leads to shorter test time, reducing test data volume is critical to low cost testing. In order to reduce the numbers of test vectors, static and dynamic methods are proposed to exploit the large fraction of don't care bits [3, 6] in the ATPG test cubes. In order to reduce the data volume stored in the tester memory, various compression techniques have been proposed. These fall broadly into three categories [7]: (1) Code-based schemes [8, **?**], (2) Linear-decompression-based schemes [2, 9, 10, 11], and (3) Broadcast-scan-based schemes [12]. Commercial tools based on the second and third kinds of methods are available since they are more efficient in compressing test vectors for industrial circuits with a large fraction of don't care bits. Broadcast-scan-based methods have the advantage of incorporating the constraints for the decompressor into ATPG to produce encodable test cubes, over linear-decompression-based schemes. However, broadcast scan has less encoding flexibility because certain groups of scan cells receive identical values. In this paper, we use a broadcast-scan-based scheme that relaxes this constraint.

Further, test power is also an important metric of test performance [13]. Low power methods are mainly DFT-based [14] or pattern-based [15]. However, many data compression schemes are not compatible with the existing low power techniques. For example, For example, Illinois Scan [11], reconfigurable Illinois scan [4], dynamically reconfigurable shared scan architecture [5],and EDT [2] can efficiently reduce test data volume and test time, but have little effect on test power. Indeed, EDT effectively randomizes the don't care bits in the test cubes, thereby resulting in scan-shift switching activity that is close to 50%. A low-power extension of EDT [16] intelligently maps scan slices to self-loop states of the ring generator to minimize transitions during scan shifts. However, the resulting improvement in test power is achieved at some sacrifice in data-compression efficiency.

In this paper, we present a unified solution to improve the three metrics simultaneously, based on the nonlinear, *double-tree scan* (DTS) architecture [17]. That work focused on scan-load power consumption, reducing both the peak load power and energy by the factor $N/logN$, as compared to a linear scan chain of length N. DTS does not address the capture power issue but is compatible with schemes, e.g. [18], proposed to solve it. The DTS architecture, however, also offers opportunities for dynamically changing the mapping from input test-bit stream to the scan cells and loading the same input stream in multiple ways. These features allow us to weaken the constraint of broadcast-based compression and achieve significant reductions in test volume and test time, on top of the inherent power savings of DTS.

The rest of the paper is organized as follows. Section II presents the background and basic theory for our proposed method. Section III gives the overall idea. The test scheme and detailed strategies are proposed in Section IV. Experimental result are given in Section V. Section VI concludes the paper.

II. Background

In this section we briefly describe the prior work that forms the basis for this paper. Specifically, we cover the salient aspects of the following: the double-tree scan architecture [17], and the no-common-successor relation between

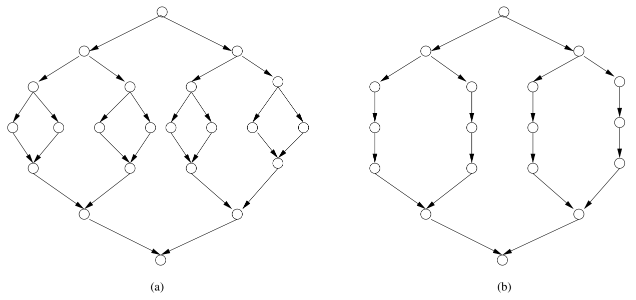

Figure 1: The structure of (a) full DTS(3) with 22 nodes and (b) a partial DTS with 18 nodes.

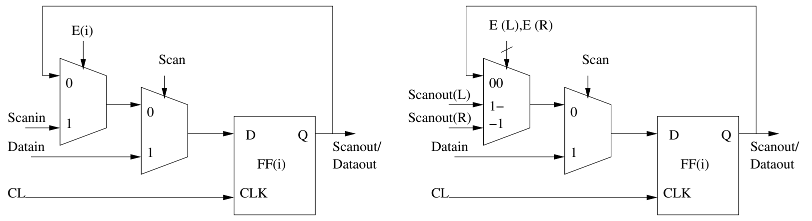

Figure 2: Scan cells used in the top part (a) and bottom part (b) of the DTS to avoid clock routing.

scan cells that is used in the clustering algorithm for assigning cells to different scan chains.

A. Double-Tree Scan

A complete binary tree of level k (considering the root at level 0) consists of 2^k leaf nodes and $(2^k - 1)$ internal nodes. The proposed scan structure resembles two complete k-level binary trees whose leaf nodes are merged pair-wise. Thus, a full *double-tree* DTS(k) consists of $N = (2^k - 1 + 2^k + 2^k - 1) = 3 * 2^k - 2$ nodes. Each node of the tree represents a scan flip-flop. All edges in the tree are directed from top to bottom. A directed edge (i, j) in the DTS indicates that $Q(i)$, i.e., the Q-output of the flip-flop i, drives $D(j)$, the D-input of the flip-flop j. For each node with in-degree 2 (i.e., a merge node) in the bottom half of the DTS, a 2-1 MUX is needed to select the predecessor flip-flop during the scan-load operation. Figure 1(a) shows a full DTS(3) with 22 nodes. By pruning a full DTS, it is possible to obtain a partial DTS with any number of nodes; Figure 1(b) shows an 18-node partial DTS obtained by pruning DTS(3). The forking (merging) nodes require the addition of a 1→2 DEMUX (2→1 MUX) to the corresponding scan cells (see Fig. 2), while the middle part can be built from standard scan cells. Both trees are balanced because each path from the source node to the sink node has the same length.

DTS can serve as a direct replacement of a scan chain by connecting the topmost (source) node to the scan-in signal and the bottommost (sink) node to the scan-out signal. The full DTS(k) has 2^k overlapping scan paths, each of length $(2k + 1)$, from the source to the sink. The control unit to load and unload the DTS repeatedly selects a source-to-sink scan-shift path in each clock cycle, so that, externally the DTS is indistinguishable from a serial scan chain of length N, where N is the number of FFs in the DTS. However, in each clock cycle, only $O(logN)$ FFs are enabled, thus providing $O(N/logN)$ improvement in test power per cycle. Note that unlike the schemes that use linear chains and

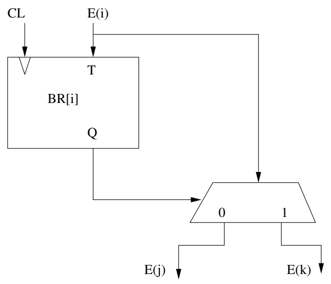

Figure 3: Extra logic added to implement the distributed control for path selection.

must minimize the chain length to minimize power consumption during scan loads, we can afford to implement larger DTS structures because of its logarithmic scan-shift depth. In other words, the design tradeoff between the number of chains and chain-length is very different for, and favorable to, DTS-implemented chains than linear chains.

A particularly simple control scheme for DTS is the breadth-first load [17], which involves selection of source-to-sink paths in a cyclic fashion until the tree is fully loaded. This idea can be realized by a counter in a centralized control, where each counter bit connects to the enable signals (E(i) in Figure 2) of scan cells at the same level in the top part of the DTS. The idea can also be realized by a distributed control by adding a toggling *branch* FF (see Fig. 3) to each scan cell in the top part to route the enable signal along the selected path. The branching FF BR[i] determines whether the next scan-shift path going through this cell will select the left or the right node of the DTS. The branch FFs can as well be inserted in the bottom part and the enable signal is routed from bottom to top so as to avoid the race condition between the data and the enable signal that might otherwise occur in the alternate design [19].

Results of our preliminary investigations on achievable power savings with DTS and its impact on the area and routing of a design became recently available [19]. A fully synchronous test chip consisting of a $sinc^3$ decimation filter with 188 flip-flops and 8000 gates was designed and simulated while clocking in a random test pattern in both the conventional scan path and the DTS scan path. The total number of logic level toggles in the DTS design decreased by a factor of 8 over the standard scan.

For measuring the hardware overhead of the DTS architecture, a RISC CPU, compatible with TI MSP 430 microcontroller core, and several ITC99 benchmarks were synthesized from the RTL-VHDL, in four different versions: without scan, with standard scan, with DTS using centralized control, and with DTS using distributed control. The designs were mapped to the IBM 0,18 micron process. The results show a centralized DTS area overhead of 16% over standard scan for most circuits. The distributed control overhead was another 15% on top of that. In all cases, it was possible to use the same standard cell utilization (over 90%) for all versions of the design, i.e. routing was not an issue.

The area overhead for both the centralized and distributed control schemes can be reduced significantly by including more than one scan FF per DTS node, with commensurate reduction in power savings. The area overhead of the distributed scheme can be further reduced by sharing of the branch FFs between two or more scan FFs at the same level of the DTS tree.

B. *No-Common-Successor Relation*

The effectiveness of the broadcast test mode in ILS depends strongly on the quality of the generated test patterns. Another approach [20], which we follow here, attempts to break the correlation between the scan cells based on the circuit topology. It makes use of the no-common-successor (NCS) binary relation between scan cells. Two scan cells are related by NCS if they cannot reach a common node in the circuit through combinational paths. Other researchers have used the complementary relation in defining a reconfigurable scan-chain architecture [21]. The usefulness of the NCS relation in minimizing correlation arises from the following lemma [20].

Lemma 1 *In a full scan circuit, if two scan FFs have the NCS relation, at most one of them needs to be assigned a binary value, in the generated test cube, to detect a single stuck at fault.*

Because of the lemma, it is clear that two scan FFs in NCS relation can be assigned the same binary value without any loss of fault coverage.

III. Main idea

Unlike the linear scan chain, there are multiple paths from the source to the sink in the DTS. We exploit this property, in the context of loading of parallel scan chains, where each chain is replaced by a DTS structure. Specifically, during the broadcast mode of Illinois Scan, this property allows the broadcast pattern to be permuted (see Section IV.A for more details) as it is loaded into different scan chains, thus largely overcoming the two major shortcomings of Illinois Scan: (1) identical bits loaded into the same bit positions of different scan chains, and (2) strong dependence of performance on

Table 1: Leftmost-first and rightmost-first loading of the DTS shown in Figure 4.

Scan FF	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5
Leftmost-first	b5	b3	b4	b1	b2	b0
Rightmost-first	b5	b4	b3	b2	b1	b0

scan-chain configuration. Further, simply by changing the relative permutations of load sequence between two chains, the same pattern can be repeatedly applied to provide additional fault coverage with negligible increase in the test-stimuli volume: with k-bits/scan-chain overhead, it is possible to specify any one of 2^k different permutations. Our results show that even two permutations of a pattern provide a dramatic increase in the achievable fault coverage.

IV. Our scheme

Our scheme depends on modifying both the DTS and ILS in significant ways that are described in this section.

A. *Input Permutations with DTS*

Earlier, we indicated that the DTS with N nodes can be loaded in a breadth-first fashion by cycling through its paths for N cycles [17]. The specific order in which the paths are cycled determines the mapping from index of the bit in the input bit stream to the index of the FF in which the bit is loaded. Therefore, simply by changing the order in which the paths are cycled, we can load a pattern with different permuted bits.

Consider the breadth-first loading of partial DTS of six nodes shown in Figure 4 for illustration. This DTS has only two paths, that can be selected by specifying one path bit. Let us assume that the path bit value 0 denotes the left shift path and 1 denotes the right shift path. If we assume the initial path bit value to be 0 and the input bit-stream to be <b5, b4, b3, b2, b1, b0>, where $b0$ arrives first, then the resulting mapping will be as shown in Table 1 under the *leftmost-first*. Similarly, when the path bit value is initialized to 1, the resulting mapping will be as shown under the *rightmost-first* in Table 1. It is noteworthy that except for the source and sink FFs, all other FFs receive different bits in the input stream. For the leftmost-first and rightmost-first loads, this observation is generally true for any balanced DTS structure. We call a mapping from input bit-stream to scan FFs an *input permutation*. For the example DTS of six nodes, we can realize two different input permutations. Similarly, in a full DTS(k), there are 2^k different paths and we can realize 2^k input permutations, simply by initializing the path counter to different states, and another 2^k different permutations by running the counter in reverse.

This feature of DTS is potentially very useful in test-volume reduction because it allows reuse of a given pattern in 2^k different ways by specifying only k bits, where k is logarithmic in the size of the DTS. For example, for DTS(10) with 3,070 FFs, 1024 different permutations can be realized by specifying just 10 bits.

B. *Scan-Cell Clustering*

Because of the lemma in Section II, it is clear that two scan FFs in NCS relation can be assigned the same binary

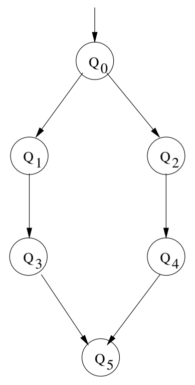

Figure 4: The balanced scan tree with six nodes.

value without any loss of fault coverage. Although the NCS relation is exact and very useful, using the strict NCS condition for each group is prohibitive for some circuit with large input cones and output cones. Instead, we relax the NCS condition to construct scan chains. For the source nodes and sink nodes, we use the NCS condition to break the correlation between them. For nodes at other levels, the NCS condition needs not hold strictly because the correlation between them can be broken by loading DTS with different permutations.

The scan-cell clustering algorithm works on the NCS relation between scan FFs, represented as a graph. The goal of the algorithm is to form groups of m scan FFs, where m is the number of DTS chains, such that the m flip-flops in a group are distributed among the m DTS chains by placing one FF at the same level of each DTS tree. To form each group, the following steps are taken:

1 Initialize a new group G to a null set.

2 Find the as-yet unprocessed node in the NCS graph with the highest degree, and use it as the seed of a new cluster S.

3 Repeatedly choose a node satisfying the NCS condition with all elements in S, until no such node can be found or the size of S is m. The resulting set S defines a cluster.

4 Put S into G. If the size of G is smaller than m, form additional clusters and put them into G until G becomes a group of size m.

After the groups are formed, we first choose two groups with the fewest clusters in them, and assign one group to all the source nodes the other group to all the sink nodes of the m DTS chains. Next, the DTS nodes connected to the source and the sink nodes are assigned in the same fashion. Proceeding in this way, all the interior levels of the DTS are filled from the boundary to the middle of the DTS chains. For the scan flip-flops in a group, it does not matter which specific chains they are assigned to, keeping all the degrees of freedom for placement and routing.

C. Test Generation and Application

We assume that we have m balanced scan chains, each of length l and each implemented as a full or balanced-partial DTS (see Figure 1(a) and 1(b)). Further, we assume that no output data compression is being performed so that there are m input channels and m output channels, for a total of $2m$ scan channels. As the cost of ATE is directly proportional to the number of scan channels, we assume m to be small, in the range of 2 to 32.

The test generation procedure is as follows:

1. Partition the scan cells into groups, using the clustering algorithm described in the Section IV.B. Then, modify the test bench to connect the scan cells in the same group to a common input.
2. Run the constrained ATPG on the circuit with the constraint that the scan flip-flops in the same group must be assigned the same value in the test generation process. For each pattern generated, use Step 3 to do fault simulation.
3. Fault-simulate with different permutations. (In the example below, we assume there are 8 chains.)
   (a) The test pattern with the loading permutation 00000000 (i.e. leftmost-first load).
   (b) The test pattern in which the loading of the first half of the chains is changed, that is, the loading permutation is 11110000.
   (c) The test pattern in which the loading of the last half of the chains is changed, that is, the loading permutation is 00001111.

   If additional faults are detected in any of the above sub-steps, the permutation is recorded.
4. Run another constrained ATPG, in which grouping of scan FFs is changed from Step (2) to realize a different permutation of the original pattern, which can be described with reference to Figure 5. We note that, except for the groups of source nodes and sink nodes, the remaining groups of m nodes in a DTS can be paired according the permuted bit values they receive in the leftmost-first and rightmost-first loads, e.g. the group of nodes with hexagon shape (flip-flops 2 and 6) and the group of nodes with square shape (flip-flops 3 and 7) form a pair. If the left tree loads data in leftmost-first while the right tree in rightmost-first, the group pairs in Fig. 5 become (2, 7) and (3, 6) from the original pairs (2, 6) and (3, 7). After running the ATPG, we repeat the three sub-steps of Step 3, for the pattern obtained.

Let us give more explanation of step 4. In Figure 5, let us assume that scan cells marked with identical shape are in the same group, i.e. cells 1 and 5, 2 and 6, 3 and 7, 4 and 8. Assume, the detection of a fault requires the values of scan cells 2,3,6,7 to be 1,0,0,X. In the constrained ATPG in Step 2, the fault is redundant because scan cell 2 and scan cell 6 must be the same value. But this fault can be detected in the Step 4. In Step 4, scan cell 2 and 6 can have the different values if one tree loads leftmost-first and the other tree loads rightmost-first. In essence, Step 4 is also the realization of using a different permutations to improve the fault coverage.

Test application under the described scheme requires that when a test pattern is repeated, the repeat code also include an m-bit loading permutation vector that is used for initialization of each scan chain before the test pattern is loaded.

V. Experimental Results

The proposed scheme was implemented and run on ISCAS89 scan circuits to assess its performance. We use Atalanta as the basic ATPG tool to realize constrained ATPG

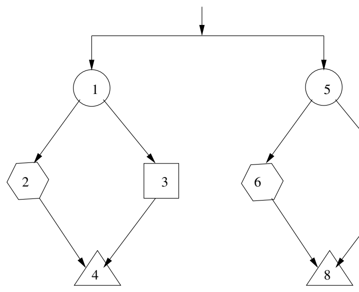

Figure 5: An example with two scan chains.

and compare results with Illinois Scan (ILS) in Table 2. In the experiment, the configuration of each chain in Illinois scan is the same as ours, therefore the comparison is fair.

Table 2 compares our method and ILS against the baseline of serial full scan. After the circuit name, the next four columns show the performance for serial full scan: "vec" is the number of test vectors, "data" and "cycles" denote, respectively, the total number of test bits and test clock cycles, and "power" is the average number of transitions in scan flip-flops during a clock cycle. The test power is estimated by the number of transitions in scan flip-flops instead of all the nodes because there is linear relationship between the two kinds of transitions [23]. The last six columns show the performance gains achieved by Illinois Scan and our scheme as ratios of corresponding measures over serial scan. These gains are shown as a function of the number of scan chains, shown in column six. In Illinois Scan, test-data reduction is achieved by connecting multiple scan chains to the same scan input in the broadcast mode, however, this introduces artificial redundant faults. In our method, such redundant faults can be avoided because the same positions of different chains may scan in different test data. In the table, the fault coverage of the three methods is the same. From the comparison, it can be concluded easily that our method is more efficient in reducing the test cost.

In Table 3, we show the effectiveness of input permutations in improving fault coverage and reducing test cost. The columns show fault coverage under the broadcast mode with and without the use of permutations. The column marked "Last stage" shows the result when we can run an arbitrary permutation permissible by DTS, in order to reach the maximum possible fault coverage by our method. The overhead of the last stage is that, for the scan chains constructed by DTS(k), every scan chain needs an extra k bits to specify permutation for a pattern. But the fault coverage is already high after the second stage employing permutations specified by just one bit per scan chain (column 6 and 7). So only a few patterns are needed in this stage. For example, for the last four circuits in the table, the last stage is not needed. From the data, we see that with the different-permutation strategies under the broadcast mode we can achieve very nearly the same fault coverage as serial scan (same as when the number of scan chains is low). The column "rpt" shows the average number of times a test vector is repeated, and the column "cbit" shows the average overhead of the of configuration bits as a fraction of the total number of bits in a test vector. The last three columns show the performance gains, as in Table 2, and the result shows that our method can reduce the test cost greatly. The test data, time and power reductions are different according to the number of scan chains, but, in all cases, the test cost reduction is significant compared to full scan. So the number of scan chains can be used as a tradeoff among the three test metrics.

We also compare our test data volume results against EDT (Table X of reference [2]) for the first six of the seven ISCAS'89 circuits shown in Table 2 above. The ratios of EDT test volume to ours for the six circuits are 1.80, 1.56, 1.19, 0.97, 1.19, and 0.98, respectively. That is, in four of the six cases we achieve substantially better results and in two cases EDT is slightly better.

VI. Conclusions

In this paper we show that in combination with an Illinois-Scan like test application scheme, DTS can yield significant improvements in three important metrics of interest to test engineers, related to the data volume, time and power consumption of tests. To achieve these benefits, we outlined control schemes for DTS, described a clustering algorithm to distribute scan cells between parallel scan chains, and proposed a test-application scheme that exploits both the ability of DTS to apply permuted test patterns and the repeat function of the ATE. Compared to linear scan, the DTS scan cells include incrementally more logic and extra fanin or fanout. We believe, the extra design cost of scan cells are well worth the price for the benefits that follow from its use.

References

[1] P. Bardell, W. McAnney and J. Savir, "Built-in Self-test for VLSI: Pseudorandom Techniques," John Wiley and Sons, New York, NY, 1987, pp. 354.

[2] J. Rajski, J. Tyszer, M. Kassab, and N. Mukherjee, "Embedded deterministic test," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 23, pp. 776-792, 2004.

[3] I. Hamzaoglu and J. H. Patel, "Test set compaction algorithms for combinational circuits," in IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 1998, pp. 283-289.

[4] A. R. Pandey and J. H. Patel, "Reconfiguration technique for reducing test time and test data volume in Illinois Scan Architecture based designs," in VLSI Test Symposium, 2002, pp. 9-15.

[5] N. Sitchinava, E. Gizdarski, S. Samaranayake, F. Neuveux, R. Kapur, and T. W. Williams, "Changing the scan enable during shift," in VLSI Test Symposium, 2004, pp. 73-78.

[6] B. Ayari and B. Kaminska, "A new dynamic test vector compaction for automatic test pattern generation," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 13, pp. 353-358, 1994.

[7] N. A. Touba, "Survey of Test Vector Compression Techniques", IEEE Design & Test, July, 2006, pp. 294-303.

[8] A. Jas, J. Ghosh-Dastidar, and N. A. Touba, "Scan vector compression/decompression using statistical coding," in IEEE VLSI Test Symposium (VTS), 1999, pp. 114-120.

[9] I. Bayraktaroglu and A. Orailoglu, "Decompression hardware determination for test volume and time reduction through unified test pattern compaction and compression," in IEEE VLSI Test Symposium, 2003, pp. 113-118.

[10] B. Koenemann, C. Barnhart, B. Keller, T. Snethen, O. Farnsworth, and D. Wheater, "A SmartBIST variant with guaranteed encoding," in Proceedings 10th Asian Test Symposium (ATS), 2001, pp. 325-330.

Table 2: Comparison of ILS and our method on test cost

name	Serial full scan					Performance-gain (ratio) over serial full scan					
						ILS (full scan/ILS)			Ours (full scan/our method)		
	vec	data	cycles	power	#chains	data	cycles	power	data	cycles	power
s5378	251	44929	45359	78	4	3.12	3.05	0.96	5.02	4.09	4.59
s9234.1	368	77648	78227	104	4	3.01	2.96	0.95	5.20	4.42	4.52
s13207.1	436	278168	279242	318	8	4.28	4.23	0.98	10.88	7.91	5.89
s15850.1	472	252048	253054	262	8	4.56	4.48	1.02	9.46	8.21	4.37
s38417	886	1449496	1452018	767	16	13.26	13.00	1.00	25.86	20.80	7.59
s38584.1	658	938308	940392	712	16	4.96	4.91	1.00	22.86	17.26	8.18
s35932	68	117504	119300	812	16	29.41	20.33	0.99	41.85	20.71	6.94

Table 3: Performance under different number of scan chains

name	coverage of full scan	#sc	coverage of broadcast		coverage of permutations		Last stage		data	cycle	power	rpt	cbits	Performance-gain (ratio) over full scan		
			vec	cov	vec	cov	vec	cov						data	cycle	power
s5378	99.131	2	252	99.163	218	99.16			19620	23202	10	1.17	0.003	2.29	1.95	7.80
		4	236	97.784	199	99.14			8955	11081	17	1.21	0.017	5.02	4.09	4.59
		8	172	88.858	137	98.92	5	99.14	3151	4335	22	1.31	0.089	13.27	10.46	3.55
s9234.1	93.475	2	356	93.667	323	93.67			34238	38410	14	1.11	0.002	2.27	2.04	7.43
		4	323	90.744	282	93.67			14946	17707	23	1.16	0.011	5.20	4.42	4.52
		8	260	85.403	226	91.41	19	93.67	6102	7523	33	1.19	0.045	10.75	10.40	3.15
s13207.1	98.462	4	468	98.431	374	98.43			59840	76148	25	1.26	0.006	5.03	3.97	12.72
		8	463	97.414	346	98.43			27680	38223	54	1.36	0.034	10.88	7.91	5.89
		16	282	86.943	223	97.22	22	98.43	8920	12242	67	1.34	0.106	30.73	24.69	4.75
s15850.1	96.682	4	443	96.531	393	96.68			52662	60475	31	1.14	0.004	4.42	3.87	8.45
		8	404	95.885	362	96.67			24616	28488	60	1.14	0.014	9.46	8.21	4.37
		16	346	93.14	296	96.67			10064	12688	92	1.22	0.079	23.13	18.43	2.85
s38417	99.475	8	744	99.467	599	99.47			123394	155862	56	1.26	0.009	11.75	9.32	13.70
		16	648	99.13	539	99.47			56056	69808	101	1.23	0.031	25.86	20.80	7.59
		32	437	84.228	366	99.46			26352	34277	129	1.28	0.086	35.61	42.36	5.95
s38584.1	95.852	8	633	95.892	514	95.89			92520	116193	50	1.25	0.010	10.14	8.09	14.24
		16	582	94.403	456	95.87			41040	54492	87	1.31	0.049	22.86	17.26	8.18
		32	515	91.809	411	95.86			18906	25723	124	1.33	0.176	49.63	36.56	5.74
s35932	89.9	8	41	89.921	32	89.92			6912	10841	71	1.28	0.010	17.00	11.00	11.44
		16	37	89.865	26	89.87			2808	5761	117	1.42	0.063	41.85	20.71	6.94
		32	33	89.837	24	89.84			1296	3597	174	1.38	0.222	90.67	33.17	4.67

[11] L. Li and K. Chakrabarty, "Test set embedding for deterministic BIST using a reconfigurable interconnection network," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 23, pp. 1289-1305, 2004.

[12] I. Hamzaoglu and J. H. Patel, "Reducing test application time for full scan embedded cores," in Twenty-Ninth Annual International Symposium on Fault-Tolerant Computing, 1999, pp. 260-267.

[13] K. M. Butler, J. Saxena, A. Jain, T. Fryars, J. Lewis, and G. Hetherington, "Minimizing power consumption in scan testing: pattern generation and DFT techniques," in Proceedings International Test Conference (ITC), 2004, pp. 355-364.

[14] N. Nicolici and P. Girard, "Journal of Electronic Testing, Theory and Applications, Special Issue on Low Power," 2008.

[15] P. Girard, "Survey of low-power testing of VLSI circuits," IEEE Design & Test, vol. 19, pp. 80-90, 2002.

[16] D. Czysz, G. Mrugalski, J. Rajski and J. Tyszer, "Low Power Embedded Deterministic Test," in IEEE VLSI Test Symposium (VTS), 2007, pp. 75-83.

[17] B. B. Bhattacharya, S. C. Seth, and Z. Sheng, "Double-tree scan: a novel low-power scan-path architecture," in Proceedings International Test Conference (ITC), 2003, pp. 470-479.

[18] S. Remersaro, X. Lin, Z. Zhang, S. M. Reddy, I. Pomeranz, and J. Rajski,"Preferred fill: a scalable method to reduce capture power for scan based designs," in Proc. of International Test Conference 2006, paper 32.2.

[19] N. Schemm, S. Balkir, and S. Seth, "Hardware Implementation of the Double-Tree Scan Architecture", submitted for publication, preprint available at http://cse.unl.edu/∼seth/pubs/Schemm.pdf.

[20] D. Xiang, K-W. Li, and H. Fujiwara, "Reconfigured Scan Forest for Test Application Cost, Test Data Volume, and Test Power Reduction", IEEE Transactions on Computers, vol. 56, pp. 557-562, 2007.

[21] S. Samaranayake, E. Gizdarski, N. Sitchinava, F. Neuveux, R. Kapur, and T. W. Williams, "A reconfigurable shared scan-in architecture," in Proc. 21st VLSI Test Symposium (VTS), 2003, pp. 9-14.

[22] I. Saha, B. B. Bhattacharya, S. Zhang, and S. C. Seth. "Planar Straight-Line Embedding of Double-Tree Scan Architecture on a Rectangular Grid," Fundamenta Informaticae, vol. 89.2, pp. 331-344, 2008.

[23] R. Sankaralingam, R. R. Oruganti and N. A. Touba, "Static Compaction Techniques to Control Scan Vector Power Dissipation", in *Proc. IEEE VLSI Test Symp.*, 2000, pp. 35-40.