Fast and Energy-Frugal Deterministic Test Through Test Vector Correlation Exploitation *

Ozgur Sinanoglu and Alex Orailoglu
Computer Science and Engineering Department
University of California, San Diego
La Jolla, CA 92093
{ozgur, alex}@cs.ucsd.edu

Abstract

Conversion of the flip-flops of the circuit into scan cells helps ease the test challenge; yet test application time is increased as serial shift operations are employed. Furthermore, the transitions that occur in the scan chains during these shifts reflect into significant levels of circuit switching unnecessarily, increasing the power dissipated. Judicious encoding of the correlation among the test vectors and construction of a test vector through predecessor updates helps reduce not only test application time but also scan chain transitions as well. Such an encoding scheme, which additionally reduces test data volume, can be further enhanced through appropriately ordering and padding of the test cubes given. The experimental results confirm the significant reductions in test application time, test data volume and test power achieved by the proposed compression methodology.

1 Introduction

High quality test for a core in a System-On-a-Chip (SOC) design necessitates full controllability of the inputs and full observability of the outputs of the core. The boundary scan scheme is widely used as this access to core I/Os is provided through shift registers which are directly controllable and observable. In this scheme, a flip-flop is connected to each I/O of the core, as illustrated in Figure 1, effectively ensuring the isolation of the core during test; these flip-flops are stitched together into a shift register, controlled from a single pin and observed through another.

Even though the test challenge is eased in a scan-based environment, the use of serial shift both for loading the test data into a shift register and for observing the test responses captured in the shift register, necessitates numerous test application cycles, equal approximately to the product of the shift register length and the number of test vectors. In addition to the prolongation in test application time, scan-based testing suffers from increased test power; during shift operations, the frequent transitions in the shift register reflect into rippling at the circuit lines unnecessarily, hence resulting in increased power dissipation. The consequent overheating of the chip during test may in turn result in damaging the chip.

Serial shift operations for observing the test responses can be eliminated through utilization of a compaction circuitry; instead of loading the test responses to a shift register and subsequently shifting them out serially, a MISR can be used to compress these responses into a signature, eliminating the necessity for a shift register for the outputs of the core. Yet, reduction in test application time necessitates getting around the serial loading of the test vectors into the shift register; instead, the test vector that already exists in the shift register can be updated to obtain the subsequent test vector by utilizing the correlation between two consecutive test vectors. Information regarding the correlation can be further encoded to enable a scheme wherein this encoded data is loaded into a

*The work of the first author is supported through an IBM graduate fellowship.

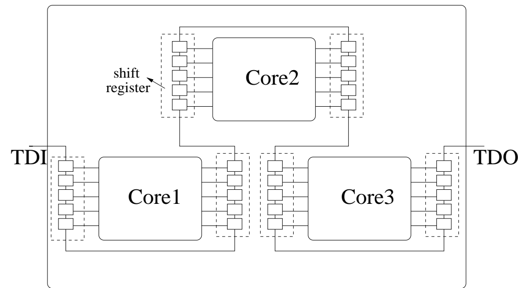

Figure 1. Boundary Scan Scheme

smaller shift register, a *correlation register*; this correlation information can then be decoded on-chip to directly update the current test vector and hence obtain the subsequent one. The size of the correlation register, which equals the number of bits in the encoded data, determines the reduction in test application time.

An encoding scheme which efficiently compresses the correlation information is needed, consequently. Such a scheme not only reduces test application time but test data volume as well, with the consequent benefit of enabling the utilization of less costly, smaller memory testers.

As the unnecessary rippling in the shift register, which drives the inputs of the core, is obviated through the techniques we propose, test power dissipation is reduced as well. An exceedingly small number of transitions, comprised of the ones essential to constructing the subsequent test vector, are the only ones reflected into the circuit.

To extract the correlation in the test data, the test patterns given, fully or possibly partially specified, can be reordered. In case of test cubes, the specification of *don't care* bits helps further increase the correlation among the resulting test vectors. The widespread use of full-scan techniques in industry enables the applicability of the proposed scheme as it imposes no order on test pattern application. Furthermore, the typically high unspecified bit density of the test data results in the effectiveness of the padding techniques in the extraction of test vector correlation.

In this paper, a test data compression scheme, which helps reduce test application time, test data volume and test power dissipation is presented. The encoded test data consist of the bit-flip locations; this information is utilized to update the current test vector and hence to construct the subsequent one. The on-chip decompression hardware necessitated is limited to a straightforward decoder.

The previous approaches in test data compression are outlined in section 2, followed up by the motivation for high test vector correlation in section 3. The proposed encoding scheme based on bit-flip locations is presented in section 3 while the details of the associated decompression hardware are discussed in section 4. Section 5 provides the test data compression algorithm which aims at minimizing test application time as well as test data volume. Experimental results and conclusions follow in sections 6 and 7, respectively.

2 Previous Work

In recent years, a considerable amount of effort has been expended in reducing test application time and test data volume through test data compression. In [1], a compression methodology is proposed wherein correlation among test vectors, in terms of the overlap between consecutive test vectors, is utilized; the leading bits of a test vector need not be shifted in if they happen to coincide with the trailing bits of the preceding vector. However, the technique necessitates another test pin through which additional test data is applied to control the test application timing. In [2], a compression methodology based on test vector reordering is proposed. The algorithmic limitations

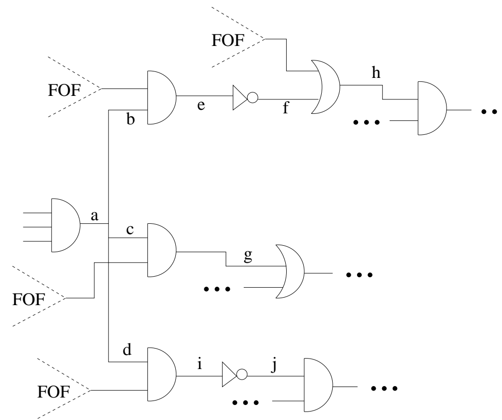

Figure 2. Structurally related faults in a circuit

of the greedy heuristic presented in the paper along with the inability of the methodology to exploit unspecified bits in test data result in limited compression ratios.

Compression methodologies based on run-length and Golomb encoding [3, 4] have also been presented. In both schemes, the test vectors are ordered so as to increase the correlation among these vectors and hence to obtain more compressed difference vectors. The encoded difference vectors are decompressed through an on-chip circuitry; the actual test vectors, which are reconstructed on-chip, are then shifted into the scan chain. As the on-chip test data decompressor decouples the scan chain from the ATE, the test vectors can be shifted into the scan chain at a faster rate than the one supported by the ATE; however, the restrictions imposed on the clock rate due to test power considerations limit their applicability.

In [5] and [6], test data is divided into blocks which are subsequently encoded. More frequently occurring blocks are encoded into shorter keywords, achieving higher levels of compression. In [6], dynamic compaction is performed so as to increase the frequency of certain blocks, whereas in [5], rarely occurring blocks are replaced with more frequent ones; in the latter scheme, fault simulation is performed after every block replacement to guarantee that the fault coverage remains intact. Both of these schemes necessitate decompressors of significant complexity, costly to implement on-chip.

In [7], a test pattern compression methodology is proposed wherein the number of virtual scan chains visible to the ATE is kept small, while the number of internal scan chains driven by the decompressed test data is increased; loading the decompressed test data in parallel to a large number of scan chains significantly reduces test data volume. Similarly, in [8], several scan chains are driven through a single test pin; only one of these chains receives deterministic test data whereas the remaining ones are fed from LFSRs in parallel, thus reducing test application time and test data volume.

3 Test Vector Correlation

Due to structurally related faults in a circuit, reasonably high correlation between the test vectors that are generated for these faults can be expected. Controllability and/or observability requirements of several faults might coincide due to the circuit structure; the similarity in the controllability and the observability requirements in turn results in the generation of highly correlated test vectors.

The illustrative circuit fragment in Figure 2 demonstrates several structurally related faults whose detection requirements coincide. The observability of all the faults in the four fanout-free regions, denoted by FOF, necessitates the justification of the line a to the value of 1. The controllability of the stuck-at-0 faults on the lines a, b, c, d, e, g, and i and the stuck-at-1 faults on the lines f, h and j necessitate the same assignment of line a to 1. It can therefore be expected that the test vectors that target the aforementioned structurally related faults are highly correlated.

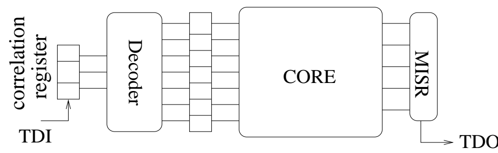

Figure 3. Utilization of a decoder for decompression hardware

4 Encoding Scheme

The expected high test vector correlation is utilized by the proposed encoding scheme. The utilization of bit-flip location information to construct the subsequent test vector not only helps exploit this correlation but furthermore limits the decompressor requirements to quite straightforward hardware; the consequent updating of the current vector is simply performed through the use of a decoder. Figure 3 illustrates the utilization of such a decoder for decompressing the encoded test data.

In this encoding scheme, the information regarding the bit position wherein the two consecutive test vectors differ is shifted into the correlation register. As the width of the correlation register is much smaller than the number of core inputs, the number of test application cycles is significantly reduced. Yet the effectiveness of this scheme is strongly determined by the number of bit-flips between consecutive test vectors; higher correlation among consecutive test vectors results in quicker construction of test vectors which in turn reflects into reduced test application times and test data volumes.

5 Decompression Hardware

Toggling a specific bit to update the current test vector necessitates a number of cycles for shifting the location of this bit into the correlation register; to prevent unintentional bit-flips during these cycles, the decoder is disabled in all but the last cycle during which the bit-flip operation is supposed to occur. A periodic signal is utilized to control enabling/disabling the decoder; the period of the signal equals the number of cycles it takes to shift in the bit-flip location completely. Such a periodic signal can easily be generated on-chip with no necessity for any additional test information or any test pins as the period is a constant value that depends on the number of core inputs. The complete decompression hardware and the timing for the decoder enable signal are provided in Figure 4; in this example, the period of the decoder enable signal is three cycles as every bit-flip necessitates the shift of the 3-bit data into the correlation register.

A core with M inputs necessitates a decoder of size $logM$ to M with the width of the correlation register being $logM$, resulting in $logM$ cycles to shift in a bit-flip location. If W denotes the total number of bit-flips between consecutive test vectors in a test set, $WlogM$ expresses both the number of shift cycles and the number of encoded test data bits. The reduction in test application

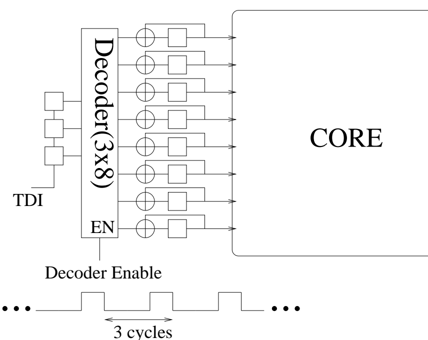

Figure 4. Decompression hardware

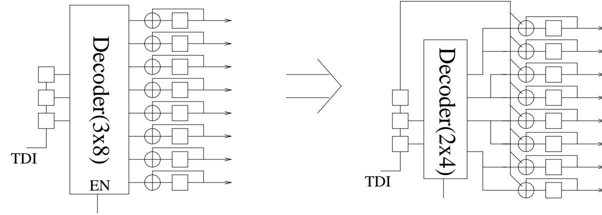

Figure 5. Reduced area-cost implementation of the decompression unit

time, σ_{TAT}, and test data volume, σ_{comp}, can hence be expressed as follows.

$$\sigma_{TAT} = \sigma_{comp} = \frac{N \times M}{W \times logM} \quad (1)$$

In the equation above, N denotes the number of test vectors. It is apparent that the total number of bit-flips, denoted by W, should be minimized for attaining the maximum reduction in both test application time and test data volume, as the number of test vectors and core inputs are predetermined values which cannot be adjusted.

Depending on the number of core inputs, the decompression hardware to be implemented on-chip may necessitate significant silicon area. To alleviate this problem, several cores in the SOC may share a single decompression unit. The consequent limited parallelism among the tests of these cores does not constitute a problem in a boundary scan environment wherein such parallelism is already limited due to the single test data input pin.

The area overhead of the proposed scheme can further be reduced through the implementation of the decompression unit as in Figure 5. In this implementation, the decoder size is halved as the number of decoder inputs is decremented by one; the least significant bit of the TDI register is not fed to the decoder. The smaller sized decoder selects a pair of scan cells only one of which is flipped based on the value of the least significant bit of TDI. Even though a more complicated set of gates needs to be attached to the outputs of the decoder, an overall decrease in the area overhead is attained due to the significant reduction in decoder size.

The significant routing overhead associated with the proposed scheme can be alleviated through the implementation of the decoder in a tree structure as in Figure 6. Such an implementation allows for a more flexible placement of the decompression unit, easing the wiring and routing processes. The aforementioned optimizations that aim at reducing the area and the routing overhead further accentuate the cost-effectiveness of the scheme we propose.

6 Compression Algorithm

To minimize the total number of bit-flips, the test cubes given can be ordered appropriately so as to achieve high correlation among consecutive test cubes. Moreover, the *don't care* bits of these

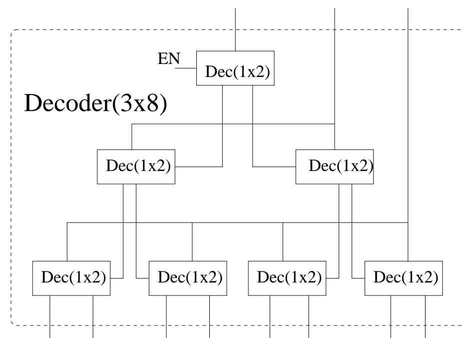

Figure 6. Implementation of the decoder in a tree structure

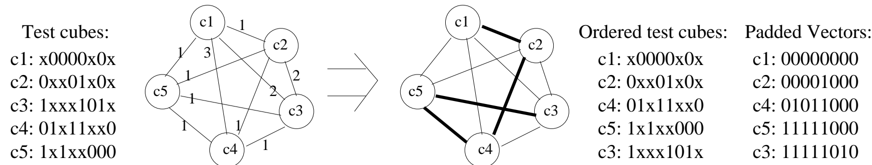

Figure 7. Test cube ordering for minimizing the bit-flips

test cubes can be specified to force the test vectors to resemble their predecessors even more. While the proposed algorithm leads to significant reductions in test application time and test data volume for the majority of the industrial cores, applicability on sequential cores can still be guaranteed by turning off the test cube ordering process.

The problem of ordering completely specified test vectors can be mapped to a traveling salesman problem [9] wherein a complete graph is constructed based on the test data; in this graph, the nodes represent the test vectors and the weights that are assigned to edges denote the number of bit-flips between the corresponding test vectors. The minimum weighted Hamiltonian path [9] in this graph corresponds to the optimal ordering of the test vectors. However, in a situation wherein the test data given consist of unspecified bits as well, finding the optimal solution imposes additional challenges; both ordering and padding of the test cubes should be considered so as to achieve the minimum number of bit-flips.

We utilize a graph-based heuristic wherein the nodes represent test cubes. The weight, assigned to an edge, consists of the number of conflicting bits of the two cubes corresponding to the nodes connected by the edge; opposite binary values in the same bit position of the two test cubes constitute a bit conflict. As long as subsequent to the reordering of the test cubes the *don't care bits* are appropriately specified, the number of bit-flips in a test data column is strictly determined by the ordering of the specified bits; a *don't care bit* and a specified bit in the same position have therefore no contribution on the weight. For instance, regardless of how the test cubes are ordered, the fourth leftmost test data column in the top-left corner of Figure 7 will contain at least a single bit-flip if the padding of the three unspecified bits is performed appropriately after reordering; unless the cube $c4$ is placed between the cubes $c1$ and $c2$, a single bit-flip is guaranteed.

A Hamiltonian path is constructed in this graph by selecting the minimum weighted edge at every step; visiting every node in the graph guarantees application of every test cube. Yet, every time an edge is selected, it needs to be ensured that the resulting path so far can still be extended to a Hamiltonian path; formation of loops should be prevented during the path construction. Application of the two selected test cubes back to back enforces specification of certain *don't care bits* of the cubes; the appropriate specification of the *don't care* bits of the cubes for minimizing the bit-flips is assumed even though the actual padding is not performed right away. The hypothetical update of the test cubes should somehow be reflected to the edge weights in the graph as the specification of the *don't care* bits might increase the bit-conflicts between the updated cube and the remaining cubes; the weights to be recomputed are the ones on the edges outgoing from the updated nodes.

The run-time of the algorithm is strictly determined by the number of test cubes as the graph size is proportional to the number of test cubes. Even though test cubes are statically compacted after test generation, their number can still be considerable for realistic circuits. Therefore, an efficient heuristic for solving this NP-Complete problem, satisfying computational efficiency requirements while delivering near-optimal solutions, is needed. The heuristic we propose is based on selecting the minimum weighted edge and performing weight updates at every step; the run-time is dominated by the edge selection process of quadratic time complexity as the complexity of the weight update process is linear in the number of test cubes. The run-time order of the algorithm is cubic in the number of test cubes, as the edge selection process is repeated once less than the number of test

Case 1:			Case 2:					
0	0		0	0	0	0	0
x	0		x	1	0	0	0
x →	0		x →	1 or	1 or	0 or	0
x	0		x	1	1	1	0
0	0		1	1	1	1	1

Figure 8. Padding of *don't care* bits for minimizing bit-flips

cubes.

Even better results could possibly be attained through the utilization of well-known heuristics for the Traveling Salesman problem, such as the Minimum Spanning Tree heuristic [10]. The solution offered by this heuristic is guaranteed to be less than twice the optimal solution if the triangular Euclidean inequality is satisfied by the edge weights in the graph. This property, inherently satisfied in the formulation we propose herein, guarantees results of bounded optimality through the utilization of the Minimum Spanning Tree heuristic for ordering test cubes.

The example in Figure 7 illustrates the formation of a Hamiltonian path on the graph constructed based on the test cubes given; the appropriate ordering is given in the same figure as well[1].

Once the test cubes are appropriately ordered, padding should be performed to minimize the bit-flips; several alternatives might exist, each leading to the minimal number of bit-flips. The appropriate padding of the ordered test cubes is given in Figure 7, resulting in 6 bit-flips in total. Appropriate specification of a string of *don't care* bits in a test data column is determined by the specified bits surrounding the string of *don't care*s. If the two binary bit values at the boundaries of a *don't care* run are identical, the *don't care* bits squeezed in between should be specified identical to the binary value present at either boundary, thus leading to no bit-flips. If the binary values at the two boundaries differ, several alternatives exist that satisfy the minimum possible number of a single bit-flip. These two cases are illustrated in Figure 8; the number of alternatives available for the opposite-boundary case, denoted by case 2 in the figure, equals one more than the number of consecutive *don't care* bits. The degree of freedom in the latter case can be exploited for selecting the padding option that leads to the maximal fault detection for test vector dropping; the number of resulting test vectors and hence the number of total bit-flips can be reduced by eliminating the test vectors with no contribution to fault coverage. Fault simulation of the repetitively reshuffled test data with the *don't care* bits appropriately specified helps identify such useless test vectors. To increase the effectiveness of this vector dropping scheme, the specification of the *don't care* bits is performed so as obtain the minimum possible difference between the number of 0's and 1's in every test data column after padding; experimental results of such a padding strategy exhibit improved test vector dropping.

7 Experimental Results

The proposed test data compression scheme has been applied to the combinational circuits in ISCAS85 [11] and the fully-scanned circuits in ISCAS89 [12]. The test vectors and the test cubes that are used to calculate the test application time, test data volume and test power reductions achieved by the proposed scheme are generated by ATALANTA [13]; the test cubes are statically compacted before the application of the proposed methodology. In this section, comparisons against other recent approaches [3, 4, 7, 8] that aim at reducing test data volume are provided.

Table 1 demonstrates the aforementioned improvements achieved by the proposed scheme. The number of circuit inputs and the size of the test set consisting of the test vectors that are generated by randomly padding the test cubes are provided in columns 2 and 3, respectively, whereas

[1]The intervening steps of the algorithm are omitted due to space constraints.

Circuit	Inputs (M)	Test vectors (N)	Test Cubes		Total bit flips (W)	σ_{TAT} (σ_{comp})	σ_{power}	Area(%)
			Original	Dropped				
c3540	50	148	196	171	403	3.06	223.89	6.49
c5315	178	119	272	199	439	6.03	1951.81	13.41
c6288	32	28	44	38	130	1.38	25.61	3.05
c7552	207	211	403	324	851	6.41	2318.23	11.03
s953	45	88	97	94	118	5.59	310.54	19.67
s5378	214	254	321	296	415	16.37	3419.02	18.49
s9234	247	371	462	423	344	33.29	8027.79	12.35
s13207	700	473	612	524	971	34.09	2639.18	16.07
s15805	611	433	676	511	1044	25.33	9401.28	20.72
s38417	1664	882	1121	1014	4811	27.73	4281.77	18.24
s38584	1464	680	926	808	2019	44.83	8201.14	16.20

Table 1. Reductions in test application time, test data volume and test power

columns 4 and 5 report the number of statically compacted test cubes before and after test vector dropping, respectively. The total number of bit-flips are provided in column 6 while test application time reductions which equal the compression ratios achieved by the proposed scheme are given in column 7; the results in column 7 are calculated based on the formula given in Equation 1. It should be noted that in the computation of the test time and test data reductions, the proposed scheme is compared against the randomly padded test vectors; even though random padding is more effective in vector dropping than the padding scheme proposed herein, significant reductions in test application time and test data volume are still attained. For instance, for the circuit $s5378$, the number of test vectors obtained through randomly padding the test cubes is 254, whereas the application of the proposed scheme and the consequent proposed specification of the *don't care* bits result in dropping the number of test cubes from 321 to 296; application of the compressed data that correspond to these 296 test cubes is actually 16.37 times faster than the application of the 254 actual test vectors for this circuit. The low density of the *don't care* bits and the small number of circuit inputs limit the reductions attained for the circuit $c6288$. Notwithstanding this exception, the proposed scheme tends to achieve significantly improved results, particularly on the circuits with more inputs. Furthermore, the enormous reductions in test power given in column 8 justify the effectiveness of the proposed scheme in preventing the unnecessary transitions in the scan chain which otherwise would boost test power[2]. For most of the circuits, three to four orders of magnitude in reduction of test power are achieved. The cost in terms of area overhead associated with the proposed scheme is provided in column 9. It should be noted that the area cost can be reduced through sharing of a single decompression unit among several cores; by allocating a single decoder for the cores, $s15805$, $s38417$ and $s38584$, the area cost can be reduced down to 7.72%.

Comparisons against previous approaches in terms of test data volume are provided in Table 2. The scheme we propose boosts compression levels significantly; the improvement over the best of the previous approaches ([7]) is around 50% for all the circuits.

8 Conclusion

In this paper, we propose a test data compression methodology for reducing not only test data volume but furthermore test application time and test power as well. The proposed compression scheme exploits the correlation between the test vectors; information regarding the location of the bit-flips is utilized to construct a test vector through updating the preceding one resident in the scan chain. This correlation is increased by the proposed compression algorithm which aims at

[2]Test power data are calculated based on the number of scan chain transitions.

Circuit	[3]	[4]	[7]	[8]	Proposed
s9234	23532	22252	NA	52824	2752
s13207	73514	41663	25344	60894	9710
s15805	44574	40718	22784	38010	10440
s38417	103520	92054	89856	154254	52921
s38584	114683	104111	38976	101185	22209

Table 2. Test data volume comparison against previous approaches

minimizing the total number of bit flips between consecutive test vectors; the test cubes given are ordered and padded appropriately. It is shown that not only does such an algorithm achieve the minimum test application time and test data volume under the encoding scheme we propose, but that it is furthermore highly efficient computationally.

Efficient construction of the test vectors through utilization of bit flip location information helps eliminate shifting of the actual test data into the scan chain and the associated unnecessary scan chain transitions. Elimination of these transitions, which otherwise reflect into the circuit and create further rippling at the circuit lines, results in a scheme wherein the test power dissipation problem effectively disappears.

Along with the appropriate implementation of the decoder, sharing a decompression unit among several cores helps reduce the area cost of the proposed methodology. Furthermore, the design of the decoder in a tree structure alleviates the routing overhead. Significant reductions in test application time, test data volume and test power can all be achieved in a cost-effective manner, consequently.

References

[1] C. Su and K. Hwang, “A Serial Scan Test Vector Compression Methodology”, in *ITC*, pp. 981–988, 1993.

[2] R. Dandapani, J.H. Patel and J.A. Abraham, “Design of test pattern generators for built-in test”, in *ITC*, pp. 315–319, 1984.

[3] A. Jas and N. Touba, “Test Vector Decompression via Cyclical Scan Chains and Its Application to Testing Core-Based Designs”, in *ITC*, pp. 458–464, 1998.

[4] A. Chandra and K. Chakrabarty, “System-on-a-chip Test-Data Compression Architectures Based on Golomb Codes”, *IEEE TCAD*, vol. 20, n. 3, pp. 355–368, March 2001.

[5] H. Ichihara, K. Kinoshita, I. Pomeranz and S. M. Reddy, “Test Transformation to Improve Compaction by Statistical Encoding”, in *VLSI Design Conference*, pp. 294–299, 2000.

[6] M. E. Ng and N. Touba, “Test Vector Compression via Statistical Coding and Dynamic Compaction”, in *IEEE Systems Readiness Technology Conference*, pp. 348–354, 2000.

[7] I. Bayraktaroglu and A. Orailoglu, “Test Volume and Application Time Reduction Through Scan Chain Concealment”, in *DAC*, pp. 151–155, 2001.

[8] A. Jas, B. Pouya and N. Touba, “Virtual Scan Chains: A Means for Reducing Scan Length in Cores”, in *VTS*, pp. 73–78, 2000.

[9] M. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, 1979.

[10] J. Pearl, *Heuristics: Intelligent Search Strategies for Computer Problem Solving*, Addison Wesley, 1984.

[11] F. Brglez and H. Fujiwara, “A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translator in Fortran”, *IEEE ISCAS*, June 1985.

[12] F. Brglez, D. Bryan and K. Kozminski, “Combinational Profiles of Sequential Benchmark Circuits”, *IEEE ISCAS*, vol. 14, n. 2, pp. 1929–1934, May 1989.

[13] H. K. Lee and D. S. Ha, “On the Generation of Test Patterns for Combinational Circuits”, Technical Report 12-93, Department of Electrical Eng., Virginia Polytechnic Institute and State University.