An Arithmetic Structure for Test Data Horizontal Compression

Marie-Lise FLOTTES, Regis POIRIER, Bruno ROUZEYRE
Laboratoire d'Informatique, de Robotique et de Microelectronique de Montpellier
161 rue ada, 34392 Montpellier CEDEX 5, France

Abstract

We propose a method for reducing test data volume of integrated circuits or cores in a System-on-Chip. This method is intended to reduce the required number of Automatic Test Equipment (ATE) output channels compared to the number of scan-in input pins in a classical multi-chain implementation (horizontal compression). Compression and decompression are based on arithmetic operations and structures which present a very low area overhead. The proposed compression scheme does not impact the fault coverage achieved by the original test sequence before compression.

1 Introduction

With the increased complexity of integrated circuits, the test data volume becomes difficult to manage. This is particularly true for SoC designs for which the final test sequence is the sum of individual cores and glue logic test sequences. Thus, the ATE capacities are reaching their limit in terms of memory size and number of test channels. A way of alleviating this problem is to reduce the amount of test data transferred between the ATE and the SoC.

To achieve such a reduction, several compaction and loss-less compression schemes are proposed in the literature. These techniques can be classified into three categories: those that intend to minimize the amount of test data per ATE channel (often referenced as vertical compression) [1,2,3], those that address the reduction of the number of ATE channels (often referenced as horizontal compression) [4,5,6,7,8,9,10] and those that address compression in both directions (e.g. [11][12]).

For the sake of ATE memory depth reduction, varied data encoding techniques have been proposed in order to minimize the data transmitted through individual test channels [1,2,3]. On-chip decompressors are used to restore the initial test sequence from the encoded data. These techniques do not allow improving the test parallelism even though ATE channels are still very expensive and test time in large designs very high. Conversely to the vertical compression, the horizontal compression is intended to reduce the number of ATE channels thus facilitating multi site testing.

The OPMISR solution [11] intends to reduce the required number of test channels as well as the amount of test responses to send back to the ATE. First, the input and output circuit ports are merged into bi-directional ports. Additionally, an MISR is inserted on scan chain outputs. The scan vector signatures (compacted responses) are transmitted back to the ATE trough I/O ports instead of bit-by-bit responses.

The most direct way for reducing the number of test pins is to serialize the test data. At the extreme, data can be entered serially with help of a single shift register connected to only one ATE channel. But this is done at the expense of increased test time and memory depth on this channel.

Another solution presented in [8] allows to test a circuit with the help of only one test pin: N-1 scan chains are loaded with pseudo-random vectors generated by on chip LFSRs, the last scan chain being directly loaded from one ATE channel. The role of the test sources with regard to the scan chains changes cyclically.

In the methods [6,7,10], a M to N bits decompressor is inserted between the M ATE channels and the N scan chains inputs ($M \leq N$). In [7], a reconfigurable switch is used to drive internal scan chains from a limited number of ATE channels. In the method proposed in [6] the decompressor is made of an Xor network. The decompressor in [10] consists in a ring generator and a phase shifter, the test sequence is specifically generated for the targeted test decompression scheme.

Finally, BIST can be seen as the ultimate mean to do compression in both directions since very few data are exchanged between the ATE and the CUT. Unfortunately, it is generally necessary to complete the pseudo-random test sequence by deterministic vectors due to resistant faults [4,5,9,12], increasing the impact of the DFT/BIST circuitry on the design.

The common problem inherent to all of these methods is their inapplicability for testing hard cores from the system designer point of view. In fact, these methods necessitate generating specific test data using ATPG and/or fault simulation. Since hard cores are delivered with their own test sequences and seen as black boxes, it is not possible to perform test generation processes on these elements.

In this paper we propose a new method for test pin optimisation that can handle standard circuits as well as SoCs including hard cores. The decompressor is inserted between the external ATE channels and the internal scan

1530-1591/04 $20.00 (c) 2004 IEEE

chain inputs and does not require any other additional design modification. The compression method does not require to generate any specific test sequence and thus does not impact the fault coverage achieved by a given pre-determined non-compressed test sequence.

In the section 2 we describe the proposed horizontal compression principle and the related decompression architecture. The compression and test time issues are respectively discussed in section 3 and 4. Before to conclude, several experiments are presented in section 5.

2 Compression principle and decompressor architecture

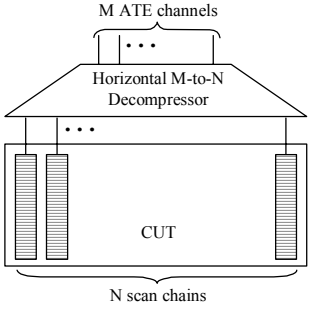

Figure 1: M-to-N horizontal decompressor

The basic architecture is presented in figure 1 for a N scan chains Circuit Under Test (CUT). A M-to-N decompressor is inserted on the circuit like in methods [6,7,10]. The N decompressor's outputs drive the N scan chains. The M decompressor's inputs are feed through M ATE channels.

In the remaining of this paper, N denotes the number of scan chains and M the number of ATE channels. Without loss of generality, we assume that all scan chains have the same length, every scan chain is thus constituted of F flip-flops. The test sequence is constituted of several test **patterns**. Each pattern of the original test sequence is composed of F Nbits **vectors**. For example, the circuit in figure 2 contains 3 scan chains, each being composed of 4 flip-flops. Thus the vector length is 3 and every pattern contains 4 vectors.

The compression principle consists to store differences between vectors instead of the vectors themselves. Hopefully, numerous differences (D_i) between the original

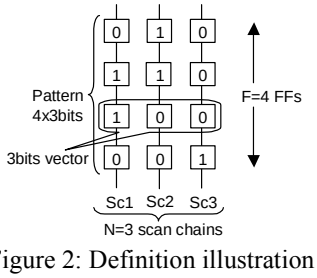

Figure 2: Definition illustration

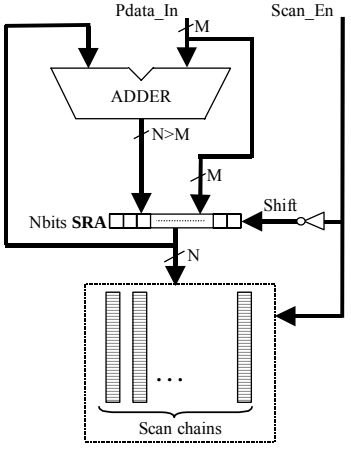

Figure 3: Decompressor structure

Nbits vectors (V_i) can be coded on M bits with M<N. Let's $\{V_1, V_2, \ldots\}$ be the initial test sequence and $D_i=V_{i+1}-V_i$ modulo 2^N be the differences between the two successive vectors V_i and V_{i+1}. D_i is stored in the ATE instead of V_{i+1} when Di can be coded on M bits, otherwise V_{i+1} is stored using $\lceil N/M \rceil \times$ Mbits memory words. The compressed stored sequence is thus composed of Mbits words, these words represent either a difference between two original vectors, or the $\lceil N/M \rceil$ subpart of an Nbits original vector.

A decompressor whose structure is given on figure 3 is inserted between the ATE and the CUT in order to restore the initial test sequence from the stored one. It is mainly composed of an adder and of an Nbits-MShifts Register/Accumulator (SRA). The idea of using an Adder/accumulator has even been investigated for test purpose but in the scope of random testing (e.g. [13]). The SRA register is composed of N flip-flops and is able to perform shift operations on M bits. An example of SRA architecture is given in figure 4 for M=3 and N=5.

The SRA register is either loaded from the adder using a parallel mode, or from the Mbits Pdata-in port using a semi-parallel mode based on Mbits shift operations. The parallel mode (Shift=0) is used for restoring a vector V_{i+1} from the difference D_i stored in the ATE and the

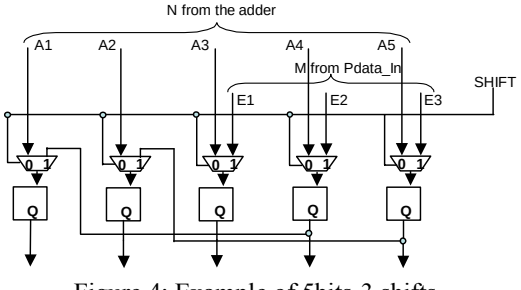

Figure 4: Example of 5bits-3 shifts Register/Accumulator

preceding vector V_i currently stored in the SRA. The SRA register can thus be set to $V_{i+1}=V_i+D_i$ in one clock cycle. The semi-parallel mode (Shift=1) is used to restore a vector V_i from the $\lceil N/M \rceil$ memory words stored in the ATE. This operation requires $\lceil N/M \rceil$ clock cycles.

At the evidence, the final test time depends of the proportion of differences (V_{i+1}-V_i) that can be coded on M bits.

However two control schemes, a regular one and an irregular one, allows to tradeoff between the control complexity and the test time.

For a regular control scheme, two sub-cases must be considered, either the differences between all the vectors within one pattern can be coded on M bits, or at least one difference cannot be coded on M bits. In the first case, the pattern is called a compressible pattern. Two phases are necessary for applying such a compressible pattern to the CUT. The SRA is first loaded from Pdata-In with the first vector of the pattern using the semi-parallel mode. Afterward, the SRA is loaded through the adder using the parallel mode.

Conversely, when at least one difference between vectors in a pattern cannot be coded on M bits, the whole non-compressible pattern is applied to the CUT using the semi-parallel mode.

The regular control scheme includes two steps. First, all the compressible patterns are applied to the CUT using $\lceil N/M \rceil$ semi-parallel mode cycles for the first vector and F-1 parallel mode cycles for loading the remaining F-1 vectors. Next, the non-compressible patterns are loaded using $F\lceil N/M \rceil$ semi-parallel mode cycles for each pattern.

For the irregular scheme, there is no more distinction between compressible or non-compressible patterns. All the differences that can be coded on M bits are stored in the ATE at the place of initial vectors. The parallel mode is used as much as possible for test time improvement while the semi-parallel mode is used for re-initializing the SRA register with a new test vector. Every pattern is restored from any number of differences and original vectors.

The advantage of the regular scheme is an efficient on-chip implementation since the control is very regular. The advantage of the second scheme is a shorter test time since the parallel mode is invoked more often than in the first scheme.

3 Compression issues

Pre-processing tasks must be applied to the original test sequence in order to compress as much vectors as possible.

Let's Dmax be the maximal difference between two vectors, the number of bits required for coding the differences (M) is equal to $\lceil \log_2 Dmax \rceil$. Obviously, the smaller Dmax is, the more efficient the compression is.

Initial sequence:

↓Bit b_4

↓Bit b_0

V_1: 1 1 0 0 1 (25) ⇒ Dmax= **V_3 - V_2**= (14-18)mod32 = 28

V_2: 1 0 0 1 0 (18) ⇒M=$\lceil \log_2 28 \rceil$=5

V_3: 0 1 1 1 0 (14)

V_4: 1 1 1 0 1 (29)

V_5: 0 0 0 1 0 (2)

Sequence after columns 1 and 5 switched:

↓Bit b_0

↓Bit b_4

V_1: 1 1 0 0 1 (25) ⇒ Dmax= V_4 - V_3=(29-14)mod32=15

V_2: 0 0 0 1 1 (18) ⇒ M=$\lceil \log_2 15 \rceil$=4

V_3: 0 1 1 1 0 (14)

V_4: 1 1 1 0 1 (29)

V_5: 0 0 0 1 0 (2)

Figure 5: Example of Column re-ordering for Dmax minimization

Two directions can be explored for reducing Dmax. First, the vectors can be reordered column-wise, and secondly, when dealing with test cubes instead of fully defined test vectors, don't care bits (X) can be efficiently assigned.

Concerning the bit reordering, let's consider for instance the example given in figure 5. By simply switching bits b_4 and b_0, M can be reduced from 5 to 4 bits. From structural point of view, the connection between the SRA and the CUT must be switched accordingly.

When dealing with test cubes, i.e. with unspecified values, the don't cares X can be assigned appropriately for decreasing the number of bits in difference coding. For instance, let's the 4-bit vectors V_1: 1X00 and V_2: X101; if both Xs in V_1 and V_2 are assigned to 1, then the difference can be coded using only one bit: $D_2=V_2-V_1=1$(decimal value). Conversely, if X=0 in V_1 and X=1 in V_2, this difference is equal to 5 requiring 3 bits for coding.

At the evidence, the exhaustive exploration of all column permutations as well as all X's assignments is inconceivable since there are N! column orders and 2^p possible assignments with p being the number of X's in the test sequence. We thus proposed a heuristic for dealing with these two issues. First, the columns $b_{N-1},\ldots,b_1,b_0$ are re-ordered by decreasing number of X's (from the most significant bit to the least one). The underlying idea is that Xs on m.s.b give more freedom for good assignments than Xs on l.s.b with regard to difference optimization. Don't care values are then assigned to '0' or '1' with the aim of reducing the difference between successive vectors. The final goal is to minimize Dmax.

V_{k0}: 1 0 1 x 0	V_{k0}: x 0 0 1 1	V_{k0}: o 0 0 1 1
V_{k1}: 0 x x 1 0	V_{k1}: 1 x 0 x 0	V_{k1}: 1 x 0 x 0
V_{k2}: x x 1 x x	V_{k2}: x x x 1 x	V_{k2}: z z z 1 z
V_{k3}: 1 x 0 x x	V_{k3}: x x x 0 1	V_{k3}: x x x 0 1
V_{k4}: 0 x 1 x 0	V_{k4}: x x 0 1 0	V_{k4}: x x 0 1 0
Initial patterns	(a)	(b)
D_{k1}: *0* 0 1 1 0	V_{k0}: o 0 0 1 1	D_{k1}: *0 0* 1 1 1
D_{k2}: *0* 1 0 0 0	V_{k1}: 1 o 0 o 0	D_{k3}: *0 0* 1 0 1
D_{k3}: *0* 0 0 1 1	V_{k2}: o o o 1 o	D_{k3}: *0 0* 1 1 0
D_{k4}: *0* 0 1 0 1	V_{k3}: z z z 0 1	D_{k4}: *0 0* 1 0 1
	V_{k4}: z z 0 1 0	
(c)	(d)	(e)

Figure 6: Full assignation process

Figure 6 illustrates the whole assignment process. First, bits are re-ordered with regards to the number of X in every column (see 6.a). Next, the most specified vector (V_{k0}) is pre-assigned to a deterministic value: V_{k0}=o0011 where the sign o (resp. z) stands for an X temporarily assigned to 1 (resp.0) (see 6.b). Differences between vectors can now be coded on 4 bits (see 6.c). The result of the next step is given in 6.d where don't care bits in V_{k2} have been re-assigned for Dmax minimisation. Now the differences shown in figure 6.e can be coded on three bits. The interested reader can refer to [14] for further details on this heuristic.

4 Timing issues

In this section we analyze the timing issues according to the regular scheme.

The required number of cycles for loading the scan chain is $T_{uncomp}=P_{uncomp}*\lceil N/M \rceil*F$ for P_{uncomp} non-compressible patterns, and $T_{comp}=P_{comp}(\lceil N/M \rceil+F)+1$ for P_{comp} compressible patterns. The total test time equals to $T_{comp}+T_{uncomp}+F$.

For instance let's consider a circuit made of 250 FFs with a 1000-patterns test sequence, and let assume that only 5 test pins are available. With a classical approach, the FFs are organized into 5 scan chains of 50 FFs each.

This test scheme results in a total test time of 1000(50+1)+50=51050 clock cycles.

Conversely, using our 5-to-10 bits decompression structure, the FFs can be organized into 10 scan chains of 25 FFs each. The test time for this new scheme ranges from 1000(2+25)+1+25=27026 to 1000*2*25+25=50025 clock cycles, depending on the number of patterns that can be compressed. The proposed upper bound is lower than the classical approach test time.

More formally, let's denote $k=\lceil N/M \rceil$ and L the scan chain length of a circuit with M scan chains. The total test time is $T_{classic}= P(L+1)+L$ cycles for the classical mutli-chain approach. Using a decompression scheme from M to N bits, the FFs can be organized into N scan chains of length L/k. Assuming that all patterns can be compressed on M bits, test time is now equal to

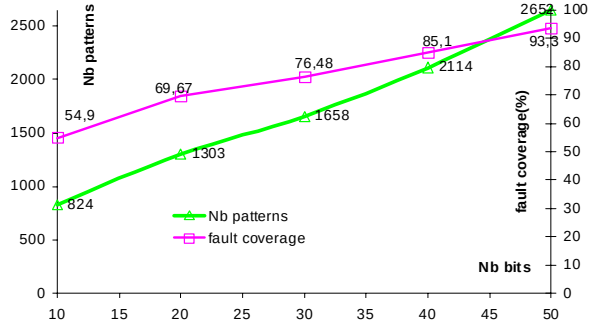

Figure 7: ATE channel usage and related achieved FC% for compressed patterns

$T_{comp}=P_{comp}(k+L/k)+1+L/k$. It comes that the ratio $T_{classic}/T_{comp}$ is close to $k(L+1)/(k^2+L)$ which is always greater than one since k<L. On the other hand the ratio $T_{classic}/T_{uncomp} \approx 1$, consequently the test time does not increase even if none of the patterns can be compressed. Thus, the proposed method can only lead to test time reduction and does not increase ATE memory depth requirements.

5 Experimental results

Experiments have been conducted on ISCAS89 benchmark circuits. All the I/Os and flip-flops are inserted in the scan chains. The reference test sequences are generated with help of an industrial test suite [15] and include test cubes.

First experiments have been conducted according to the regular scheme (i.e. a pattern is compressed iff the differences between every pair of successive vectors in the pattern can be coded on M bits).

The figures of merit of the compression scheme for the s9234 benchmark circuit are reported in figure 7. The circuit's FFs are organised into 50 scan chains of length 5. The curves report the number of compressible test patterns and the related fault coverage with respect to desired number of ATE channels. For example with M= 30 instead of 50, only 1658 patterns among the 2652 can be compressed. The corresponding fault coverage is 76,48% instead of 93,3% for the original sequence. Therefore, non-compressible patterns must be loaded in the semi-parallel mode into SRA for restoring the initial fault coverage.

Table 1 resumes the same kind of results for various benchmark circuits. In the first row, *circuit-name(N x F),* N and F denote respectively the number of scan chains and their length. For each circuit, the first column (M) reports the user-given number of ATE channels and the second column shows the percentage of initial patterns compressible on M bits. The third column reports the fault coverage achieved with these patterns. These preliminary results show that most of the time, the compressed test

s420 (9 x 4) Initial FC=100%			s1196 (8 x 4) Initial FC=100%			s1423 (13 x 7) Initial FC=90.4%		
M	P(%)	FC(%)	M	P %	FC(%)	M	P(%)	FC(%)
7	98.18	99.29	6	97.54	99.5	10	97.53	90.3
5	88.73	96.44	4	91.55	98	7	77.94	88.31
3	63.27	83.14	2	51.58	76.67	5	60.67	84.72
1	34.18	64.13	1	28.35	56.58	3	53.70	82.44

s5378 (43 x 5) Initial FC=99.05%			s5378 (10 x 22) Initial FC=99.05%			s9234 (10 x 25) Initial FC=93.01%		
M	P(%)	FC(%)	M	P(%)	FC(%)	M	P(%)	FC(%)
35	93.97	98.9	8	98.77	99.05	8	98.16	92.73
30	81.99	97.54	6	95.94	98	6	67.33	82.21
25	61.99	92.3	4	70.66	91.25	4	31.92	59
15	46.79	87.79	2	55.17	85.13	2	23.50	52.4

Table 1: Percentage of initial patterns compressible on M bits and related fault coverage

sequence have to be completed for achievement of the initial fault coverage.

We also compared our methodology to the standard multi-scan chain strategy in terms of ATE requirements. Table 2 reports our results for the S9234 benchmark circuit. We compare 3 versions of the circuit. The first one contains 6 scan chains of length 42 (version 1). The second one contains 10 scan chains of length 25 (version 2). The third one contains 10 scan chains of length 25 and a 6-to-10 decompressor (version 3). The same test sequence (TS) is used for all the three versions. Concerning version 3, the maximum fault coverage is achieved with the help of a sub-sequence of compressible patterns (TS_{comp}) plus a sub-sequence of non-compressible patterns (TS_{uncomp}). Both sub-sequences are issued from the original sequence TS: $TS=TS_{comp} \cup TS_{uncomp}$.

As it can be seen in Table 2, our method allows to reduce the test pin number from 10 to 6 at the expense of a 33% increase in test time and 36% in memory depth compared to the solution with 10 test pins while the total test data volume reduces of 20%.

Our solution has also to be compared with the modification of the circuit into the version 1 with 6 scan chains and no compression. Such a solution would increase test time by 24%. Our strategy results also in a 24% memory depth saving due to shorter scan chains. While only horizontal compression is targeted, the proposed method indirectly induces a vertical compression.

Whereas in previous experiences the actual applied test sequence is composed of compressed data and uncompressed data issued from the original uncompressed test sequence, further test length improvement can be achieved if fault dropping and useless pattern elimination are performed after don't care bit assignment. Such improvements are presented below.

	Version 1	*Version 2*	*Version 3*
#Test pins	6	10	6
# uncomp. patterns	2660	2660	869
ATE memory depth	111720	66500	43450
# comp. patterns	------	------	1791
ATE memory depth	------	------	46566
Test dat volume	670Kbits	665Kbits	540Kbits
Test time	114422	69185	91833

Table 2: 6 and 10 classical multi-scan chain implementations vs 6-to-10 decompression

Table 3 presents compression results for 4 benchmarks and a compression ratio of 50% (M=N/2). The first row reports the number of implemented scan chains N, the number of ATE channels M, and the scan chain length F. Initial fault coverage and related number of test patterns are reported in the second and third rows. Because of the don't care assignment in the compression step, some of the compressible patterns may become redundant. A pattern is redundant if it is identical to another pattern or if it covers faults tested from another source. Test pattern comparison and fault dropping allow to delete redundant patterns. The fourth row gives the final number of compressed patterns after elimination of these redundant patterns. The next row gives the number of uncompressed patterns required for achieving the initial fault coverage. The overall test time and the test data volume per pin are reported in the two following rows.

Another strategy for the same ATE usage would consist in re-designing the circuit with two times less scan chains. The second part of the table reports test length,

	s15850	s35982	s38417	s38584
N/M/F	77/39/8	118/59/15	152/76/11	122/61/12
Initial FC%	96.37	88.61	99.44	95.47
#patterns	314	345	460	762
#comp. patterns	54	36	60	68
related FC	73.54	86.81	70.06	77.78
# uncomp. patterns	132	14	189	222
test time	2661	1048	4950	6293
data volume per pin	2598	996	4878	6212
N=M/F	39/16	59/30	76/22	61/24
# patterns	175	49	243	299
data volume per pin	2800	1470	5346	7176
test time	2992	1549	5611	7499

Table 3: Regular scheme for 50% horizontal compression

	s15850	s35983	s38417	s38584
N/M/F	77/8/8	118/12/15	152/16/11	122/12/12
Initial FC%	96.37	88.61	99.44	95.47
#patterns	314	345	460	762
#comp. patterns	51	25	65	65
related FC	70.74	80.44	70.54	73.43
# uncomp. patterns	121	23	190	232
test time	11933	4375	24747	31825
data volume per pin	12434	4696	22265	29205

N=M/F	8/78	12/148	16/104	12/122
# patterns	175	49	243	299
data volume per pin	13475	7203	25272	36478
test time	13903	7449	25619	36899

Table 4: Irregular scheme for 90% horizontal compression

data volume and test time for this second strategy. Please note that, for fair comparison, we did not use the same initial test sequence made of test cubes, but a fully specified and optimized one using static and dynamic ATPG compaction. It can be seen from these results that our approach always leads to test data volume and test time reduction compared to the classical approach. If we had used the test sequence made of test cubes in this second classical strategy, the comparison results would even be better. For instance, the test time of a classical 8-scan chain implementation for the s15850 benchmark circuit would be 314(16+1)+16=5354 clock cycles for an initial test sequence including 314 patterns. This test time has to be compared with our test time for the same ATE usage: 2599 clock cycles using a regular control scheme.

As discussed before, an irregular control scheme allows us to compress even more data than the regular one. Related results are reported in Table 4 for a compression ratio of 90%. Again, experimental results compare favorably our strategy to the classical one.

6 Conclusion

In this paper we have presented a simple and effective method for test pin reduction based on a low cost decompression structure. It must be noticed that the decompression structure is independent of the patterns and does not impact the fault coverage of a given test sequence. Furthermore, the compression technique does not require any knowledge of the embedded cores.

Finally it compares favorably with the traditional approach since in one hand it does not require any re-design step for achieving a given ATE usage in terms of number of channels for feeding the scan chains and, on the other hand, it improves the test time and the test data volume.

References

[1] A. Chandra and K. Chakrabarty, “Test data compression for system-on-chip using golomb codes”, VTS’00, pp: 113-120.

[2] A. El-Maleh, S. Al Zahir, and E. Khan, “A geometric-primitives-based compression scheme for testing system-on-chip”, VTS'01, pp 54-59

[3] A. Jas, J. Ghosh-Dastidar and N.A. Touba, “Scan vector compression/decompression using statistical coding”,VTS’99, pp:114-120.

[4] D. Das and N.A.Touba, “Reducing test data volume using external/LBIST hybrid test patterns”, ITC’00, pp: 115-122.

[5] S. Hellebrand, H.-G. Liang and H.-J. Wunderlich, “A mixed mode BIST scheme based on reseeding of folding counters “, ITC’00, pp:778-784.

[6] A. Orailoglu, “Test volume application time reduction through scan chain concealment”, DAC01, pp: 151-155

[7] H. Tang, S.M. Reddy and I. Pomeranz, “On reducing test data volume and test application time for multiple scan chain designs”, ETW’03, pp 341-346

[8]A.Jas, B. Pouya, and N. A. Touba, "virtual scan chain: A means for reducing scan length in cores”, VTS’00, pp: 73-78

[9] Rainer Dorsch, H-J Wunderlich, “Accumulator based deterministic BIST”, ITC 98, pp:412-421

[10] J. Rajski et al., "Embeded Deterministic Test for Low cost Manufacturing Test", Proc. ITC 2002, pp 301-310.

[11] C. Barnhart, V. Brunkhorst, F Distler, O. Farnsworth, B. Keller and B. Koenemann “OPMISR: The foundation for compressed ATPG vectors“, ITC’01, pp:748-757.

[12] H.-G Liang, S. Hellebrand and H.J. Wunderlich, “Two-dimensional test data compression for scan-based deterministic BIST”, ETW’01, pp: 291-298.

[13] F. Mayer, A. Stroele,"Configuring Arithmetic pattern generators and response compactors from the RT-module of a Circuit", Proc. ATS 98, pp 15-20.

[14] M.L. Flottes, R. Poirier, B Rouzeyre,"On using Test Vector Differences for Reducing Test Pin Numbers", Proc. DELTA 2004, 28-30 Jan. 2004, Perth, Australia.

[15] Synopsys test suite version 4.1