Improving the Test of NoC-Based SoCs with Help of Compression Schemes

Erika Cota, Julien Dalmasso, Marie-Lise Flottes, Bruno Rouzeyre

Improving the Test of NoC-based SoCs with Help of Compression Schemes

Julien Dalmasso[1] Érika Cota[2] Marie-Lise Flottes[1] Bruno Rouzeyre[1]

[1]LIRMM
Univ. de Montpellier / CNRS
161 rue ada, 34392 Montpellier CEDEX 5
France
{dalmasso,flottes,rouzeyre}@lirmm.fr

[2]PPGC – Instituto de Informática
UFRGS
Po Box 15064, 91501-970
Porto Alegre, BR
erika@inf.ufrgs.br

Abstract

Re-using the network in a NoC-based system as a test access mechanism is an attractive solution as pointed out by several authors. As a consequence, testing of NoC-based SoCs is becoming a new challenge for designers. However, the effectiveness of testing methods is highly dependent on the number of test interfaces with the tester. This paper proposes the use of a test data compression scheme to increase the number of test interfaces (thus increasing test parallelism) without increasing the number of required Automated Test Equipment (ATE) channels. We show that the combination of compression and NoC-based test scheduling allows a drastic reduction of the system test time at the expense of a very small area overhead.

1. Introduction

Network-on-Chip (NoC) architectures are now known as a good alternative to bus based. Systems-on-Chip (SoC) architectures, providing efficient communication among cores at lower design and power costs. To become an industrial reality, this new design paradigm also depends on the definition of feasible and efficient test mechanisms. Such mechanisms must tackle both the test of the network itself and the test of IPs or user-defined logic connected to the NoC.

Concerning the test of the functional blocks (or cores), reusing the NoC as a Test Access Mechanism (TAM) is an attractive solution since it allows saving the cost of extra test dedicated communications. However, this alternative is very challenging since routers, channels, interfaces, and cores placement in the system are generally optimized in the first place for communication in mission-mode, not for test.

Efficient test scheduling approaches have been presented in the literature to tackle the test of IP cores through embedded NoC [1-6]. Test time optimization is achieved with the help of a better utilization of the network channels at the price of more complex test wrappers, design effort, area overhead, and, possibly, power consumption during test.

A preemptive test scheduling is presented in [4] where one test packet contains only one test vector or one test response for a given core. Since no NoC's path is reserved for any core, the first available path is used for the current unscheduled packet issued from an ordered list. Test packets are scheduled individually and core testing pipeline can be interrupted.

A non-preemptive scheduling is presented in [6] where one test packet contains all test vectors (or all test responses) of the Core Under Test (CUT). The scheduler assigns one routing path to each core. All the resources on the path (including input/output ports) are reserved for the test of this core until the entire test set is applied. Test vectors are routed from a system input to the core, and test responses are routed from the core to a system output in a pipelined way. The numbers of test inputs and outputs are assumed to be equal in this case. While apparently preemptive scheduling offers more flexibility, non preemptive scheduling leads to better test times in practice [4, 6].

The test packet format in the above cited NoC-based approaches assumes that one packet contains the test data (vector or response) of a single core under test but recent works have proposed other packets formats aiming at further reducing test application time by increasing the network usage e.g. [1][2]. Despite a better usage of the network channels, the design effort and area overhead involved in the implementation of those approaches are important drawbacks.

Note that if the system test time is related to the test scheduling efficiency, it is also strongly correlated to the usage of the test interface. Test ports are built in the first place from functional inputs/outputs, but extra test pins are eventually used for test time improvement since increasing the number of test interfaces allows larger test parallelism. However, the main drawback when reusing the NoC for test purpose is precisely the limited number of test interfaces. On the one hand, a small number of test ports limits the possibilities of test parallelism, and on the other hand, there is a cost associated with the introduction of extra test ports: the circuit level cost (extra pins) and the resource level cost (extra ATE channels for test pin monitoring). It is clear that a test strategy capable of increasing the test parallelism without increasing the number of visible test interfaces is highly desirable to make current NoC-based test approaches actually feasible.

The paper is organized as follows: in Section 2 we detail the consequences of test compression at system level. Section 3 presents the application of a compression technique during NoC-based test

978-0-7695-3170-0/08 $25.00 © 2008 IEEE
DOI 10.1109/ISVLSI.2008.86

scheduling. Test space exploration is presented in Section 4, and Section 5 presents experimental results. Conclusions are drawn in section 6.

2. Test Data Compression

Several Test Data Compression (TDC) techniques aiming at reducing the number of visible scan chains have been developed since the early 2000s. Concerning test vector compression, they consist in compressing original test data off line, storing the compressed data in the ATE, and decompressing these data on-chip for restoring the initial test vectors (Figure 1). TDC techniques are built toward the compression of test patterns for standalone cores and they rely on the fact that test patterns originally contain don't-care bits. These don't care bits do not have to be stored into ATE but can be supplied on-chip in some other ways. LFSRs [7,8], Xor networks [9,10], ring generator [11], RAM [12,13], arithmetic units [14], and test pattern broadcasting among multiple scan chains [15-17] constitute a range of solutions for minimizing the number of data to be stored in the ATE and the number of visible test interfaces (scan chains in this case).

Note that for a fixed number of scan chains (N), the reduction of the required number of ATE channels (W<N) may result in additional test time compared to a solution where the number of ATE channels equals the number of scan chains (W=N). Indeed, compressing an N-bit test vector on a W-bit word is not always feasible. Non-compressible test data must either be serially loaded in the core under test or substituted by additional compressible test patterns in order to maintain the test quality. In any case, a side-effect is an increase in the test time of the core under test.

At system level, a first compression strategy can be to build the system from plug-and-play cores including dedicated decompression hardware in their wrappers, but this solution may induce a large area overhead. A second strategy consists in using a centralized architecture where decompressors are shared between

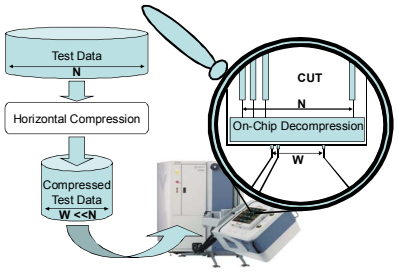

Figure 1: Compression scheme

several cores limiting thus the extra hardware. Note that in this case, the decompressor structure must be independent from the test sequences and the core netlists.

Concerning test responses, several space compactors have been recently proposed (e.g. [18, 19]). Such compactors are designed to tolerate a given number of Xs occurring in test responses. Conversely to the compression of test patterns, spatial test response compactors designed for a given X-tolerance do not impact test time. This means that there is no penalty in the test time when reducing the number of visible output pins. Spatial and/or time response compactions of test responses will not be further discussed in this paper.

We propose here a new strategy for testing cores in a NoC-based architecture that combines a test data compression scheme and a non-preemptive test scheduling approach. We will show that a limited number of system pins can be used in such a way that test parallelism in the NoC during test is not precluded.

3. Application to NoC-based Testing

We propose to use a vector compression scheme [14] to deal with a limited number of test interfaces. Let us assume that functional I/O pins of the system are used for connecting the system to the ATE. Let T_i be an N-bit test interface (N is the NoC channel width). Without compression and according to the number F of functional I/O pins, a number T of test interfaces can be defined by $T=\lfloor F/N \rfloor$. Using compression, we can decrease the number of I/O pins dedicated to each test interface and thus increase the total number of test interfaces T. A higher number of test interfaces leads in turn to higher test parallelism.

Figure 2 shows a typical NoC architecture with a single input port F_i and a single output port F_o used as test interfaces. Cores are connected to the NoC infrastructure using standardized protocols, which can be implemented as wrappers. Hereafter we give an overview of the interfaces between cores and NoC and between the SoC and the ATE.

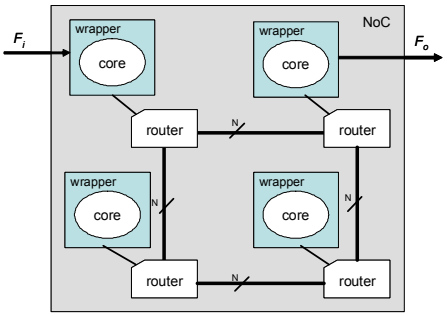

Figure 2: NoC-based SoC architecture

The first role of the core wrappers is to adapt the NoC interface to the core functional interface. When reusing the NoC as TAM, this wrapper must also implement new functions to connect not only the functional interface, but also the test interface of the core to the interconnect platform. Amory *et. al.* propose in [20] and [21] a method to design IEEE 1500 compliant test wrappers when functional interconnections are reused for test purpose. We assume here that such wrappers are available for each core being tested.

The NoC-based testing approach also assumes the reuse of system functional pins to connect the TAM (here the NoC) to an external tester. In this case, an interface adapter is also required to perform a parallel-to-serial conversion between the functional/test SoC interfaces and the NoC as proposed by Amory et al. in [22]. Figure 3 shows the modification entailed by inserting a decompressor for test patterns. This functionality can be embodied into the ATE interface replacing the existing serial-to-parallel converter and vice-versa. Thus, in mission mode, the connection between the system pins and the NoC are as defined by the system designer. During test, the ATE interface formats the data coming from the tester and injects them into the NoC as normal data. The core under test, with its wrapper configured in test mode, receives N-bits data (as in mission mode) generated by the on-chip decompressor logic. Similarly, an ATE interface at the system output receives the response data from the cores and converts it to the ATE format, before sending it back to the external tester.

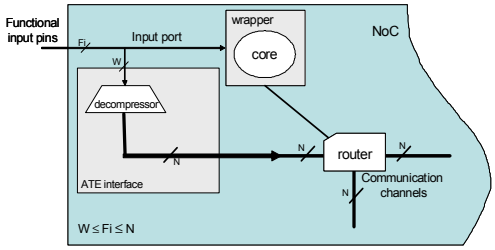

Figure 3: ATE interface with compression

Data are transmitted through the NoC in the form of messages called packets. A packet is composed of a header giving among other information the address of the target core, the body of the message called the payload, and a tail indicating the end of the message. The payload is composed of several pieces of information called flits whose bitwidth equals the NoC bitwidth. For test purpose, a test header is also added at the beginning of the message.

Under the compressed vectors assumption, each flit of the payload is compressed. But as explained in Section 2, some original flits may not be compressible and thus are split into several flits to fit with W. In the same way, tail, packet and test headers are split into narrower flits. As a result, the compressed test message contains more flits than the non-compressed one. As a consequence, the test time for individual cores is increased.

The problem of minimizing the system test time depends thus on two optimization procedures: i) one must find the best partition of system input pins into input test ports (called hereafter *test architecture*); and ii) one must define which input port should be used by which core (*test scheduling*). These two procedures are detailed in the next section.

4. Test Space Exploration

4.1. Test Scheduling

In the non-preemptive test scheduling, an access path is reserved during the whole test of a core, i.e., all test patterns are sent serially while all test responses are also sent back to the tester. To evaluate the core test time within the NoC, one may consider thus a single input test packet containing all test patterns traversing the network from the test input port up to the core. Similarly, a single output test packet contains all test responses collected from the core. It must be noted that within the network, both input and output test packets of a given core have the same size (number of flits), which is given by the length of the longest wrapper scan chain of that core multiplied by the number of test vectors.

As mentioned before, the input packet coming from the ATE might have more flits when compression is used. It must be noted that the narrower the input port, the longer the payload since it is more difficult to compress the patterns. In addition, there is a latency to inject this data in the network, which increases the core test time. To sum up, the total system test time varies with the number of input ports and the number of pins available at each port of the system. However, as more input ports are available, we expect that the increase in the test parallelism compensates this increase in the size of the test packet.

A NoC test scheduling algorithm that deals with the compression scheme requires information regarding the length of the test packets for each core with respect to each available input port (i.e. with different numbers of pins). The non-preemptive test scheduling algorithm described in [6] was adapted to include the extra compression information per port. The new scheduling algorithm receives then, in addition to the cores test data and NoC data: i) the number, location, and bitwidth of available test input ports; ii) for each core, the size of the input test packet per port. When selecting a test access path to a core, the algorithm evaluates the time required to transfer, decompress, and transmit all test patterns from the ATE to the core, as well as the time required to transmit all test responses back to the ATE. The access path (input and output) is blocked until the last test response is received by the ATE. In the new algorithm we also improved the procedure that selects an I/O pair for a core, since each path implies a different test time increase for each core. Now, the free I/O pair assigned to a core is the one that increases the least the core test time. The pseudo-code of the test scheduling algorithm is shown in Figure 4. More details on the algorithm can be found in [6].

Procedure Compressed_NoC_schedule

```
1 Start with sorted cores in decreasing order of test time;
2 Define I/O pairs list;
3 For every possible permutation between Inputs and Outputs
4    While there are unscheduled cores
5       For each unscheduled core
6          Find a free I/O pair (at current time) that gives the
              shortest core test time;
7          If no free I/O pair
8             Update current time, repeat from 4;
9          else
10            Check the corresponding routing path;
11            If path is blocked
12               If all cores have been attempted
13                  Update current time, repeat from 4;
14               else
15                  Try next core in the list;
16            else
17               Assign core to the path, update time labels;
18 Repeat from 2 for a user-defined number of cores permutations;
```

Figure 4 - Non-preemptive test scheduling considering compression data

4.2. Test Architecture

The total system test time varies with the number of input ports and the number of pins available at each port of the system. On the other hand, the available ATE channels can potentially be connected to any input pad of the system. The ATE channels can be partitioned into test input ports in many ways and only some partitioning lead to minimal test time. It is important, thus, to explore the space of possible partitions of available ATE channels (or system input pins) into input test ports to find the best one in terms of system test time.

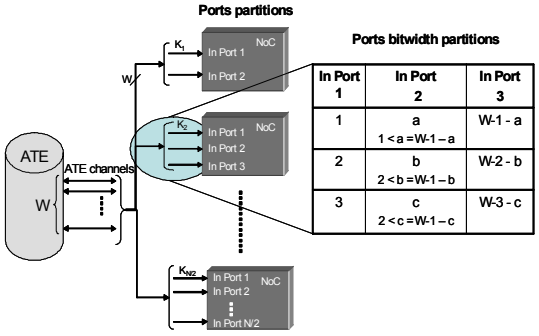

Figure 5 - Partitions space

The partition problem can be stated as follows: for a number W of available ATE channels, and a number T of required test input ports, find all possible partitions of W into T. The number of test input ports T varies from 1 to R, the number of nodes in the network. For T=1, there is a single partition and test parallelism is minimum. For T=R, all cores have direct access to the ATE and no test scheduling is necessary. In practice, T is ranging from 2 to R/2 input ports, as it will be detailed in the experimental results.

Figure 5 presents the partitions space that must be sought. For a given number W of ATE channels that serve the test input pins, and for every configuration k_i of input ports ($2 \le k_i \le R/2$), all distinct partitions of W into k_i is checked. For each partition, test scheduling is run and the best test time among all partitions is selected. We note that not all possible partitions are tried. This is because the test scheduling algorithm already checks a number of cores and I/O pairs permutations, which allows different assignments between cores and input ports. Moreover, the impact of an input assignment to the core test time is very low with the non-preemptive test scheduling since the path is reserved for the core. Thus, the test time difference between two partitions with the same pin distribution is also very low, and does not justify the exploration of that space dimension. Figure 6 presents the pseudo-code of the algorithm for the test space exploration.

Procedure Compressed_NoC_Exploration

```
1 Inputs: R = number of NoC nodes, W = number of ATE channels
2 For T=2 to R/2
3   For all valid partitions ki of W into T
4       TestData = Define packet size for each core and each input port
5       Test_time = Compressed_NoC_Schedule(T, ki, TestData)
6       If Test_time < Best_Test_time
7          Best_Test_time = Test_time
8          Select partition (T, ki)
9 Repeat from 2;
```

Figure 6 - Test space exploration algorithm

5. Experimental Results

The proposed scheme was implemented and results were generated for three SoCs based on three ITC'02 benchmarks [23]. The original SoCs are systems d695, g1023 and p93791, which will be called herein d695c, g1023c and p9379c, respectively. For those systems, we used random test patterns to generate the compressed test data, since the test vectors are not provided by the ITC'02 benchmark suite (only their number is given). The random vectors contain around 80% of don't cares bits. Compression results were obtained using the method presented in [14]. We note that uncompressed vectors for industrial circuits usually contain more than 90% of don't care bits, which allows better compression results. However, we have considered a worst-case scenario to evaluate whether the NoC parallelism is well explored. For all systems, a NoC with 32-bit channels is used, and each core is assumed to have at most 32 wrapper scan chains.

The input payload per core depends on the number of bits of each possible test input port, as explained before. Table 1 presents an example of the payload increase for system d695c. Column 2 presents the original payload size for each core (in number of flits) considering the 32-bit bitwidth of the NoC channels. Columns 3, 4 show the payload size for the configuration with three input ports 1, 2, 3 with 12, 10, and 10 bits respectively.

Three initial experiments were performed on system d695c to show that the compression scheme leads to a better utilization of the network resources. For those

experiments, no test space exploration was performed. In addition, the number of output ports is always equal to the number of input ports as required by the non-preemptive test scheduling. However, all output ports are assumed to be 32-bit wide (no output compaction). Table 2 shows the resulting test time (column 2) and the number of ATE input channels (column 3) required for each system configuration. The percentages presented in lines 3 and 4 indicate the difference between that solution and the one presented in line 1.

Table 1 - Payload increase for system d695c

Core	Original Payload (32-bit)	Port 1 Payload (12-bit)	Port 2,3 Payload (10-bit)
1	12	41	45
2	511	1207	1376
3	2400	2507	2507
4	5670	5829	5829
5	6050	13088	14960
6	9594	14270	15727
7	3230	5037	5577
8	4462	4605	4605
9	768	1704	1936
10	370	8171	9354

One can observe from Table 2 that without compression, a 200% increase in the number of ATE input channels is necessary to achieve 58% reduction in test time. With compression, one can keep the number of ATE channels of the original solution and still increase the number of test interfaces. Thus, despite the average increase of more than 300% in the payload size of the cores, test time is reduced by 33% without increasing ATE costs. Note that for this result a single (and random) partition of the input pins has been considered. With the exploration of all possible partitions proposed in Section 4.2 even better test times can be achieved while keeping the ATE costs constant.

Table 2 - Test time results for d695c

System Configuration	Test time (cycles)	number of ATE input channels
1 32-bit ports	36588	32
3 32-bit ports	15293 (-58.2%)	96 (+200%)
3 ports of 12, 10, and 10 bits	24395 (-33.3%)	32 (+0%)

Table 3 presents more detailed results for the three SoCs considering the test space exploration. Let us assume that the number of functional input pins of the system defines the number of available ATE input channels. For each system and for each number of I/O pairs (column 2), we show the number of system input pins (column 1), the test time (column 3) and the number of ATE input channels (column 4) when no compression is used. Similarly, columns 5 and 6 show respectively the test time and the number of ATE input channels for the best partition when compression is used. Column 7 shows the number of bits per input port for the partition that leads to the best test time. One can observe that for a similar ATE cost, test time of the compressed solution is always better (bold numbers in Table 3). System p93791c is an exception, since for two inputs and 118 available pins there is no need to use compression. However, for a given number of ATE channels, the solution that uses the compression scheme achieves a much better test time. Let us consider system p93791c, for example. For a budget of 118 ATE channels, the proposed approach achieves a test time improvement of 32%. Shaded cells in Table 3 show that for a similar test time, the ATE cost is much smaller when compression is used (around 50% for systems d695c and g1023c, and around 30% for p93791c).

Note that test time improvement for systems g1023c and p93791c saturates for a certain number of I/O pairs (4 and 5, respectively). This happens because there is a dominant core in the system that actually defines the test time. In both cases, the dominant core uses a 32-bit input, i.e., no compression is required, whereas the remaining cores share the remaining input ports using compression when necessary. Finally it is interesting to notice in column 7 of Table 3 that in some cases the best partition uses fewer pins than the maximum available. In fact, the algorithm presented in Figure 6 stores the first valid partition that gives the best test time. Information on extra available pins can be further exploited to combine the proposed technique with different packet formats (e.g. [1]). The best partitions indicated in the table also show that the number of decompressors in the systems may be smaller than the number of input ports. Again, dominant cores use the uncompressed ports while less critical cores can increase their test time without penalizing the system as a whole. In any case, although the total number of ATE channels grows with the number of 32-bit output ports, this growth is much smaller when compression is used,

6. Final Remarks

We have proposed the combination of horizontal test vector compression and test scheduling schemes to tackle an important limitation of the NoC-based testing, namely, the number of available test interfaces. A test strategy composed of a test data compression mechanism, a non-preemptive test scheduling algorithm, and a test architecture exploration method was proposed and implemented. Experimental results have shown that for the same ATE cost, test times are improved by 23% in average. On the other hand, for similar test times, an ATE costs reduction as large as 50% can be achieved.
In this paper, only the benefit of test pattern compression on ATE drive channels has been questioned, compaction of test responses can be also be explored as well to deal with limits on receive channels.

Table 3 - Test time X ATE cost with and without compression

System	Number of I/O	No Compression		With Compression		
		Test time (cycles)	# of input ATE channels	Test time (cycles)	# of input ATE channels	Best partition
d695c (32 in)	**1/1**	**36588**	**32**	n/a	n/a	n/a
	2/2	19788	64	**22737**	**32**	4/28
	3/3	15293	96	20945	32	2/15/15
	4/4	9652	128	18067	32	1/1/4/26
	5/5	9652	160	12853	32	1/1/1/2/27
g1023c (56 in)	**2/2**	**23777**	**64**	25453	52	20/32
	3/3	16051	96	**18883**	**56**	12/12/32
	4/4	14453	128	14865	50	6/6/6/32
	5/5	14453	160	**14865**	**56**	6/6/6/6/32
p93791c (118 in)	**2/2**	593225	64	593225	64	32/32
	3/3	395818	96	458164	66	2/32/32
	4/4	311520	128	332843	101	5/32/32/32
	5/5	244550	160	268459	106	5/5/32/32/32
	6/6	244550	192	268459	106	5/5/5/32/32/32
	7/7	244550	224	268459	106	5/5/5/5/32/32/32

7. References

[1] M. Li, W. Jone, Q. Zeng. An efficient wrapper scan chain configuration method for network-on-chip testing. *IEEE Computer Society Annual Symposium on Emerging VLSI Technologies and Architectures,* 2006.

[2] C. Liu, J. Shi, E. Cota, V. Iyengar. Power-aware test scheduling in network-on-chip using variable-rate on-chip clocking. *IEEE VLSI Test Symposium,* 2005 pp:349 – 354.

[3] C. Liu, Z. Link, Z.; D.K. Pradhan. Reuse-based test access and integrated test scheduling for network-on-chip. *Design, Automation and Test in Europe, 2006.* pp:303-308.

[4] E. Cota, L. Carro, M. Lubaszewski. Reusing an on-chip network for the test of core-based systems. *ACM Transactions on Design Automation of Electronic Systems* (TODAES), Volume 9, Issue 4 (October 2004) pp: 471–499.

[5] C. Liu, E. Cota, H. Sharif, D.K. Pradhan. Test Scheduling for Network-on-Chip with BIST and Precedence Constraints. *IEEE Int. Test Conference*, 2004, pp. 1369-1378.

[6] Cota, E.; Liu, C. Constraint-Driven Test Scheduling for NoC-Based Systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* Volume 25, Issue 11, Nov. 2006 pp:2465 – 2478.

[7] A. Jas, B. Pouya, N.A. Touba. Virtual Scan Chains: a means for reducing scan length in cores. *IEEE VLSI Test Symposium,* 2000, pp: 73-78.

[8] L-T Wang et al. VirtualScan: a new compressed scan technology for test cost reduction. *IEEE Int. Test Conference* 2004. pp: 916-924.

[9] I. Bayraktaroglu, A. Orailoglu. Test volume application time reduction through scan chain concealment. *ACM/IEEE Design Automation Conference*, 2001, pp: 151-155.

[10 K.J. Balakrishman, N.A. Touba. Reconfigurable linear decompressor using symbolic Gaussian elimination. *Design, Automation and Test in Europe, 2005*, pp: 1130-1135.

[11] J. Rajski et al. Embedded deterministic test for low cost manufacturing test. *IEEE Int. Test Conference* 2002, pp: 916-922.

[12] L. Li, K. Chakrabarty. Test data compression using dictionaries with selective entries and fixed-length indices. *ACM TODAES*, Vol. 8, No. 4, October 2003, pp: 470-490.

[13] A. Würtenberger, C.S.Tautermann, S.Hellebrand. Data compression for multiple scan chains using dictionaries with corrections. *IEEE Int. Test Conference*, 2004, pp: 926-935.

[14] J. Dalmasso, M.L. Flottes, B. Rouzeyre. Fitting ATE Channels with Scan Chains: a Comparison between a Test Data Compression Technique and Serial Loading of Scan Chains, *DELTA'06*, pp: 295-300.

[15] N. Sitchinava et al. Changing the scan enable during shift. *IEEE VLSI Test Symposium* 2004, pp: 73-78.

[16] H. Tang, S.M. Reddy, I. Pomeranz. On reducing test data volume and test application time for multiple scan chain designs. *IEEE Int. Test Conference* 2003, pp: 1079-1088.

[17] B. Arslan, A. Orailoglu. CircularScan: a scan architecture for test cost reduction.. *Design, Automation and Test in Europe* 2004, pp: 1290-1295.

[18] S. Mitra, K.S. Kim. X-compact: an efficient response compaction technique for test cost reduction. *IEEE International Test Conference* 2002, pp: 311-320.

[19] J. Rajski,et al. Finite memory test response compactors for embedded test applications. *IEEE Trans. on CAD*, April 2005, Vol. 24-4, pp: 622- 634.

[20] A.M. Amory, K. Goossens, E.J. Marinissen, M. Lubaszewski, F. Moraes. Wrapper Design for the Reuse of Networks-on-Chip as Test Access Mechanism. *IEEE European Test Symposium, 2006*, pp:213 – 218

[21] A.M. Amory, K. Goossens, E..J. Marinissen, M. Lubaszewski and F. Moraes. Wrapper design for the reuse of a bus, network-on-chip, or other functional interconnect as test access mechanism. *IET Comput. Digit. Tech., 2007,* v. 1, Issue 3, pp. 197–206

[22] A.M. Amory; F. Ferlini ; M. Lubaszewski; F. Moraes. DfT for the Reuse of Networks-on-Chip as Test Access Mechanism. *IEEE VLSI Test Symposium, 2007*, pp: 435-440

[23] E.J. Marinissen, V. Iyengar, K. Chakrabarty. A set of benchmarks for modular testing of SOCs. *IEEE Int. Test Conference* 2002 pp:519–528.