Improving NoC-based Testing Through Compression Schemes

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

HAL Id: lirmm-00170833
https://hal-lirmm.ccsd.cnrs.fr/lirmm-00170833v1

Submitted on 10 Sep 2007

Improving NoC-based Testing Through Compression Schemes

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

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

[2]Laboratoire d'Informatique, de Robotique et de
Microelectronique de Montpellier
161 rue ada, 34392 Montpellier CEDEX 5
France
{dalmasso,flottes,rouzeyre}@lirmm.fr

Abstract

The effectiveness of the NoC reuse during test is very dependent on the number of test interfaces with the tester. This paper proposes a test scheduling method based on the use of a compression scheme to increase the number of test interfaces with the tester (thus increasing test parallelism) while still reusing available SoC pins and tester channels. We show that the combined approach allows test time minimization with minimal area overhead for systems with very few test interfaces.

1. Introduction

Test scheduling algorithms that use the available NoC as Test Access Mechanism (TAM) have been proposed in [1]. This previous work is based on two assumptions: i) non-BISTed cores need access to an external Automatic Test Equipment (ATE), which is achieved by reusing system functional pins; ii) Let N be the NoC channel width, the interfaces with the tester are N-bit wide. If the last assumption is not true, the test can still be performed, but for every test vector there will be an extra latency. For instance, for an N-bit wide channel and a M-bit interface, N>M, the latency per vector is given by $\lceil N/M \rceil$. Moreover, there remains the problem of the number of channels available in the tester. The more interfaces are used to reduce test time, the more channels are required in the tester, which may not be feasible. Thus, it is clear that while the use of more test sources and sinks can be a potential advantage for increasing test parallelism, system and tester limitations can make it impossible the implementation of the ideal solution including large ATE/Core interfaces. However, NoC–based test approaches may benefit from compression strategies to reduce the amount of test data transferred between the tester and the SoC and thus decrease ATE channel requirements.

Let us assume that system functional I/O pins are reused to connect the system under test to the ATE. Let TI 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 $T = \lfloor F/N \rfloor$. We propose to use a compression scheme presented hereafter to deal with the limited number of test interfaces. Using compression, we can decrease the number of I/O pins dedicated to a test interface and thus increase the number of test interfaces T allowing higher test parallelism. In this paper, we only consider test vector compression. Test response management is out of the scope of this work and not discussed in this paper.

Section 2 reviews the compression technique used for test vectors. Section 3 explains the application of this technique to the NoC-based test scheduling. Section 4 presents preliminary results, and Section 5 concludes with some current and future works.

2. Compression Technique

Let us consider a core including P scan chains and C the number of available ATE channels. The compression principle [2] consists in storing the differences between two scan slices within a test pattern instead of the complete P-bit scan slice. Let $\{S_1, S_2, \ldots\}$ be the initial sequence of scan slices, and $D_i = S_{i+1} - S_i$ modulo 2P be the differences between the two successive slices S_i and S_{i+1}. D_i is stored in the ATE instead of S_{i+1} when D_i can be coded on C bits, otherwise, S_{i+1} is stored in the ATE using $\lceil P/C \rceil$ words of C bits. The compressed sequence is thus composed of C-bit words. These words represent either a difference between two original slices, or the $\lceil P/C \rceil$ subpart of a P-bit original slice.

A decompressor is inserted in the test interface to restore the initial test sequence from the stored one. It is mainly composed of an adder and of a P to C Shift Register/Accumulator (SRA). SRA register is composed of P flip-flops and is able to perform shift operations on C bits. The SRA register is either loaded from the adder using a parallel mode, or from a C-bit interface using a semi-parallel mode based on C-bit shift operations. The parallel mode is used for restoring a slice S_{i+1} from the difference D_i stored in the ATE and the preceding vector S_i currently stored in the SRA. The SRA register can thus be set to $S_{i+1} = S_i + D_i$ in one clock cycle. The semi-parallel mode is used to restore a slice S_i from the $\lceil P/C \rceil$ memory words stored in the ATE. This operation requires $\lceil P/C \rceil + 1$ clock cycles.

We note that whatever the compression technique is, the reduction of necessary ATE channels is achieved at the expense of an increase in test time of individual cores. Test time increases with the ratio of the wrapper bitwidth P over the number of ATE channels C.

3. Compression Applied to NoC-based Testing

Let us assume that one test packet contains one test vector for a given core under test, as shown in Figure 1a. Within the network, the payload width of this packet equals the channel width N, which also defines the maximum number of wrapper scan chains in the embedded core. Thus, each word of the payload corresponds to one scan cycle.
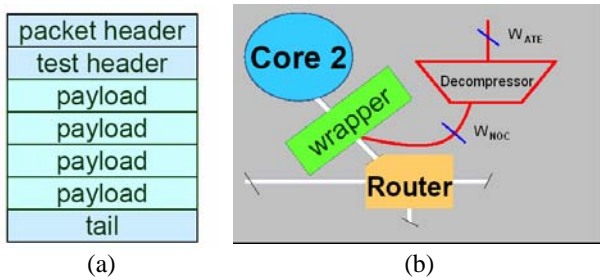

(a)   (b)

Figure 1: (a) test packet format; (b) decompression scheme

In the new compression-based approach, the payload width stored in the tester will be defined accordingly to the bitwidth of each test input port rather than the NoC channel width N. The compression will be applied to each test vector considering the difference between two words of the payload. Tail, packet and test header are not compressed.

In the chip, a decompressor hardware is attached to each test input port, as shown in Figure 1b, so that, in test mode, test vectors are decompressed. Thus, within the network, test packets are decompressed to use the whole available channel bitwidth keeping the original number of wrapper scan chains per core.

The problem is thus to define which input port should be used by each core so that total system test time is minimized.

4. Test Scheduling

A non-preemptive test scheduling is assumed, since it simplifies the wrapper design and avoids the interruption in the scan pipeline. In the non-preemptive test scheduling an access path in the network 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.

The test scheduling algorithm receives: i) 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 free access path to a core, the algorithm evaluates the time required to transfer, decompress, and transmit all test patterns 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 for as long as the test time of the core.

5. Experimental Results

The proposed scheme was implemented and preliminary results are presented in Table 1. System d695 is a simple ITC´02 benchmark that was defined in [3]. This benchmark is based on ISCAS cores, for which test vectors can be generated and compression results can be obtained. A NoC with 32-bit channels is used, and each core is defined to have at most 32 wrapper scan chains. Test vectors were generated using Tetramax tool from Synopsys. The input payload per core depends on the number of bits of each possible test input port. In our experiments with compression, three input ports of 12, 10, and 10 bits, respectively, were assumed.

Three experiments were performed to show that the compression scheme leads to better test time while better reusing the available functional pins. In all experiments 3 output ports of 32 bits each (total of 96 output pins) were assumed.

Table 1: test time results for d95

System Configuration	Test time
1 32-bit input port	63451 cycles
3 input ports of 12, 10, and 10 bits	**48231 cycles**
3 32-bit input ports (total of 96 pins)	25223 cycles

One can observe that using the horizontal compression scheme more test parallelism can be achieved leading to a test time 23% smaller without increasing the number of pins.

6. Final Remarks

A large number of configurations can be considered to explore the test and design space. For instance, decompressors connected to the cores could contribute to the reduction in the size of the test packet (vertical compression) in addition to the horizontal compression. Furthermore, the total number of pins in the chip can be partitioned in several configurations of I/O ports to connect to the ATE, considering both, test time reduction and wiring and decompressor area overhead. Those issues are being considered in the current work.

7. References

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

[2] 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. In: *IEEE DELTA'06,* pp. 295-300.

[3] V. Iyengar, K. Chakrabarty, E.J. Marinissen. Test Wrapper and Test Access Mechanism Co-Optimization for System-on-Chip. *Journal of Electronic Testing: Theory and Applications* 18, pp. 213–230, 2002.