A Technique for Optimization of SOC Test Data Transportation

Anders Larsson, Erik Larsson, Petru Eles and Zebo Peng

Embedded Systems Laboratory
Linköpings Universitet
SE-582 83 Linköping, Sweden

Abstract

We propose a Tabu search based technique for time-constrained SOC (System-on-Chip) test data transportation. The technique makes use of the existing bus structure, where the advantage is, compared to adding dedicated test buses, that no additional routing is needed. In order to speed up the testing and to fulfill the time constraint, we introduce a buffer at each core, which in combination with dividing tests into smaller packages allows concurrent application of tests on a sequential bus. Our technique minimizes the combined cost of the added buffers and the test control logic. We have implemented the technique, and experimental results indicate that it produces high quality results at low computational cost.

1. Introduction

The increasing test application times for SOC (system-on-chip) designs mainly due to the high amount of test data required for testing can be minimized by an efficient organization of the test data transportation. In general, the test data is transported on a TAM (test access mechanism) and a TAM can be an added *dedicated* infrastructure for testing purpose only, or an *existing* functional structure, such as the system bus, for example. Typically, a fixed test time is given for the testing of a system given by, for instance, the ATE memory, and it is up to the designer to design a test solution that does not violate the time constraint.

Several approaches assuming a dedicated TAM for test data transportation have been proposed [3], see Figure 1(a). Harrod demonstrated the use of the AMBA-bus for test purpose [1]. The main disadvantage with a dedicated TAM is the additional routing it requires. Despite efficient scheduling techniques, basically a wide TAM is required to reach a low test time. The general disadvantage with using the existing bus for test data transportation is that the bus operates sequentially. It means the tests are executed one after the other and only one test is active at a time, which leads to long testing times.

We propose a buffer-based architecture that makes use of the existing bus structure for test data transportation, which has the advantage that additional wire routing is not needed, and by introducing buffers we separate the transportation from the application of test data. In our scheme, we divide the test sets into smaller packages, which are sent on the bus. As soon as a package arrives at a core, testing can start. In our scheme, even if the bus is organized sequentially, the application of tests at cores is performed concurrently, leading to shorter testing times.

We make use of an example to illustrate the architectures for (1) dedicated TAM (Figure 1), and (2) our proposed buffer-based TAM scheme using the existing bus (Figure 2). Note, that in all cases, a bus is required in normal operation. A test architecture with a dedicated TAM is in Figure 1. The cores are scan-tested and to ease interfacing with the TAM, each core has a wrapper. In normal operation the inputs and outputs of each core is connected to the bus, while in testing mode the test data is transported on the dedicated TAM. A dedicated TAM for the transportation of test data has the advantage of high flexibility. It also offers the possibility of a trade-off between the test time and the number of wires used in the TAM. Note however, that the bus in the system is not used at all during testing mode (Figure 1(b)).

Figure 2(a) shows an architecture where buffers are introduced between the existing bus and the cores. The advantage with such a scheme is that by inserting buffers, the transportation of test data is separated from the application of test data. The application of a test vector at a core, that is shift-in and capture, takes longer time than sending the test vector from the ATE to the core. The idea with buffers at cores is combined with the division of the test set for each core into small packages of test vectors. It means that as soon as a package of test data has arrived to a core, testing at the core can start. Since the test application at a core takes longer than sending test data, the bus can be used to fed test data to other cores. And, hence, the cores are tested concurrently (Figure 2(b)), leading to a reduced test application time.

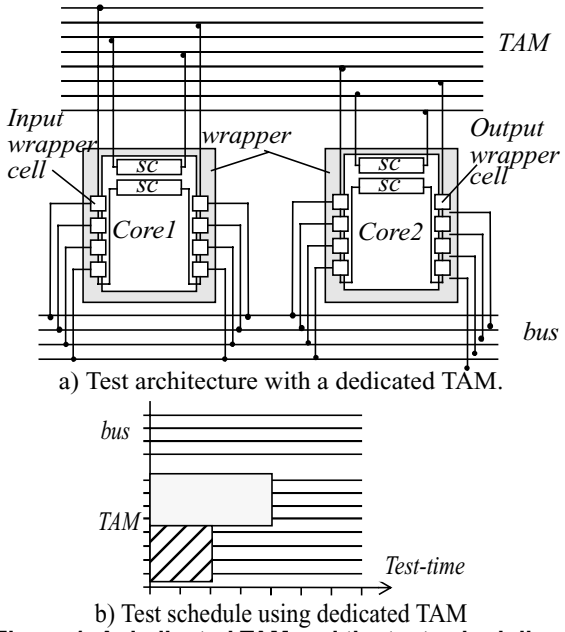
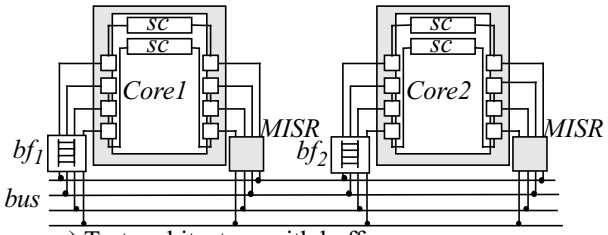
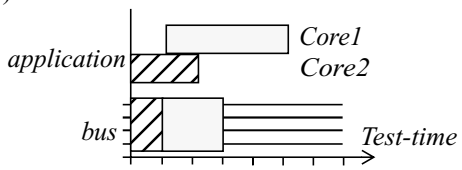


Figure 1. A dedicated TAM and the test scheduling.



a) Test architecture with buffers.



b) Test scheduling and application with buffers

Figure 2. Functional bus and buffers

2. Problem Formulation

Given is a system consisting of a set of cores $C = \{c_1, c_2, ..., c_N\}$, where N is the number of cores, and each core c_i, has a buffer bf_i where bs_i is the buffer size (initially bs_i is not determined). The maximal allowed test time for the system T_{max} is given as a constraint. Also given is the set of tests $T=\{T_1, T_2, ..., T_N\}$, where T_i is a set of test vectors, which is to be applied to the core c_i. For each test T_i, the following information is given: the application-time T_i^{appl}, the transportation-time T_i^{send}, the size (the number of test vectors) of the test T_i^{size}, and the minimum size of a package $T_i^{size\text{-}p}$.

A test T_i, is divided into m_{T_i}packages, each of the same size $T_i^{size\text{-}p}$ where $P=\{p_{1,1}, p_{1,2}, ..., p_{2,1}, p_{2,2}, ..., p_{N,mT_i}\}$. The total cost for the test is computed by a cost function, that consists of the system's total buffer size and the complexity of the controller [6] given as follows:

$$Cost_{Tot} = \alpha \times Cost_{Controller} + \beta \times Cost_{Buffer} \quad (1)$$

where α and β are two coefficients used to set the weight of the controller and the buffer cost.

3. The Tabu Search Based Algorithm

We have implemented a Tabu search based optimization heuristic for the problem. The algorithm consists of three steps: In step one, the tests are sorted according to their application time, t_i^{appl}, and the initial schedule is built.The slack, which is the difference between the end time of the schedule and T_{max}, is calculated. In step two, the initial schedule is improved by distributing the slack between the packages, hence, decreasing the buffer size. After this step the slack is zero. The schedule is then further improved in step three, were a Tabu search based strategy [7] is used to find the best solution. When the Tabu search terminates, the solution with the lowest cost is returned. The algorithm takes as inputs the tests T, a minimal test time possible for the tests, T_{min}, and the maximum allowed test time, T_{max}. T_{min} is the theoritical minimal time needed for transportation and application of tests T, with unlimited buffer and controller cost. This value can be computed by a CLP (Constraint Logic Programming) model in very short time (none of the experiments we have used needed more than 700 ms for computing this value).

4. Experimental results

In the experiment we compared the proposed Tabu search based technique [7] with the approach used in [4] where the CLP-tool CHIP (V 5.2.1) [2] was used. We assume that a maximal test time (T_{max}) is given and we also use CHIP to compute the optimal test time (where cost for buffer and controller are ignored) (T_{min}).

We have used the following four designs; Ex1, ASIC Z, Kime and System L (detailed information for these benchmarks can be found in [5]). The experimental results are collected in Table 1. The execution time needed to find the optimal value of T_{min} where below one second for each example. The cost is minimised according to the cost (Eq. 1) where $\alpha=\beta=1$.

References

[1] P. Harrod, "Testing Reusable IP-a Case Study," *Proceedings of International Test Conference (ITC),* pp. 493-498, 1999.

[2] P. Van Hentenryck, "The CLP language CHIP: constraint solving and applications," *Compcon Spring '91. Digest of Papers ,* 25 Feb-1 Mar 1991 , pp. 382 -387, 1991.

[3] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Test Access Mechanism Optimization, Test Scheduling, and Test Data Volume Reduction for System-on-Chip", *Transactions on Computer*, Vol. 52, No. 12, Dec. 2003.

[4] A. Larsson, E. Larsson, P. Eles, and Z. Peng, "Buffer and Controller Minimisation for time-Constrained Testing of System-on-Chip," *Proceedings of International Symposium on Defect and Fault Tolerance in VLSI Systems*, pp. 385 - 392, 2003.

[5] E. Larsson, A. Larsson, and Z. Peng, "Linkoping University SOC Test Site," *http://www.ida.liu.se/labs/eslab/soctest/*, 2003.

[6] B. Mitra, P.R. Panda, and P.P Chaudhuri, "Estimating the Complexity of Synthesized Designs from FSM Specifications," *Design & Test of Computers,* Vol 10*,* pp. 30-35, 1993.

[7] C. R. Reeves (Editor), "Modern Heuristic Techniques for Combinatorial Problems", *Blackwell Scientific Publications*, ISBN 0-470-22079-1, 1993.

Design	*Nr. cores*	t_{max}	Constraint Logic Programming		Proposed Algorithm		Cost Comparison
			CPU time (s)	Total cost	CPU time (s)	Total cost	
Ex1	3	111	160	62	<1	62	0%
Kime	6	257	27375	274	2	290	+5,8%
Asic Z	9	294	47088	208	2	215	+3,4%
System L	14	623	n.a	n.a	3	811	n.a

Table 1. Experimental results.