Controlling the CAS-BUS TAM with the 1149.1 Features : A Way of Testing Systems on a Chip

Walid Maroufi Mounir Benabdenbi Meryem Marzouki
LIP6 Laboratory
Couloir 55-65, 4 Place Jussieu
75252 Paris Cedex France
Tel. (+33)1 44 27 39 67 - Fax. (+33)1 44 27 72 80
E-mail : {Walid.Maroufi,Mounir.Benabdenbi,Meryem.Marzouki}@lip6.fr

Abstract

This paper describes the control of CAS-BUS, a P1500 compatible Test Access Mechanism (TAM). Boundary Scan features are used to allow controlling of the TAM and the P1500 wrappers. The final architecture characteristics are its flexibility, scalability and reconfigurability. It also allows trade-off to optimize test time and area overhead.

1 Introduction

While electronic systems are becoming more and more complex, testing capabilities face difficulties to follow the design evolution. By integrating complete systems on a single chip, the designers can answer new consumer application demands. Hence, they improve IC performances: large bandwidth, high speed, low power consumption, reduced area... However, these Systems On a Chip (SoCs) introduce new challenges. SoC design is based on the integration of core designs (DSP, microprocessor, RAM,...) provided by different companies. Thus testing these cores, which can be deeply embedded in the chip, can only be done after the SoC has been manufactured, contrarily to PCBs (Printed Circuit Boards) or MCMs (Multi Chip Modules), where chips are pretested and known as good. Three types of features are needed to test a SoC: test sources and sinks, a Test Access Mechanism (TAM) and core test wrappers [1]. Like for PCBs, with the IEEE 1149.1 standard in the past decade, the industry needs some standardization for SoC testing. The IEEE P1500 working group [1] proposes to only standardize the core wrapper. The SoC integrator is then responsible for the choice of TAMs, sources and sinks well adapted to the chip constraints. Many TAM architectures [2], [3], [4] and some test control solutions [2], [5], [6] can be found in the literature.

We have presented in [7] a scalable and reconfigurable TAM named CAS-BUS. This TAM can be controlled by different means but our choice has been to use the well known IEEE 1149.1 features to provide not only the TAM's control but also the wrappers' control at the same time. The first part of this paper briefly presents the TAM architecture and its improvements. The second part is dedicated to the global SoC test control architecture and its benefits. Then, future work is presented before concluding.

2 The CAS-BUS TAM

The CAS BUS (figure 1) is a TAM which main function is to provide access to embedded cores whatever the wrapper is. This TAM is made up of two principal elements:

- **a bus** consisting of N wires,
- **a Core Access Switch (CAS)** (figure 2)

Each CAS selects from the N wires of the bus (e_i) the P ones that will be applied to the core wrapper inputs (o_i). It also connects the P outputs of the wrapper (i_i) to the CAS outputs (s_i). The CAS is composed by a Switch and a CAS Instruction Register (CIR). It is controlled by several signals. Signal c_f can connect the CIR to the Wrapper Instruction Register (WIR). Signals c_i control the Switch.

Thanks to the CAS Instruction Register (CIR), three functional modes are available.

- the **configuration** mode: All c_i are set to 1. A k bit word is shifted in the CIR through the first wire of the bus. In this mode, all CIRs are serially connected. Each loaded word corresponds to a N/P switch scheme. Depending on c_f value, the Wrapper Instruction Register (WIR) can be serially connected to the CIR.
- the **bypass** mode: The N inputs e_i go through the Switch to the N outputs s_i.
- the **test** mode: When the loaded instruction is updated, the routing is made through the Switch. While the P wires o_i (i_i) are correctly multiplexed to (from) the wrapper, the N-P re-

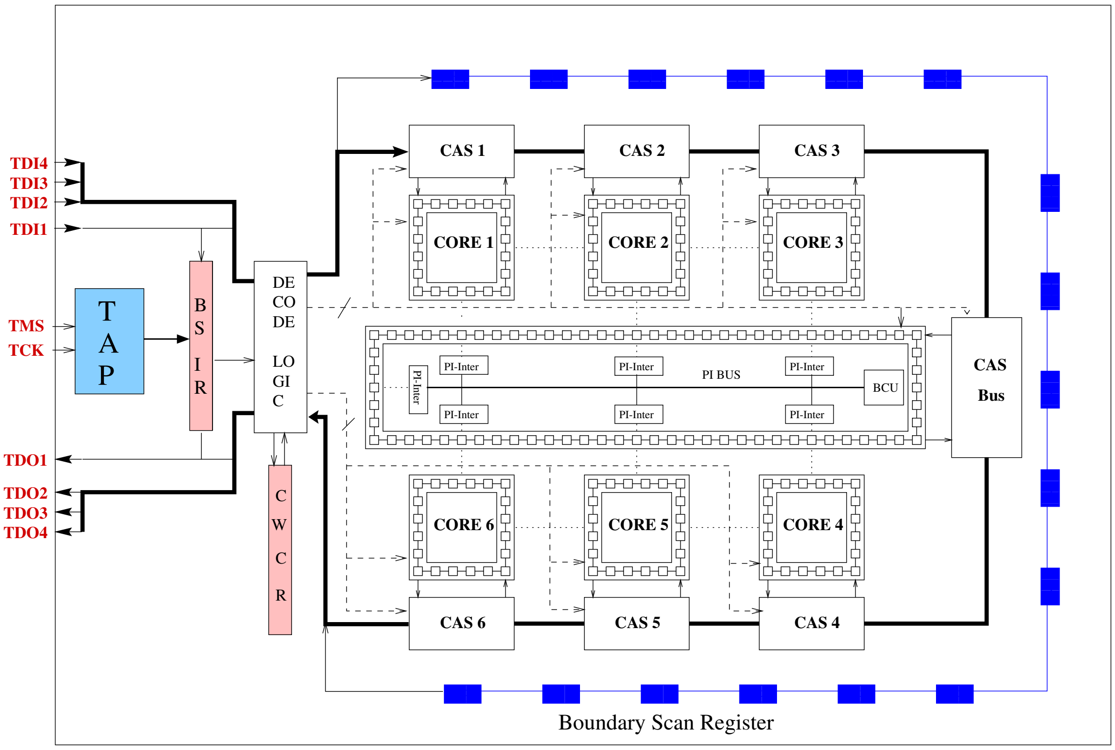


Figure 1. SoC Test architecture

maining ones are bypassed. The core is then ready for test.

The Switch part of the CAS is controlled by the k bit word of the CIR. The Switch task is to route the test stimuli and the test responses to/from the core wrapper. The Switch (figure 3) is made up of one or more decoders and of some logic gates. The decoder controls the multiplexing of the N/P I/Os. By selecting P wires from the N ones existing at the Switch inputs, the decoder must be able to address all the possible N/P combinations.

The number of combinations as well as the decoder area obviously depend on N and P values. Consequently, two different ways of the Switch implementation are proposed:

- Switch with one decoder per input wire (figure 3 (a)).
- Switch with one decoder used for all input wires (figure 3 (b)).

These two architectures allows the SoC integrator to make a trade-off between the total k bits word control and the area overhead induced by the decoder size. In the first case, the total k bit word is longer but the total number of gates is reduced. Synthesizing a decoder for the whole N inputs requires more gates but fewer control bits. A trade-off can be done to optimize the CIRs length and the area overhead. When summing up all CIRs and WIRs length, we found out that it is far smaller than a core scan chain length. In the same way, the area overhead is not significant when compared to a core area. Some results are presented in table 1. The total length of the CIR (k) is given for the two possible switch implementations (figure 3 (a) and 3 (b). This table also give the number of combinations (Cb) each decoder has to provide for each architecture. Cb gives an idea of the decoder's area.

This CAS-BUS TAM architecture is independent from any industrial proprietary SoC architecture. It can be used with any kind of wrappers and thus gives the SoC integrator more freedom for designing its SoC test architecture. The CAS-BUS is scalable, flexible and reconfigurable. More details on this architecture can be found in [7].

3 SoC Test Architecture and Control

The control of this TAM architecture must be simple without degrading the TAM advantages. Two options have been considered: defining an ad hoc test control mechanism or reusing a standard control architecture. The decision to reuse the IEEE 1149.1

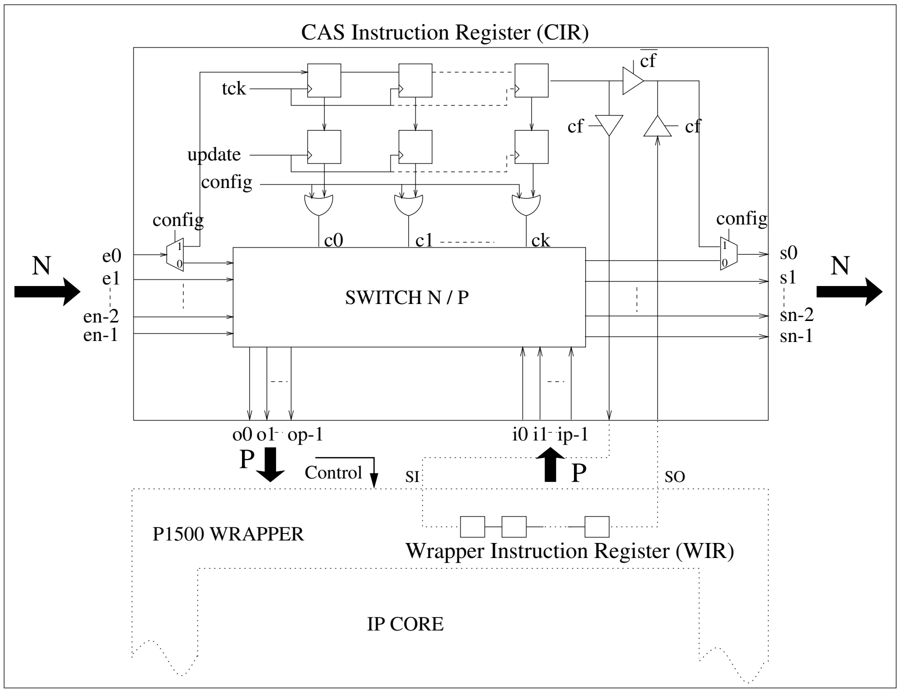


Figure 2. CAS architecture

N	P	k (a)	k (b)	N x Cb (a)	Cb (b).
2	2	4	4	2 x 3	7
3	2	6	4	3 x 3	13
3	3	9	7	3 x 4	106
4	2	8	5	4 x 3	21
4	3	12	8	4 x 4	253
6	2	12	6	6 x 3	43
6	4	18	14	6 x 5	12685
8	2	16	7	8 x 3	73
8	4	24	16	8 x 5	40745
8	6	24	22	8 x 7	2264520
10	6	30	23	10 x 7	8321607
18	10	72	27	18 x 11	67577095

Table 1. Some evaluation criteria for Switch size trade-off

Test Access Port (TAP) naturally came to manage our TAM functionalities. This solution fits well with our SoC test architecture, since it allows TAM control, wrapper control and usual Boundary Scan functionalities at the same time. The test control must allow to manage three mandatory test steps:

- the configuration of the different core wrappers,
- the configuration of all the CASes,
- the application and the analysis of the test vectors.

Two ways of wrapper configuration are possible. The first one is to connect the CAS instruction Register (CIR) to the Wrapper Instruction Register (WIR), thus the WIR configuration is made together with the CIR configuration. The other possibility is to let the SoC test integrator define a specific test strategy to configure the WIR if this one is not connected to the CIR.

Since it has been decided to allow both options, we have extended the normal Boundary Scan architecture with a CAS-Wrapper Coupling Register (CWCR). The CWCR width is equal to the number of CASes. Each bit of this register corresponds to the value of the configuration signal c_f of each CAS (figure 2).

Three new instructions are needed to control the three mandatory test steps. They have been defined as Boundary Scan optional instructions.

- the **CAS-Wrapper_Coupling** instruction (figure 4 (a)).
  It defines which CIR/WIR pairs must be connected in order to configure at the same time a core wrapper and the corresponding CAS. TDI1 and TDO1 are connected to the CWCR. c_f values are shifted in the CWCR to indicate which WIRs will be connected to the CIRs.
- the **CAS_CONFIG** instruction (figure 4 (b))

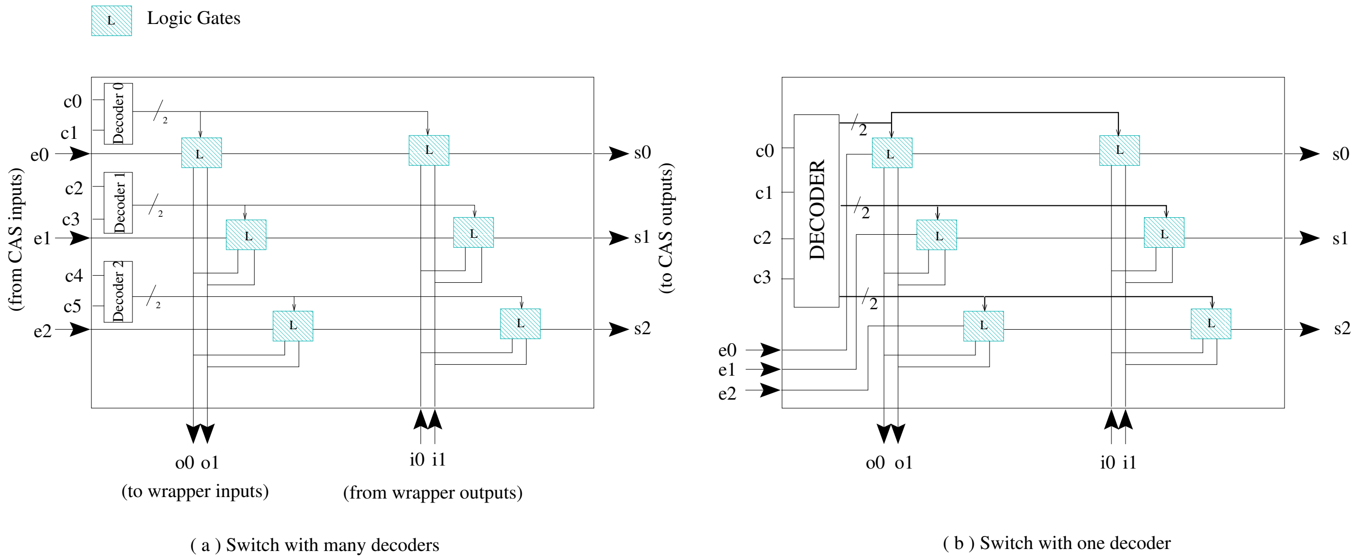


Figure 3. The Switch architecture

This instruction configures all the CASes, together with the wrappers selected with the instruction CAS-Wrapper_Coupling. The config signal is set to 1 to provide access to the CIR of each CAS. TDI1 and TDO1 are respectively connected to input e0 of the first CAS and to output s0 of the last one. The CIR and the WIR are then loaded with the appropriate values, putting each CAS in the correct scheme and each wrapper in the chosen mode.

- the **CAS_TEST** instruction (figure 4 (c)). When loaded, this instruction allows the test vectors to be applied to the different cores and the test stimuli to be propagated to the SoC test outputs. All TDIs and TDOs are connected to the I/Os of the CAS BUS. The set of test vectors can be concurrently shifted in to the IP cores inputs and the responses shifted out from the IP outputs.

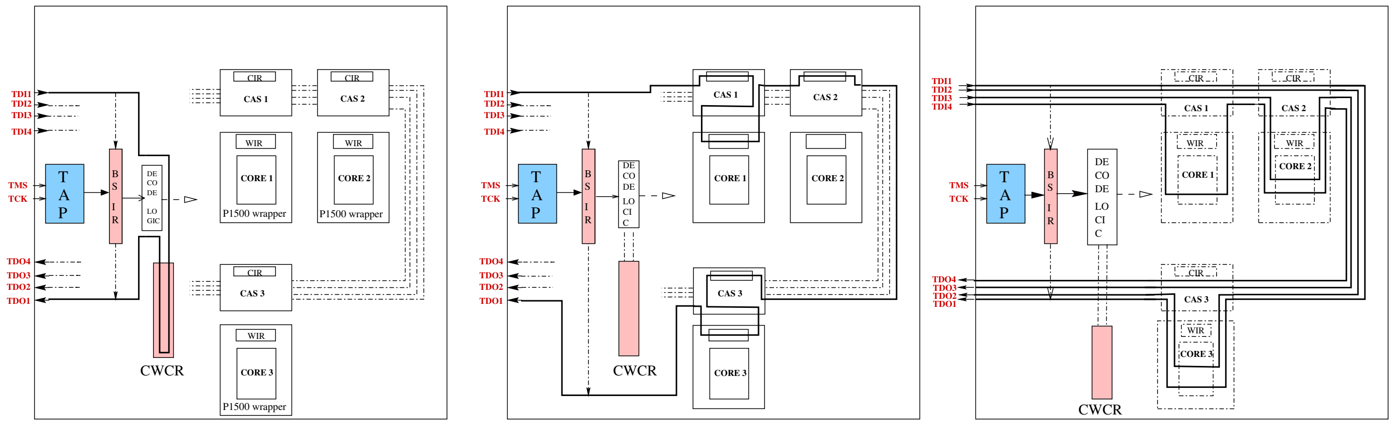
In our modified TAP architecture, TDI and TDO is a bus. This TDI/TDO bus width may vary from 1 (normal Boundary Scan architecture) to N (width of the CAS bus). This width can be chosen by the SoC integrator, depending on SoC pins availability. The simplest case occurs when TDI/TDO bus width is equal to N (figure 4). When the TDI/TDO bus width must be lower than N, then it is necessary to expand the TDI inputs to the CAS inputs. This expansion is made at the expense of test application time.

A decompression/expansion module is currently being developed, in order to be added to the test architecture in case no more than one TDI/TDO couple can be used [8]. This module decodes through TDI compressed test stimuli and applies them to the CAS-BUS inputs. The test responses are compressed in a signature and are analyzed through TDO. The proposed architecture remains Boundary Scan compliant when this decompression/expansion module is added.

Using the IEEE 1149.1 control part with scalable TDI/TDO lets the CAS-BUS TAM keep its main benefits and adds to it flexibility.

The global architecture is not only scalable and flexible but also dynamically reconfigurable. By correctly configuring the CASes, the test programmer can choose during each test session which core scan chains must be serialized in order to optimize their total length. The goal is to have for each of the N wires the same test length. If not, when testing the SoC, don't care stimuli must be applied at TDI inputs to complete the balance between scan chain lengths. Another advantage is that WIRs whose mode remains unchanged between two test configurations can be bypassed during the CAS CONFIG phase. Using this reconfigurability aspect, test time can be saved.

Compared to other approaches described in the literature, the global architecture, TAM plus control features, is very flexible and easily controlled using the well known TAP controller. It allows the concurrent application of test stimuli to many wrapped cores through a scalable bus. Even if they are too different to allow any comparison, architectures like HTAP [5] or TLA [6] have only one TDI/TDO couple and do



(a) CAS-WRAPPER_COUPLING (b) CAS_CONFIG (c) CAS_TEST

Figure 4. The three new mandatory instructions

not allow to test many cores at the same time.

In the CAS-BUS architecture, Boundary Scan I/Os are used not only to control the CASes and the core wrappers, but also to apply test stimuli to cores inputs, contrarily to approaches like [9]. Indeed, even if this architecture includes Boundary Scan features at SoC level like our architecture, the TDI/TDO wires are only used to control the global SoC test, the TAM being directly connected to the SoC primary I/Os.

The CAS-BUS architecture is simple, allows trade-off and provides a complete solution for testing systems on a chip.

4 Future Work

With this architecture, a simple way to control the TAM and the wrappers has been defined, by the way of using IEEE 1149.1 features. However, some improvements can be made regarding the control of the wrappers not configured during the CAS_CONFIG phase. Having a central TAP controller and many smaller ones integrated in the architecture may be an interesting way to improve the global test control. Some other solutions are currently being studied.

Three types of software modules also are in a development phase:

- the trade-off and generation module:
This program will first help the SoC Test integrator in evaluating the optimized characteristics of the test architecture (Nmax, Switch structures...). The choices will be made depending on the core DFT (Scan or Bist), the number and the length of the different scan chains, the test time,... These softwares modules will then automatically generate the appropriate elements of the architecture, with the parameters provided by the trade-off.

- the test plan generation module:
This software will transform a global SoC test plan in a reorganized and optimized test plan. The test plan generated will be a C program where all test sessions and test vector sets will be structured to save test time. The module will evaluate the best CAS configuration for each test session.

- the test plan application module:
The structured test plan generated in the C language will have to be executed from different test platforms (PC, Sun station,..). Through an interface, the test vectors set will be applied to a PCB containing SoCs. This third kind of software modules will run the C program on the desired platform and will manage the test execution through the interface.

5 Conclusion

The global architecture presented in this paper is both P1500 compliant at core level and 1149.1 compliant at SoC level. The CAS-BUS TAM associated with the 1149.1 control offers a complete solution for testing SoCs. This global test architecture is flexible and scalable, allowing test designers and test programmers to optimize test time and area overhead. Trade-off regarding the Switch design, the choice of Nmax and the CAS configuration can be done. Some software modules are currently under development and will allow to automatically generate the architecture features and to drive it.

References

[1] E.J. Marinissen, Y. Zorian, R. Kapur, T.Taylor, and L.Whetsel. Towards a standard for embedded core test: An example. In *IEEE International Test*

Conference (ITC), pages 616–627, Atlantic City, NJ, September 1999.

[2] L. Whetsel. Addressable test ports an approach to testing embedded cores. In *IEEE International Test Conference (ITC)*, pages 1055–1064, Atlantic City, NJ, September 1999.

[3] E. J. Marinissen and al. A structured and scalable mechanism for test access to embedded reusable cores. In *International Test Conference*, Washington, DC, October 1998.

[4] P. Varma and S. Bhatia. A structured test reuse methodology for core-based system chips. In *International Test Conference*, Washington, DC, October 1998.

[5] D. Bhattacharya. Hierarchical test acces architecture for embedded cores in an integrated circuit. In *IEEE VLSI Test Symposium (VTS)*, pages 8–14, Dana Point, CA, April 1998.

[6] L. Whetsel. An ieee 1149.1 based test acces architecture for ics with embedded cores. In *IEEE International Test Conference (ITC)*, pages 69–78, Washington, DC, November 1997.

[7] M. Benabdenbi, W. Maroufi, and M. Marzouki. Cas-bus: A scalable and reconfigurable test acces mechanism for systems on a chip. In *IEEE Design Automation and Test in Europe (DATE)*, pages 141–145, Paris, France, March 2000.

[8] W. Maroufi. A new compression/decompression method for non correlated test patterns: Application to test pins expansion. In *IEEE European Test Workshop (ETW)*, Cascais, Portugal, May 2000.

[9] J. van Beers and Harry van Herten. Test features of a core-based co-processor array for video applications. In *International Test Conference*, pages 638–647, Atlantic city, NJ, September 1999.