Computer-Aided Test Flow in Core-Based Design

V. A. Zivkovic, R. J. W. T. Tangelder and H. G. Kerkhoff

Abstract: **This paper copes with the test-pattern generation and fault coverage determination in the core-based design. The basic core-test strategy that one has to apply in the core-based design is stated in this work. A Computer-Aided Test (CAT) flow is proposed resulting in accurate fault coverage of embedded cores. The CAT flow is applied to a few cores within the Philips Core Test Pilot IC project.**

I. Introduction

The development of semiconductor technology in recent years and the strong expectation, that this trend will furthermore continue, enable the design of complete systems-on-chip (SOC). In order to use the design resources in an efficient manner while building such systems, a new design style, *Core-based* design, has been established. The main point of the Core-based design style is the integration of reusable, parametrized blocks, so called *cores*. Cores, also described as IP's (Intellectual Property), modules or blocks, can appear in hard (layout), firm (netlist) or soft (RTL-level) form. Core-based design has led the IC design community into two groups: core providers and core users. One of the challenges facing design and test engineers in this kind of environment is the elaboration of a comprehensive test strategy. The importance of manufacturing tests for these devices is very obvious if one has their astonishing complexity in mind. Therefore, special attention has to be paid to design-for-testability (DfT) circuitries and test pattern generation of such complex circuits. An efficient high-coverage test approach is crucial to ensure that only good products will be shipped to the customers at reasonable test costs.

There have been a number of academical and industrial approaches to tackle this problem such as the ones described in [1-5], etc. In order to establish the worldwide standardisation regarding the core-test, a number of companies discuss IEEE P1500 [6].

This paper is organised as follows. The second section outlines briefly the core-based test described within the Philips Core Test Action Group - CTAG [7]. Next, the basic flow of the test-pattern generation in an embedded core environment is explained from the point of view of the core provider. The determination of accurate fault coverage in this environment is also included in the third section. The core-based test flow has been applied to a Philips pilot IC and the results are presented in section IV. Section V concludes the paper.

V. A. Zivkovic, R. J. W. T. Tangelder and H. G. Kerkhoff are with the MESA+ Research Institute, The University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands, E-mail: V.Zivkovic@el.utwente.nl

II. Core Test

The principal motive for core-based design and test is the time-to-market reduction. Hence, the precomputed tests for manufacturing defects should be linked to the cores. There are new challenges [5] that must be tackled if one applies core-based test. First, the description of all test-data aspects of the core such as test protocols, patterns, modes, etc. must be available. Second, there must be DfT techniques available that are suited for the core-based design style. Finally, test expansion of core-level tests into IC-level tests is required. The final goal is high fault coverage constrained with the least number of patterns and the least area for DfT.

Usually, the cores are embedded, i.e., the pins of the cores are not necessarily the IC pins. Therefore, the boundary-scan test standard can not be used directly for the core test resulting primarily from the access problem. The core-test standard [7] copes with the test-access mechanism using so called *test-shells*, which are wrapped around each core (Fig. 1), isolating in that way the core from the rest of the system. The inner part of the core is labelled IP. The term "core" is now related to the IP equipped with the test-shell.

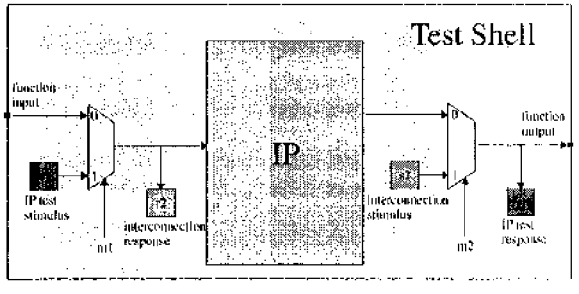

Fig. 1. The behavioural view of the test-shell

There are four basic modes of the test shell:

- *Function Mode*; this is for normal operation of the circuit, i.e., the test shell is transparent.

0-7803-5235-1/99/$10.00 © 1999 IEEE

- *IP test mode*; the IP surrounded with this test shell is now being tested. The test-shell ensures that the test stimuli coming from the IC inputs are transported to the IP module under test and vice versa, i.e., that the test responses are transported from the IP module to the IC outputs.
- *Interconnect test mode*; in this case, the interconnections and the glue logic between the cores are tested. The test stimuli from the IC level for the interconnection logic succeeding the outputs of the core are brought via the test shell. On the other hand, the test responses that come from interconnection logic in front of the inputs of the test-shell are captured and transported to the IC level.
- *Bypass mode*; now, the test stimuli for the other cores are transported via the test-shell.

The previously described modes are applied to the test-shell only, and they do not depend on any test mode of the IP surrounded with that test-shell.

Following the above drawing and explanations, it is obvious that the test-shell consists of flip-flops connected into scan chains, multiplexers and some logic. Of course, the actual implementation of the test-shell is flexible and depends on the optimisation of the area and throughput. The test-shell modes are controlled via a standardised test-control mechanism (TCM). More details regarding the core-test can be found in [7].

III. Basic Core Test Strategy

The result of the test pattern generation for a core has to be straightforward in order to simplify the usage for the core user ("plug and play"). The test-data access circuitries expressed as the test-shells enable the execution of these test patterns with respect to the core under test. The DfT circuitries from the test shell that are used to isolate inputs, outputs and input/output pins of the cores are shown in Figs. 2, 3 and 4 respectively [8].

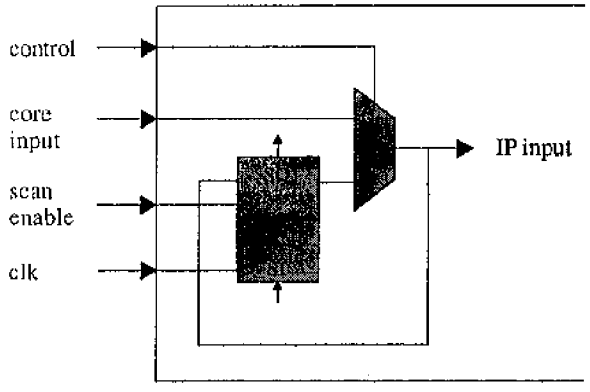

Fig. 2. The DfT input structure

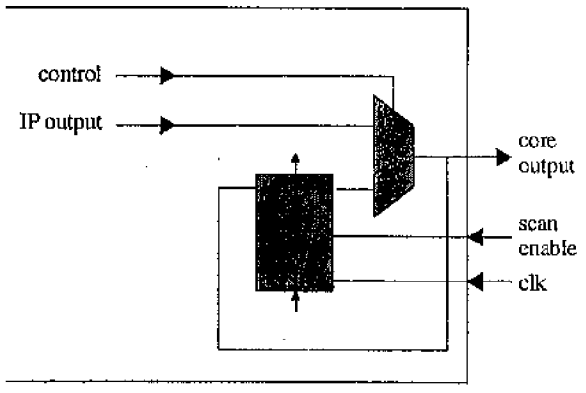

Fig. 3. The DfT output structure

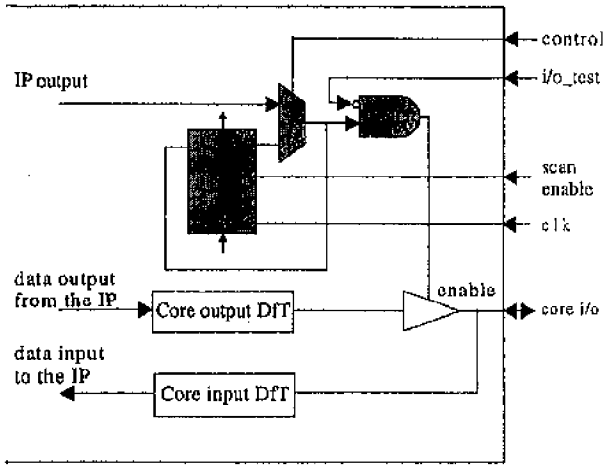

Fig. 4. The DfT circuitry for I/O isolation

The additional logic in Fig. 4 is necessary since during the IP-test mode, the on-chip bus must not influence the cores. This is provided with the i/o_test signal forcing the tristate buffer into Z-state.

The flip-flops in the above circuitries are connected into scan chains and they can be used for test-stimuli application or test-response capturing. The used DfT structures remind on the circuitries from the boundary-scan technique [9]. Nevertheless, there is strong difference between them because of the two reasons. First, the core-test verifies the interconnections between the cores as well as their internals. Second, the number of extra pins on the boundaries of the cores is not limited as is in the case of the boundary-scan standard. The cores are still to be designed and one is free to use as many pins on the boundary of the core as required.

However, similar to the boundary-scan test where the chips are manufactured and tested, the cores must also come with generated test patterns resulting in accurate and high fault coverage. This is the task of the core provider, and hence the core user knows what to expect.

The fault coverage of the core is related to the fault coverage of the IP equipped with the test shell. However, one is not free to generate the test patterns and calculate the fault coverage by simply running the ATPG on the core. The reason for this is that the cores are embedded and the test-shells around the cores have different modes with forced values on input pins. Therefore, it is necessary to split the ATPG run according to the different test-shell modes. The test generation for the inner part of the core (IP) will be carried out when the test-shell has a controlled value set for the IP-test mode. IP test patterns will not cover the faults in the test-shell and glue logic. These faults can be detected during the interconnect test. Hence, another ATPG run is required when the control signals of the test-shell are fixed for the interconnection test mode. Of course, the interconnections between the cores and glue logic are minor part of the design and they will require fewer test patterns compared to the IP test. The actual test patterns and exact fault coverage of the embedded core will be then obtained by combining these two patterns.

The core-based test generation flow that results in the core patterns and determines the real fault coverage for any environment in which the core can be embedded is shown [10], [11] in Fig. 5. The flow proposed in this paper is in full agreement with the CTAG standard. All CAT tools in this flow are developed with features to ease the core-based test flow. Hence, the flow is highly automated.

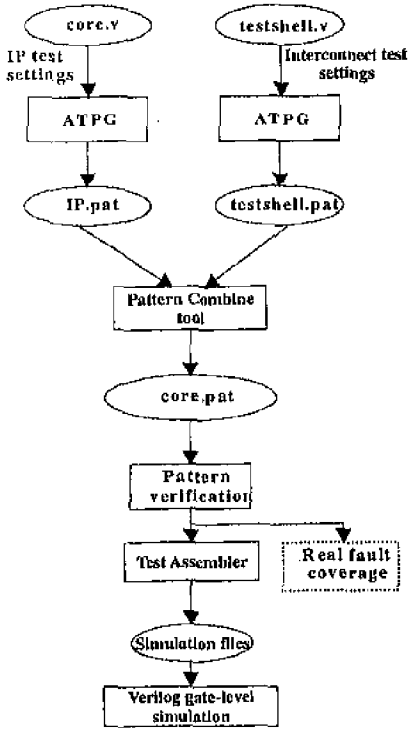

Fig. 5. The new proposed Computer-aided-test flow [10], [11]

This flow can be regarded in the following way. The "core.v" and "testshell.v" are scannable netlists of the core and test-shell netlist, respectively. According to the previous explanation, two ATPG runs are required in order to generate the test patterns for the IP and the test-shell. By combining them, one will obtain the patterns that can be used to test the complete core (Fig. 6).

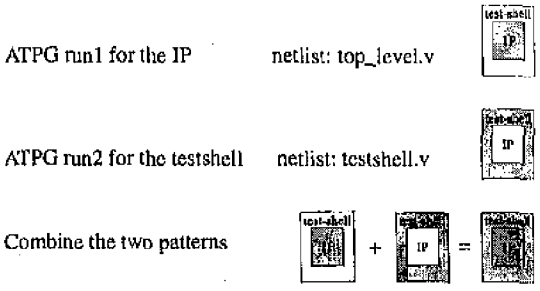

Fig. 6. Core test scheme

A pattern verification tool calculates the real fault coverage of the core by applying the patterns to the core. A test assembler can subsequently generate simulator test benches for many simulators with both stimuli and response information.

The flow depicted in Fig. 5, carried out by the core provider, provides the core user with the test patterns and the fault coverage of each core in the chip. The core user now only has to expand the core test patterns and to implement the top-level infrastructure by the test-pattern expansion tool [5].

IV. Results

The proposed CAT flow for the test-pattern generation and exact fault coverage calculations of the cores have been applied to a Philips pilot IC project chip of which the main characteristics are shown in Table I.

TABLE I

THE MAIN CHARACTERISTICS OF THE PILOT IC

process	CMOS 5M
feature size	0.35µm
total die area	50 mm^2
number of cores	8
complexity	$\approx 20\times10^6$ transistors
number of the test-patterns	3500
size of the largest core	21.4 mm^2
size of the smallest core	0.45 mm^2

The fault coverage of the cores varies between 98.5 and 99.94 %. The obtained results are much better as compared to a chip of similar complexity and functionality as

reported in [12] with the average fault coverage of 91% and more than 4000 patterns. Note that the number of test patterns is obtained after simple addition of the test patterns of each core. Hence, the test patterns have still to be expanded and compressed. However, it will not increase the number of test patterns, since only the test-protocols will be changed, not the test data.

The DfT area overhead, another constraint that has to be taken into account, was also quite low. It amounted to 7.7% of the total chip area. 4.5% is due to the core internal DfT and 3.2% results to the test-shells around the core which is quite acceptable. It has been noticed during the design that the small sized cores require relatively larger area for DfT. This occurs since the test shell has the most impact on a small core with many ports. The DfT area overhead for the smallest core is 30.2% while the area overhead due to the test shell for the largest core was only 0.5 %. Therefore, further improvement of the core test strategy will focus on reducing the DfT area overhead for the smaller cores. Of course, that reduction may not influence the already achieved efficiency in terms of the test patterns and fault coverage.

V. Conclusion

The basics of the core test proposed by Philips CTAG group is explained in this paper. The so-called test shell is introduced providing the infrastructure for the test and isolation of embedded cores. The test of the embedded core consists of the test of the inner part of the core (IP test) and the test of its interconnection together with the test shell. The automatic test-pattern generation flow according to the core test has been proposed and implemented. It results in a test pattern set and accurate fault coverage of the core that can be used in an arbitrary environment. The proposed flow has been used in the Philips pilot IC project. The obtained results with respect to the number of the test patterns and the fault coverage were better compared to another similar design that has been done at the other place. Also, the total DfT area overhead was quite acceptable.

Acknowledgement

The large part of this paper is the result of the practical work carried out within Philips Semiconductors, Eindhoven. The authors acknowledge the contributions of R. Arendsen, E. J. Marinissen, M. Lousberg and all employees of the Philips Semiconductors ASG/ESTC group.

References

[1] R. Chandramouli, S. Pateras, "Testing Systems on a Chip," *IEEE Spectrum*, pages 42-47, November 1996.

[2] P. Varma, S. Bhatia, "A Structured Test Re-Use Methodology for Systems on Silicon," *Proc. of the 1st IEEE Workshop on Testing Embedded Core-based Systems*, Washington D.C, November 1997, paper 3.1.

[3] A. Jas, N. Touba, "Test Vector Decompression via Cyclical Scan Chains and its Application to Testing Core-Based Designs," *Proc. Int. Test Conf.*, October 1998, Washington D.C, pages 458-467.

[4] F. Beenker, B. Bennetts, L. Thijssen, *Testability Concepts for Digital IC's – The Macro Test Approach*, volume 3 of *Frontiers in Electronics Testing*, Kluwer Academic Publishers, Boston 1995.

[5] E.J. Marinissen, G.E.A.Lousberg, "Macro Test: A liberal Test Approach for Embedded Reusable Cores," *Proc. of the 1st IEEE Workshop on Testing Embedded Core-based Systems*, Washington D.C, November 1997, paper 1.2.

[6] IEEE P1500 Web Site, http://grouper.ieee.org/groups/1500/.

[7] E.J.Marinissen, R. Arendsen, G. Bos, H. Dingemanse, M. Lousberg, C. Wouters, "A Structured and Scalable Mechanism for Test Access to Embedded Reusable Cores," *Proc. Int. Test Conf.*, October 1998, Washington D.C, pages 284-293.

[8] R. Arendsen, M. Lousberg, "Core Based Test for a System on Chip Architecture Framework," *Proc. of the 2nd IEEE Workshop on Testing Embedded Core-based Systems*, Washington D.C., October 1998, paper 5.1.

[9] IEEE Computer Society, *IEEE Standard Test Access Port and Boundary-Scan Architecture – IEEE Std 1149.1 – 1990*, IEEE, New York, 1990.

[10] Various authors, "*Core-based Test: Cookbook*," Philips Internal publication, 1998.

[11] V.A. Zivkovic, R.G.J. Arendsen, "Accurate Fault Coverage Determination in Core-based Test," ASG/ESTC/98.7156, October 1998, Philips Internal Documentation.

[12] J. Saxena, P. Policke, K. Cyr, A. Benavides, H. Malpass, F. Ngoh, "Test Strategy for TI's TMS320AV7100 Device," *Proc. of the 2nd IEEE Workshop on Testing Embedded Core-based Systems*, Washington D.C., October 1998, paper 3.2.