Modelling SoC Devices for Virtual Test Using VHDL

Marco Rona, Gunter Krampl
Infineon Technologies AG, Microelectronic Design Centers Austria, Villach

Abstract

Virtual Test (VT) is a new technique to cut the time-to-market especially for SoC products that inherently contain complex mixed-signal blocks. VT allows debugging test programs in a simulation environment if a fast and sufficiently accurate IC model can be made available. VHDL behavioural models turned out to be a very promising approach to cover both the needs of designers for the sign-off simulation on chip level and of test engineers for VT. The trade-offs between modelling effort, simulation performance and accuracy of results will be discussed for VT applications based on an industrial example.

1. Introduction

The traditional design style, consisting in the succession of device design, test developing and its debugging on the tester is no more suitable for today's needs. Therefore a new paradigm, Concurrent Engineering, has been introduced, whose underlying concept is the parallelism of device design and test development phases.

For mixed-signal SoC the most promising approach seems to be the creation of a virtual environment in which the Device Under Test (DUT), as well as the Automatic Test Equipment (ATE), are substituted by software models. This new technique, known under the name of Virtual Test (VT), allows the debugging of test programs through simulation, avoiding the need of expensive ATE time and of real devices.

The work here presented has been funded under MEDEA by the Austrian government (Forschungsförderungsfond no. 801392).

2. Mixed-signal SoC model

VT implies the need of a complete DUT software model, able to substitute the real device. From the test engineer's perspective such a model should be as much equivalent as possible to the real SoC, providing the same interface to the test program. This can be done by providing a hierarchical model with a top-level view in which all, and only, the signals corresponding to DUT pins appear. Important is also the completeness of the model, which means that at least all the test-relevant parts have to be modelled. Furthermore the whole chip has to be modelled, that means, for example, that if a part is replicated several times, it has to be modelled the same number of times.

Since mixed-signal SoC are composed of an analog and a digital part, which directly interact with one another, their DUT model has to describe both parts, as well as their interactions. For the digital blocks the question is not the modelling technique, since VHDL or Verilog are normally used, but the abstraction level. Since we want to simulate test programs, which can take several milliseconds at the physical ATE, the performance of the model is extremely important. Therefore, as long as we consider only functional tests, models at a high level of abstraction, that is VHDL behavioural models, seem to be the best choice.

Also for the analog blocks we have to reject all the low-level techniques, and we have rather to model at a very abstract level. We could use ad-hoc languages and tools, such as COSSAP, but in this case we need an analog and a digital simulator, coupled together [1]. But for VT we simply need functional models, that is, models that describe the behaviour of the analog blocks rather than their internal structure. As the consequence we can use the same technique already used for the digital blocks, that is the VHDL language. Indeed, it is quite easy to model analog blocks in VHDL in terms of their function. The VHDL Real type is suitable for representing analog signals, and the function of a basic block can be described in terms of its transfer function. For example, a signal amplifier can be modelled as the product between input signal and gain. We can write a library of simple analog blocks (filters, decimators, amplifiers, etc.), and then combine these blocks for modelling more complex devices. This kind of models can be improved with the addiction of other features. But, acting in this way, we always have to look for the optimal trade-off between model accuracy and its performance, by modelling only what is really needed, with only the necessary accuracy, avoiding both useless parts and excesses of accuracy.

3. The concept of virtual tester

To perform VT we need something that generates stimuli for and captures responses from the DUT. We can imagine a lot of different scenarios: from an ATE emulator, able to understand test programs as they are, to the description of test programs directly in VHDL, with varying levels of ATE dependency. In this paper we present a solution, which is ATE-dependent at its beginning, but which soon becomes ATE-independent. It is based on three main functions: (1) the conversion of the test program into an intermediate format, (2) a VHDL block able to generate stimuli for and capture responses from the DUT, named virtual tester, and (3) a library of virtual mixed-signals instruments.

With the conversion of the test program into a simple, universal and completely ATE-independent intermediate format, we reach the ATE independence. The virtual tester is a VHDL component able to generate digital stimuli and capture digital responses in a way similar to the one followed by most industrial ATEs. All the information it needs to work (pin timing, stimuli, expected responses etc.), are coded in the intermediate format and read from the virtual tester at the beginning of the simulation.

For a mixed-signal test, besides timing, stimuli and expected responses, we have to generate and capture digital and analog waveforms. For this reason our solution contemplates a VHDL library of virtual mixed-signal instrument models able to generate or capture signals in a way similar to the one followed by the real ones. They read (or write) the samples of the waveforms from (to) files and are triggered by control signals coming from the virtual tester component.

In order to perform a mixed-signal test simulation a VHDL testbench, composed of an instance of the DUT model, an instance of the virtual tester and as many instances of mixed-signal instruments as needed, has to be written. Since each part of the testbench is entirely written in standard VHDL and all the external information it needs are directly read from files, any commercial VHDL simulator can perform the test simulation.

Our solution works also in co-simulation environment, as far as all the DUT pins are on the VHDL side, since they have to be connected to the virtual tester component.

4. Re-usability, design efforts and additional issues

In Chapter 2 we have postulated the need of a complete software model of the entire chip. Although such a model is often created in the product definition or IC sign-off verification phases, normally it doesn't have all the characteristics we need. For example it may not be written in VHDL, may be too abstract or even incomplete, etc. Since it doesn't make sense to write a new model from scratch only for VT purposes, we need to upgrade the design flow and make mandatory the delivery of behavioural level models suitable for VT. What remains to be done to prepare VT execution is to convert the test program in the intermediate format and to write the VHDL testbench. But these tasks can be easily made automatic with the use of conversion utilities, testbench generators and scripts.

Some words on performance. As test case we used a 4-channel AD/DA converter with data and voice separation. The digital part (interpolators, decimators and I/O interface), is designed in VHDL, while the analog one (A/D and D/A converters and analog filters), is modelled in COSSAP. The two simulators have been coupled with an ad-hoc coupling mechanism. This chip has 58 pins, while the test has about 6700 vectors, uses 12 mixed-signal instruments and works for about 2 ms. The entire simulation has taken about 200 seconds, that is 100 sec/1 ms.

5. Conclusion

A complete and very fast solution for mixed-signal VT has been presented. Backbones of the concept are the conversion of ATE-specific test programs into an ATE-independent intermediate format, a virtual tester modelled in VHDL able to generate stimuli for and capture responses from the DUT, and several mixed-signal instrument models triggered by the virtual tester and able to generate and capture waveforms. A complete software model of the DUT is needed and, though in the ideal case this model is entirely written in VHDL, our solution works also in co-simulation environments.

In order to minimise the required efforts, there is a need for some adaptation of the standard SoC development flow to enable the reuse of system level models used in the product definition phase as well as of IC-level models used for sign-off simulation.

Tested in the worst case with the use of two simulators and a complex mixed-signal chip, this solution shows performances in the order of 100 seconds computer time for 1 ms of real time.

6. References

[1] G. Krampl, M. Rona, K. Einwich, "Model and Simulator Coupling for Virtual Test", 6th IEEE International Mixed-Signal Testing Workshop, Montpellier, France, 2000.