ipPROCESS: Using a Process to Teach IP-core Development

Marília Lima, André Aziz, Diogo Alves, Patrícia Lira, Vitor Schwambach, Edna Barros
Informatics Center
Federal University of Pernambuco
Recife, PE, Brazil 50740-400
{msml, aaca, djca, pfal, vsc, ensb}@cin.ufpe.br

Abstract

The reusing of Intellectual Property cores has been an alternative to the increasing gap between design productivity and chip complexity of emerging System-on-chip (SoC) designs. But the design of IP-cores has its own challenges like portability, reusability, standards interfaces, well-defined and useful documentation, easy to integration and so on. All these characteristics together make the design of an IP-core a complex task and in this way teaching this discipline has became a new challenge for educators. In this paper we present an experience about how the utilization of a well-defined development process can be used to facilitate and speed-up students learning.

1. Introduction

The design of complex ICs based on pre-designed and pre-verified IPs is emerging to solve known problems in ICs design [4]. Aiming to guarantee the quality design of each component of the system, it is essential to teach IP-core and System-on-Chip based on industry standards, design methodologies and processes. Aligned with these demands the Brazil-IP Network has developed a process for IP-core development as a result of the Fenix Project [1].

This article presents the ipPROCESS, a process for IP-core development with implementation in FPGA. This process has been used in system design training of undergraduate students in the institutions involved at Brazil-IP Network. The goal of learning IP design by using a well-defined design process is to facilitate and speed-up students learning, once the process is well-defined in terms of activities, each team members roles, what must be done and when. It is expected that a process defined in such way provides a more detailed view of the entire flow, and guides the students more clearly towards what must be done during the development of the IP-core. Result of joint effort of Brazil-IP members, the ipPROCESS process has as its main pillars: iterative and incremental design, UML-RT diagrams [5], use of coding guidelines, functional verification and pair programming [2, 4]. The next sessions present the process in more details and shows how it is being used for teaching IP-core design.

2. ipPROCESS

At the present version of ipPROCESS are represented five major workflows of an IP design, namely: Requirements, Analysis & Design, Implementation, Functional Verification and FPGA Prototyping. In other words, every workflow groups a set of related activities.

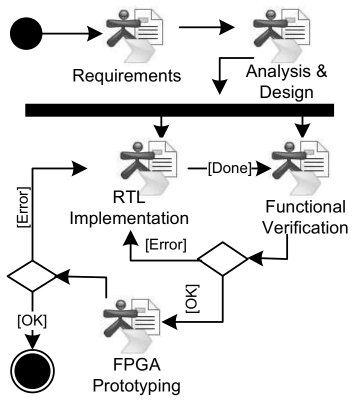

Figure 1. The ipPROCESS Workflows.

The general IP-core development flow is shown in Figure 1 according to the previously mentioned workflows. Development starts capturing the requirements of the IP-core in order to define the design based on the analysis of the identified requirements. Once the design is done, the team can be divided into two sub-teams: the verification and the implementation teams. Verification team shall build the verification model, while design team does the implementation (Verilog, VHDL, SystemC or any other HDL) of the modules that compose the IP-core. After the IP-core implementation is done, it ought to be submitted to functional verification. Thus, in case the IP-core implementation passes the verification and is considered functionally correct, we advance to the next step, FPGA implementation and testing.

Next we present a brief description of the mentioned workflows: (1) Requirements: the main purpose of this workflow is to establish and to maintain agreement with the customers and other users on what the IP-core should do and to provide system developers with a better understanding of the system functional and non-functional (like performance, power, size, cost etc) requirements;(2) Analysis & Design: the purpose is to transform the requirements into a design of the IP-core and to adapt the design to match the implementation environment; (3) Implementation: the purpose is to define the organization of the code, to implement the design elements and perform unit tests on the developed modules; (4) Functional Verification: this workflow focuses on evaluating the IP-core quality through the following activities: finding and describing defects in the implementation, validating the IP-core functionality and also validating the requirements are implemented as specified and (5) FPGA Prototyping: the purpose is to synthesize the implementation into a FPGA and validate the IP-core requirements as specified in the Requirements workflow.

Every workflow mentioned above is defined in terms of activities, roles and artifacts to be created (e.g. code lines, documents, UML diagrams etc). Moreover, to facilitate and speed-up the execution of each workflows activities, templates have been defined for the documents to be created, tool mentors to some tools and checklists to the activities. More detailed information of each workflow and the description of the process itself can be found at the website http://www.brazilip.org.br/ipprocess.

The ipPROCESS has been used for training undergraduate students participating in the Fenix project. In this paper we present as an example the development of the 8051 Microcontrollers IP-core. The team was composed of nine students (divided in sub-teams) and the design process has taken fourteen months. After the requirements for the 8051s IP-core has been defined, the project was partitioned onto the following modules: CPU, USART, IO Ports, Interrupt Manager, Timers and OCP-IP Interface (standard interface among IP-cores [3]). Thus, each sub-team was responsible for the detailed specification of one of the modules. In this stage documents were produced from the available templates and some UML diagrams were created to facilitate the understanding the functionality of each module. After detailing and specifying the communication interface among the modules, the team started the verification and implementation stages. On those stages we adopted the strategy of pair programming [2], that way each module has been implemented by two students. It is important to notice that each pair specified one module, implemented other module and verified another one. This strategy allowed us to identify the specification errors sooner and guaranteed to each team member a broader understanding over the core being developed, thus minimizing the communication and integration errors among the cores modules. After implemented and verified, the sub-team that implemented a given module (soft core) was responsible for synthesizing, implementing and testing it in Xilinx Virtex II XC2V1000-4FG456C FPGA. This implementation of the 8051 Microcontroller has 26,539 lines of SystemC RTL code and to verify it was developed 817,623 test cases (including random, compliance and corner) and 10,566 lines of SystemC TL code were written. The core occupies 40% of 4-input LUTS and 9% of slice flip-flops of the FPGA and has been tested on-chip to a frequency of up to 12MHz.

2.1. Conclusion and Further Work

The process proposed by Brazil-IP Network, the ipPROCESS, has been defined in terms of activities, roles and artifacts that extend from the understanding of what the IP-core should provide until its final FPGA implementation. Its utilization in the training of undergraduate students, as mentioned previously, has shown that document templates, pre-validated coding standards, practical tool tutorials and guidelines are very useful during the learning process. The use of the ipPROCESS makes easier and speed-up the learning process, and also supports team work. Furthermore, some strategies used to define the ipPROCESS as pair-programming, use of a coding guideline and functional verification [2, 4] has led to the creation of IPs with improved quality, fewer errors and with well-defined interfaces.

Nowadays the ipPROCESS is being taught in regular undergraduation and graduation classes where the challenge is even greater: guarantee that the students experience all the stages of the development of an IP-core in only few months. One next step is to specify a workflow to define and group the activities related to IP-cores distribution, in other words, what must be delivered along with the code so that the IP-core may be easily reused by third-parties [4].

References

[1] Brazil ip network. http://www.brazilip.org.br.
[2] extreme programming. http://www.extremeprogramming.org.
[3] Open core protocol specification. http://www.ocpip.org.
[4] M. Keating and P. Bricaud. *Reuse Methodology Manual*. Kluwer Academic Publishers, 2002.
[5] P. Kruchten. *The Rational Unified Process*. Addison Wesley, July 2000.