A CORE-INTEGRATION EXAMPLE IN A SYSTEM-ON-CHIP DESIGN APPROACH

Carlos Rodrigues, Juliana Xavier, Viviane Aureliano, Manoel Lima
{cemr, jmcx, vcoa, mel}@cin.ufpe.br

Centro de Informática - Universidade Federal de Pernambuco
Cidade Universitária, Recife, PE – CEP 50732-970

ABSTRACT

The fast growing complexity and short time-to-market of embedded system's designs, besides the great increase in capacity of today's chips, are mobilizing the industry towards to the development of System-on-Chip (SoC) solutions. However, as the complexity of such systems grows, the problems associated with the integration of the IP-Cores necessary to form the system arise as a major problem in the design flow. In this paper, we propose a short design flow to help designers to gain productivity by using the SoC approach, while avoiding integration problems by introducing automatic tools with this purpose in the process. An example is also presented.

1. INTRODUCTION

System-on-Chip (SoC) is a major revolution taking place in the design of integrated circuits due to the unprecedented levels of integration possibility. The key concept in SoC design is that a chip can be designed rapidly using third party and internal IP-Cores, where IP-Core refers to pre-designed behavioral or physical descriptions of a standard component.

However, the increasing complexity of such systems has brought a major problem to engineers: integration of heterogeneous cores has become a bottleneck of the design flow [1, 2], compromising the tight time-to-market constraints. In this way, the use of standardized interfaces in the IP-Cores [2] combined with modern CAD tools that automate the integration process [3] is essential to successful embedded systems designs.

Besides, the adoption of reconfigurable chips, like FPGAs (Field-Programmable Gate Arrays), as intermediate prototyping platforms, can help the system designer to make improvements to the system and ensure that a design meets its requirements before it is implemented as an ASIC (Application Specific Integrated Controller). SoCs, when implemented in such platforms, are commonly known as SoPC (System-on-Programmable-Chip).

In this work, a top-down SoC design flow is presented, aiming the development of a complete system to control a small robot, based on the Altera Nios Processor [5]. This methodology goes from the initial system specification, followed by IP-Cores implementation in VHDL, to their integration using Altera SoPC Builder tool [3] and further prototyping on an Altera APEX20KE FPGA family platform board (Excalibur) [6].

In section 2, a design flow overview is presented. Sections 3 and 4 present results and conclusions respectively.

2. METHODOLOGY OVERVIEW

The first phase of the methodology consists of the system specification, in which the main components of the system are identified. In this phase, a hardware/software partitioning is made, as well as the choice of the IP-Cores that will be used and the common interface that they should have in order to be integrated with the system.

The SoC architecture, Excalibur, is based on the Altera Nios Processor, the central processing unit, because of two main reasons. First, it is an extremely flexible and powerful RISC processor core that can be easily customized according to user needs. Second, it is totally compatible with the Altera Avalon Bus [2], which has a very simple specification, thus allowing us to easily add new cores to the system with a minimum overhead. Besides, it is completely integrated with the SoPC Builder Tool, which would be used further to integrate the IP-Cores.

After choosing the processor and other proprietary IP-Cores available in the SoPC Builder library, it is time to implement the new user modules in the system. The implementation is performed by Quartus II design tool [6], from Altera, which is a rather natural choice considering that SoPC Builder is, in fact, a Quartus' plug-in.

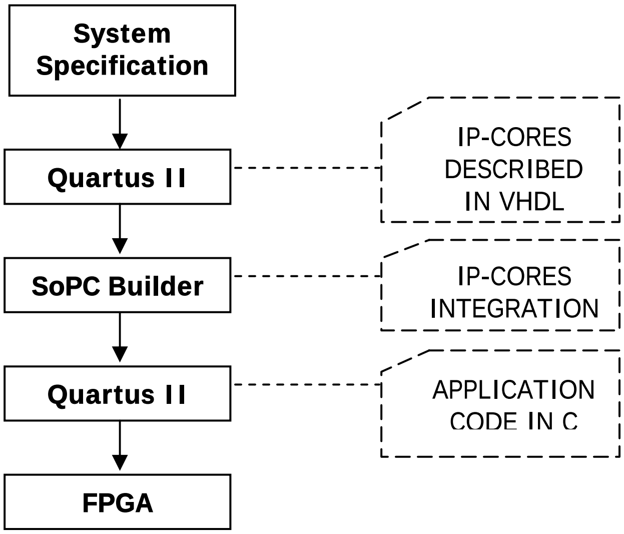
In order to use the IP-cores correctly in the SoPC Builder tool, their interfaces have to be developed based on the Avalon Bus specifications. Each core is synthesized and validated, before the integration step.

The system integration by using SoPC Builder tool is incredibly simple. The designer just needs to specify the devices that will form the SoC (among the built-in cores or user-created ones), configure a few parameters of each (such as the desired operating frequency), and ask the tool to generate the system. After synthesis, a completely integrated system is created as a unique module ready to be tested.

Finally, it is necessary to write the software partition, in C code, to be run in the Nios Processor. Even in this time, the SoPC Builder tool has a fundamental role in the process:

during system generation, it creates all the C headers and libraries containing the information needed by the user to access all the devices of the system, like memory mapped device's addresses and some functions to simplify their usage. In this way, the generation of the code is made much faster, as the designer does not need to worry about such details.

The design flow is shown below:



3. EXAMPLE

In order to test the adopted methodology, a prototype of a small robot was developed. It contains a pair of DC motors (controlled by an H-Bridge) to provide movement to the robot and an infrared receiver that is used to receive commands from a remote controller. The controller was implemented in an Altera Excalibur (Nios) Prototyping board [6], equipped with an infrared transmitter.

As the controls of the motor and of the infrared sensor are very specific applications, their cores had to be implemented. Also, a small program written in C was developed to be run on the Nios processor, in order to control data flow among the devices of the system. It simply reads the command sent by the transmitter, decodes it, and sends it to the H-Bridge, causing the robot to move.

The physical structure of the robot was constructed using Lego MindStorms [7]. This well-known toy consists of small plastic pieces, motors, sensors and engines that can be grouped in order to create robots. In our design, the Altera prototyping board replaced the standard programmable control unit that comes with the toy.

For this example, the productivity gain was notorious. The whole integration process could be completed in a few minutes. If the traditional approach were used, several hours would be needed to generate a completely error-free code. Also, without the need of writing the code for the processor, for the communication bus, and for the integration code, the whole system could be implemented in approximately 930 lines of code. This includes the VHDL code needed to implement the remaining IP-Cores and the transmitter (715 lines), besides the C code used to control the system (215 lines).

The physical implementation in the FPGA target technology (APEX200KE) resulted in approximately 2700 logic elements, with the C code occupying almost 27 Kbits of the chip onboard memory. The working frequency of the system was 33.333 MHz, provided by the clock source contained in the Excalibur board.

4. CONCLUSIONS

A design flow for developing of SoPCs has been presented. This works shows how easy and productive is the development of such systems by using modern design CAD tools. It also shows that integration problems can be almost completely avoided by automating the integration process. A successful example has been presented.

5. REFERENCES

[1] Jong, Gjalt de, "*A UML-Based Design Methodology for Real-Time and Embedded Systems*", SGDA, 2002.

[2] E. Wein, "*Core integration: overview and challenges*", IEEE/ACM International Conference on Computer-Aided Design, 1998, pp. 450-452, November 1998.

[3] Avalon Bus Specification Reference Manual. Avaliable: *Altera Inc. Site*. http://www.altera.com/literature/ mnl_avalon_bus.pdf. Visited on May, 2004.

[4] SoPC Builder User Guide. Avaliable: *Altera Corp. Site*. http://www.altera.com/literature/ug/ug_SoPCbuilder.pdf. Visited on May, 2004.

[5] Nios 3.0 CPU Datasheet. Available: *Altera Corporation Site*. http://www.altera.com/literature/ds/ds_nioscpu.pdf. Visited on May, 2004.

[6] Altera Development Kits and Design Software. Available: *Altera Corporation Site*. http://www.altera.com. Visited on May, 2004.

[7] Lego Mindstorms Kits. Available: *Lego.com Site*. http://mindstorms.lego.com/eng/default.asp?bhcp=1. Visited on May, 2004.