

Co-design techniques within digital system design curricula

Luís Gomes, Anikó Costa

¹ Dept. of Electronic Engineering, Faculty of Science and Technology, Universidade Nova de Lisboa & UNINOVA
Campus da FCT, Caparica, PORTUGAL; <lugo|akc>@uninova.pt

ABSTRACT

This paper presents teaching experiments on digital system design including hardware–software co-design techniques for embedded systems. Petri nets are used as the central formalism supporting different activities ranging from modeling and specification of the behavior of the system to implementation of Petri nets sub-models into software or hardware, and including usage of tools for validation through simulation, property verification and automatic code generation (C and VHDL). A typical project is used to illustrate the teaching flow method.

1. INTRODUCTION

The development of complex digital systems enforce in many situations to consider techniques for hardware–software co-design. Usage of design automation tools and reconfigurable platform to support prototyping are very common in engineering development flows. This paper focus on the experience on teaching digital system design using techniques for hardware–software co-design, based on FPGAs as implementation platform and having a strong emphasis on the modeling of the system behavior (control part of the system) through state diagrams and Petri nets [1].

Those subjects have being taught within the disciplines “Digital System Design” (mandatory, 80 students enrolled) and “Co-design and Reconfigurable Systems” (optional, 10 students enrolled, minimum), both offered in the second cycle studies (MSc degree on Electrical and Computer Engineering) at the Faculty of Sciences and Technology of Universidade Nova de Lisboa, Portugal (according with the Bologna process reformulation).

2. LEARNING OBJECTIVES

The lecturing part of the courses has a strong emphasis on the laboratory assignments and usage of Project based learning attitude (PbL). Several small and medium-sized projects are proposed to the students. A set of computational tools are available to the students allowing exercising the whole development flow [4], [6], from specification of the system model (using a Petri net graphical editor) till the deployment into the implementation platform (using a configuration tool), including partitioning of the system model into a set of parallel components (using a split tool), automatic code generators from Petri nets to VHDL and from Petri nets to C, and verification and simulation tools as well. Figure 1 presents an overview of the set of tools.

Complementing the experimentation at physical level, three kinds of simulation environments are considered: 1) at the model level, where the Petri net model is simulated using a “token-player attitude”; 2) at the implementation level, where a time simulator is used based on the VHDL code that will be deployed into the FPGA platform; 3) at the application level, where an animated interactive

graphical synoptic is used allowing the association with the system model execution.

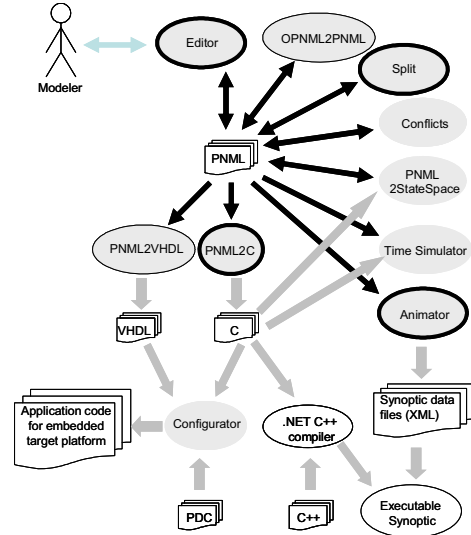


Figure 1 Tool overview

Experimentation both at physical level and simulation environments are realized, comparing different types of partitioning of the model into components, as well different solutions for each component. For instance, the students are encouraged to compare implementations of one specific component through different solutions, ranging from a microcontroller based implementation using C code to direct hardware implementation described in VHDL code using different clocks, allowing different performances and costs. The different possibilities of mapping to implementation platforms are presented in Figure 2.

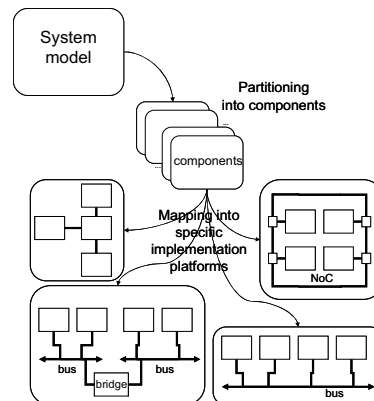


Figure 2 From models to code through model partitioning and mapping

For the physical implementation a didactic kit from Digilent, Inc. is used. This kit includes a FPGA Spartan3 XC3S200, containing several peripherals, namely PS/2, RS-

232 and VGA ports, several switches and push-buttons, and several LEDs and 7-segment displays, as well. All student working groups (typically composed by two or three students) hold one of these didactic kits during the whole semester, only returning the kit to the storage at the end of the semester.

3. A TYPICAL PROJECT EXAMPLE

One of the small projects proposed for the student is an example of an automation system integrated in a manufacturing system, allowing the control of a four cell first-in-first-out system with four conveyor belts (viz. Figure 3). At the entry and end positions of each conveyor is installed a presence detector sensor to detect objects arriving and exiting (except for the last one). These sensors will be modeled as the system's input signals. In order to obtain an executable model of the controller of this system is possible to choose between several possible attitudes. One of them is to choose Colored Petri nets as used in [3], other is to use a low level Petri net model, as presented in Figure 4 using IOPT (Input Output Place Transition) Petri net [5].

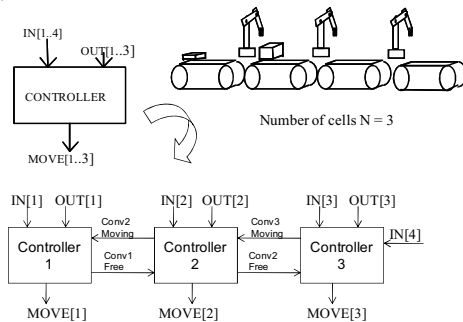


Figure 3 First-in-first-out system

Students have two lab classes of two hours to exercise this project. They edit the Petri net model using the editor, then invoking the PNML2VHDL code generator tool they obtain the VHDL code for the implementation of the global controller. In the other hand invoking the Split tool [2], they obtain the models allowing a distributed execution using autonomous controller for each conveyor as shown on the Figure 5 and implement that model using the VHDL code generator or/and using Picoblaze microcontroller IP. Students are strongly encouraged to compare those implementations and to propose any other solutions. As a result, they elaborate a report emphasizing the advantages and disadvantages of each proposed solution.

4. A BRIEF DISCUSSION AND CONCLUSIONS

Overall, students enjoy and are motivated to participate actively in the lab activities. The fact that they hold the development framework all the time facilitates that they come up with different solutions. The comparison between the different modeling attitude and different implementations helps to develop engineering skills and to illustrate through practice the formal equivalence between hardware and software, which is the basis for applying co-design techniques.

5. ACKNOWLEDGMENTS

The presentation of this work is supported by the Portuguese FCT sponsored POSC/EIA/61364/2004 FORDESIGN project (<http://www.uninova.pt/fordesign>).

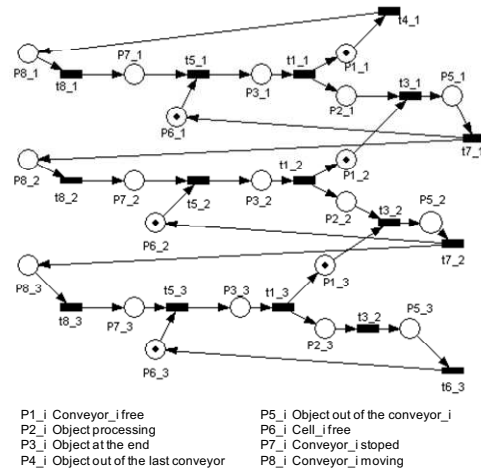


Figure 4 Petri net model of the system

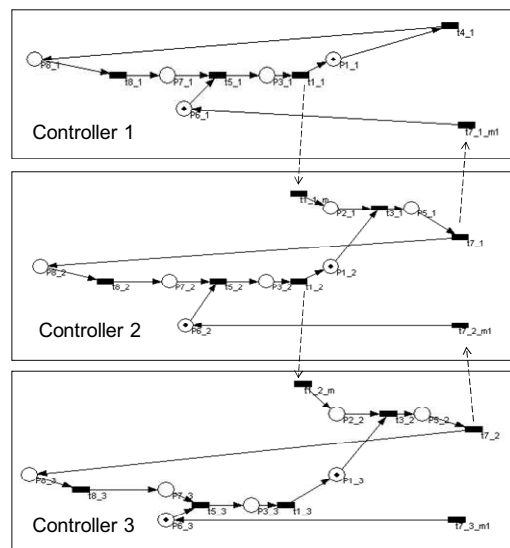


Figure 5 Three controller model

REFERENCES

- [1] Desel, J., Reisig, W.: Place/Transition Petri Nets. In Reisig, W., Rozenberg, G., eds.: Lectures on Petri Nets I: Basic Models. Volume 1491 of Lecture Notes in Computer Science. Springer (1998) 122{173 ISBN: 3-540-65306-6.
- [2] Costa, A., Gomes, L. Petri net Splitting Operation within Embedded Systems Co-design. In: INDIN'2007 - 5th IEEE International Conference on Industrial Informatics, 23-26 July 2007, Vienna - Austria.
- [3] Gomes, L., Barros, J. P., Costa, A.: Structuring mechanisms in Petri net models: From specification to FPGA based implementations. In Marian Adamski, Andrei Karatkevich, M.W.E., ed.: Design of embedded control systems, Springer (2005) pp. 153-166
- [4] Gomes, L., Barros, J. P., Costa, A. Petri Nets Tools and Embedded Systems Design. In: PNSE'07 - International Workshop on Petri Nets and Software Engineering, June 25-26, 2007, Siedlce - Poland.
- [5] Gomes, L., Barros, J. P., Costa, A., Nunes, R. The Input-Output Place-Transition Petri Net Class and Associated Tools. In: INDIN'2007 - 5th IEEE International Conference on Industrial Informatics, 23-26 July 2007, Vienna - Austria.
- [6] <http://www.uninova.pt/fordesign>, 2007 (FORDESIGN project home page).