A Platform FPGA-based Hardware-Software Undergraduate Laboratory

Joseph Schneider, Mikel Bezdek, Ziyu Zhang, Zhao Zhang, and Diane T. Rover
Department of Electrical & Computer Engineering, Iowa State University, Ames, IA
{josschne,mbezdek,zyzhang,zzhang,drover}@iastate.edu

Abstract

Almost all universities offer introductory courses that focus on microcontroller-based systems and embedded programming. Advanced course offerings vary, and are often not available until the graduate level, leaving a gap in training undergraduates. However, courses are emerging that take advantage of new embedded development platforms that support hardware-software codesign. At Iowa State University, the Department of Electrical and Computer Engineering is developing a new upper-level design course on embedded systems design (CPRE 488) that sits between the introductory course on microcontrollers (CPRE 211) and a graduate course on system-level design (CPRE 588). CPRE 488 pulls together pedagogy from leading textbooks in embedded systems (such as Wolf, and also Vahid and Givargis) and puts the concepts into an intensive laboratory incorporating platform FPGA technology. The lab utilizes Xilinx's Virtex II Pro FPGA, which includes a hard-core dedicated processor as well as FPGA fabric, allowing for a complete hardware-software system to be explored entirely within the FPGA.

1. Introduction

Undergraduates are often introduced to microcontroller software design in an introductory course taken during the second year of classes. The course typically deals with software and hardware issues, but leaves the two relatively separate. Because of the growing importance of system-level design methodologies, aspects of hardware-software codesign should be incorporated into the curriculum. Indeed, several books [1], [2] are available which are geared towards undergraduate system-level embedded design courses. Undergraduate courses (e.g. [3]) have even been taught using system-level design languages (SLDLs) such as SpecC[4] or SystemC[5] for design. However, system-level design is still in its infancy, with many competing technologies and methodologies. Thus, system-level design issues are often taught only in graduate-level classes. Iowa State recognized the need to bridge a gap between the graduate level codesign course and the undergraduate introductory course in a way that would teach system-level design techniques without spending time learning the methodologies and languages behind such techniques. Such a course is useful for students pursuing both industrial and academic careers.

Therefore, a new course has been created (CPRE 488), along with the corresponding lab described in the paper. The laboratory strives to give students practical problems in a real-world environment while preparing them for future study in codesign methodologies.

2. Lab overview

Iowa State has created a FPGA-based laboratory with approximately 15 stations each equipped with Xilinx Virtex II Pro boards, specifically the XC2VP7. This device has a built-in PowerPC 405 processor which can run at up to 300 MHz, and has 3,008 slices which allows for fairly-large designs. The development boards we are using allow additional peripherals to be attached through a generic 160-pin header, a capability which we use extensively throughout the lab.

The main design environment for the lab is the Xilinx EDK/ISE environment. In this environment, programs for the PowerPC block can be written in C or C++. Hardware in the EDK is represented as separate IP blocks. Students use this system-level view to connect IP blocks to create their designs. This allows the students to gain system-level design experience without spending the time to teach them the often complicated nuances of a system-level design language, which is left to the graduate-level course (CPRE 588).

The labs are designed to emphasize the issues involved in codesign. Because we do not spend time teaching the students SLDLs, we have more time to explore real-world examples of system-level design techniques. The students investigate issues such as performance tradeoffs, meeting tight timing or power constraints, and managing communication between system components. A second component of the labs is an RTOS, such as VxWorks or Xilinx's Xilkernel, which is used to explore multi-threaded hardware-accelerated embedded systems. Using an RTOS, students optimize the system performance through multithreaded implementation and system-level profiling.

These topics are explored in the realm of two main examples, a digital camera and an MP3 player. Both examples offer opportunities to gain experience with real world system-level design decisions. Additionally, multimedia applications are particularly suited to teaching embedded systems optimization because they often require tight timing constraints.

3. Laboratory units

The lab units are designed to build students' confidence gradually in dealing with codesign issues. First, students are introduced to the lab environment in the more familiar realm of software programming. Hardware aspects are introduced next, and the issues that arise from combining hardware and software to form a system are explored. Finally, designs utilizing an RTOS are implemented. The following is a summary of each lab unit.

Lab 1: Introduction to toolset and hardware

Students are introduced to the Xilinx ISE / EDK toolset and the boards which they will be using during the semester.

Labs 2-3: Hardware introduction

Learning how to interface with custom hardware, especially memory-mapped hardware, is imperative for students to understand hardware-software systems. This lab ensures that students have mastered this skill. The lab then asks them to apply that knowledge to build their own simple hardware device.

Lab 4-6: Digital camera

To explore the profiling, mapping, and optimization of a hardware-software system, students are given a software-only implementation of a JPEG camera which reads an uncompressed image from a camera, encodes it, and saves it to a Compact Flash (CF) card. Also given are the requirements for the desired hardware-software system. A variety of profiling methods are used to identify candidate functions to be moved to hardware. After identifying the appropriate function to move to hardware, creating custom hardware, and implementing the design, students time their design to see if it has met requirements. When finished, the camera will be fully functional, a feature that may motivate students more than a simple software simulation of a camera.

Lab 7-9: MP3 player

Again, students are given a software design of a MP3 player which is complete, but needs additional functionality. Students are asked to add functionality to display the song title and song updates with specific timing, a task which is difficult without an RTOS. These labs introduce the real-time operating system and ask students to port the design to the RTOS. Students are introduced to the benefits of developing in an RTOS environment by separately developing threads and combining the threads to form a system that meets the requirements.

4. Conclusion

The labs given here represent the introduction of hardware-software tradeoffs into the undergraduate curriculum at Iowa State University. While system-level design is gaining momentum, it is a rapidly changing target. By teaching the design methods to undergraduates without taking the time to teach specific SLDLs or refinement steps, students have time to focus on a more general view of system-level design that should prove useful immediately in industry and academic realms.

ACKNOWLEDGMENTS

This work was partially supported under NSF grant no. EEC-0088071, and through support from Rockwell Collins Foundation and Xilinx, Inc.
We gratefully acknowledge the contributions of Andrew Larson, Jason Boyd, Robert Walstrom, and Ramon Mercado to development of the laboratory.

REFERENCES

[1] J. Staunstrup and W. Wolf, editors, *Hardware/Software Co-Design: Principles and Practice*, Kluwer Academic Publishers, 1997.
[2] F. Vahid and T. Givargis, *Embedded System Design: A Unified Hardware/Software Introduction*, John Wiley & Sons, 2002.
[3] *Teaching trade-offs in system-level design methodologies*, K. Sakiyama, P. Schaumont, D. Hwang, I. Verbauwhede, Microelectronic Systems Education, 2003. pp. 62- 63.
[4] *SpecC website*, Available: http://www.specc.org
[5] *SystemC website*, Available: http://www.systemc.org