Engaging Sophomores in Embedded Design using Robotics

Amir Momeni
ECE Department
Northeastern University
Boston, MA, USA
momeni@ece.neu.edu

Fritz Previlon
ECE Department
Northeastern University
Boston, MA, USA
previlon@ece.neu.edu

Agamemnon Despopoulos
ECE Department
Northeastern University
Boston, MA, USA
despopoulos@ece.neu.edu

Gunar Schirner
ECE Department
Northeastern University
Boston, MA, USA
schirner@ece.neu.edu

John Kimani
ECE Department
Northeastern University
Boston, MA, USA
kimani@ece.neu.edu

David Kaeli
ECE Department
Northeastern University
Boston, MA, USA
kaeli@ece.neu.edu

ABSTRACT

The material covered in a typical Computer Engineering class tends to be heavily focused on a single subject area. Students often struggle to see how different subjects are interrelated or how they can be combined to address a wider range of problems. They generally have to wait until after they have completed subsequent related courses before they are able to put the previously material in context. For example, the concepts students learn in a typical digital logic course usually make more sense when one is taking computer architecture.

A second issue is that students fail to grasp the richness and diversity of Computer Engineering until they have taken courses in hardware, software, networking and computer architecture. This may push students away from the field prematurely.

To address these issues, we have designed a new class targeted at sophomores that covers a broad slice of Computer Engineering. The class teaches students many of the fundamental concepts of Computer Engineering. The course is required for students pursuing degrees in Computer Engineering, Electrical Engineering and Computer Science at Northeastern University. It provides them with a hands-on experience and presents the basics of the Unix/Linux operating system, high level programming concepts, introductory digital design, computer organization and wireless networking. Students get the opportunity to directly apply the theory presented in the classroom as they build a working remote-controlled robotic arm.

The class is taught in a colaboratory, an integrated laboratory-classroom environment. The room facilitates team-based design, active learning and exploration, while allowing the instructor to move seamlessly between lecturing and hands-on laboratory experiences. By the end of the course, the students acquire a general understanding of the different areas in Computer Engineering. They leave more prepared and more excited to tackle the more specialized courses that they will take later in the curriculum. They are also better prepared for their first cooperative education experience, which many of them will do immediately after taking this class.

WCAE '15 June 13 2015, Portland, OR, USA
 ISBN 978-1-4503-3717-5/15/06 $15.00
DOI: http://dx.doi.org/10.1145/2795122.2795131.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer science education

1. INTRODUCTION

Traditionally, the undergraduate curriculum for electrical and computer engineers is made up of specialized courses, each focusing on a specific aspect of the field. The students generally take individual courses on programming, algorithms, digital logic design, circuits, computer architecture, networks, etc. This approach is very fitting when a student has a general understanding of how all these layers fit into the computing stack. However, for a freshman or a sophomore, it can be a challenge to place these different specialized courses in perspective.

At Northeastern University, more than 95% of our engineering students pursue cooperative educational options - paid work experiences in their field of study. Students begin their first *coop* midway through their sophomore year. It is a challenge to the faculty to prepare students with a marketable set of skills that help them land their first coop job.

Given this motivation, we have developed a course titled *Embedded Design: Enabling Robotics*. The concept for the theme of the course was motivated by a senior capstone design called EyeTracker. The system allowed a quadraplegic to feed themselves using their eyes to control a robot-equipped feeding instrument. This motivating example capstone project was featured on MSNBC. What we found most exciting was how engaged students became when working on a compelling application. The Embedded Design class utilizes a Wiimote

to control the robot. We hope to extend this interace to allow control to utilize camera input as well. The key observation was that students become highly engaged when working with robotics, so incorporating this into a hands-on class that teaches Computer Engineering fundamentals seemed like a good idea.

Embedded systems utilizing systems on chip (SOCs) often require a fair amount of knowledge in both software and hardware concepts. Engineers need to understand how they should utilize hardware and software to come up with appropriate solutions. This offers an opportunity to introduce these concepts to second year students in Electrical and Computer Engineering, as well as in Computer Science. After taking this course, students will have a clearer view of the different layers of the computing stack. This understanding can also guide them later in their choice of electives for the completion of their course requirements.

The Embedded Design course aims to teach students many of the basic fundamentals of Computer Engineering. The course revolves around a development board (ZedBoard), a Bluetooth-enabled remote control (Wiimote) and a robotic arm. The ZedBoard is equipped with a dual-core ARM processor (capable of running the Linux operating system), and Xilinx series-7 Programmable Logic cells. Laboratory experiments are coordinated with the lectures and provide students the ability to control the robotic arm with the Wiimote, using a combination of digital logic design, networking and C programming. This course allows the students to directly apply the fundamental subjects of an Electrical and Computer Engineering education in a fun and interactive environment.

This new class rolled out in Fall 2014, and student reviews have been highly positive. One recurring piece of feedback is that the class affords the students the opportunity to not only learn some of the fundamental concepts in Computer Engineering, but even more importantly, how these concepts can be applied together to build solutions to impact in their everyday lives.

2. RELATED WORK

Embedded systems laboratories are an important part of the curriculum in ECE departments in most universities around the world. Several universities have proposed various embedded systems laboratories. In previous efforts at Northeastern [6], we developed an undergraduate embedded systems lab based on the Blackfin processor to address a number of applications, including controller design, RS-232 communication, encryption, and image processing. California Polytechnic State University [8] has also proposed an Embedded system course targeted at the Blackfin processor. The University of Alabama [11], and Marquette University [7] also proposed embedded systems laboratories in their curriculum. Their laboratories are based on VMEbus architecture and LinkSys WRT54G routers, respectively.

Several other universities have used FPGAs in their embedded systems courses. A series of three courses are designed at the Rochester Institute of Technology [9]. They used FPGAs as the target platform to cover advanced topics such as computer organization, custom intellectual property (IP) components and capstone design. Several courses are also offered at Iowa State University [12] to provide hands-on experience on embedded systems design using FPGAs and in particular, the ZedBoard. However, all of these courses target undergraduate students at the junior or senior level. Some introductory courses have also been designed for freshmen such as *Introduction to EECS* at MIT [1] and *Introduction to ECE* at CMU [2]. The primary goal of these courses is to make the first year students familiar with the basic concepts in Electrical and Computer Engineering. The laboratory described in this paper, is designed for freshmen and sophomores to provide them with a general understanding of the variety of areas in Computer Engineering. The laboratory delves into more detail than the introductory courses, while still covering a wide range of areas in Computer Engineering.

3. COURSE DESCRIPTION

This course covers a wide range of areas in Computer Engineering. Our goal is to focus on breadth, as the students will get the opportunity to learn materials in much more depth later, in more specialized classes. The objectives of the Embedded Design course are:

- To introduce ECE students to many of the fundamental concepts in Computer Engineering.
- To become familiar with Unix/Linux and embedded programming.
- To introduce students to digital design principles.
- To acquire knowledge of embedded system design.
- To be exposed to wireless networking and robotic control.
- To develop an appreciation for the software/hardware interface.

The course is taught in an environment that favors the immediate application of the topics covered in class. The laboratory experiments were specifically designed to be integrated with the course agenda. By completion of all the laboratory experiments, students will have built a working remote-controlled robotic arm.

3.1 Lectures

A variety of topics are taught during the lectures. The lectures are based on the textbook *Introduction to Computing Systems: From Bits and Gates to C and Beyond* by Yale Patt and Sanjay Patel [10]. Because the classroom environment favors the application of the theory learned in class, students have the opportunity to immediately test and apply the concepts taught during the lecture. These topics include:

- The Linux operating system, C programming in the Linux environment
- Data structures and algorithms
- Digital logic design, including boolean algebra, simple and compound gates
- Object-oriented programming
- Hardware/software interface
- Memory systems
- CPU architectures

Table 1: Interfaces on the ZedBoard used in the colaboratory.

USB-JTAG
10/100/1G Ethernet
USB OTG 2.0
SD Card
Five Digilent Pmod compatible headers (2×6)
Seven Push Buttons
Eight dip/slide Switches
Nine user LEDs

3.2 Laboratory

The strongest feature of the class is its laboratory component. It includes a 2 hour weekly lab and 2 1-hour lectures. Many times, the lectures become laboratory sessions, allowing students to work with their embedded boards, or practicing some programming concepts. Students work in pairs and receive hands-on experience on an actual embedded platform, (the ZedBoard) that is based on the Xilinx Zynq system on a chip (SOC). The platform runs xillinux [3], a flavor of the popular Ubuntu Linux distribution. Students learn how to develop C programs on Linux for the ZedBoard, create digital designs that are embedded to run on the FPGA of the Zynq SOC, interface to a wireless Wii remote, read from memory-mapped switches and push buttons and write to LEDs on the ZedBoard, and control a robotic arm with the ZedBoard. The lab exercises are designed to follow the classroom topics and provide open-ended design experience. The final lab involves controlling the robotic arm with both hardware and software, bringing together many separate components developed through the labs. A more thorough description on the laboratory experiments can be found in Section 5.

4. COLABORATORY EQUIPMENT

The hands-on elements of the class are based on the Xilinx ZedBoard. The students learn how to use the ZedBoard to control a robotic arm. They utlize both programming running on an ARM CPU, and then a digital design synthesized to an FPGA, to control the robotic arm. The input commands come from a Wiimote which interfaces to the ZedBoard through a Bluetooth connection.

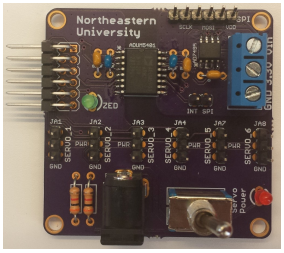
4.1 ZedBoard

We use the ZedBoard as the main platform in this class. The ZedBoard (see Fig. 1a) is an evaluation and development board designed by Xilinx. It is based on the Xilinx Zynq-7000 Extensible Processing Platform which combines a dual core Cortex-A9 Arm processor with 85000 Series-7 Programmable Logic cells [5].

Table 1 lists the specific subset of interfaces on the ZedBoard that are used in this class. The ZedBoard runs an embedded Linux distribution which is installed on an SD Card. The board is configured such that the ZedBoard automatically boots from the SD Card and starts Linux when the students power the ZedBoard. The ZedBoard is connected to a PC through an ethernet interface. The students log into the ZedBoard from the PCs via SSH. A USB On-The-Go (OTG) adapter is also used to connect a Bluetooth dongle to the ZedBoard. This Bluetooth dongle is used to establish a connection between the Wiimote and the ZedBoard. The robotic arm is connected to the ZedBoard through the Pmod connectors. We designed a custom interface board (see Fig. 1c) to ease the connection between the robotic arm and the ZedBoard.

(a) The Xilinx ZedBoard[5]

(b) The robotic arm [4].



(c) The custom-designed interface board.

(d) The Wiimote.

Figure 1: Colaboratory equipment.

The students use both the ARM CPU and the FPGA on the ZedBoard to run their C programs and their digital logic designs, respectively. They use the Xilinx Software Development Kit (SDK) to write and debug their C programs remotely on the ZedBoard. They also use Simulink as the software platform to develop digital logic, and Simulink HDL Coder to synthesize the Simulink models and download them to the FPGA using the JTAG interface. Simulink HDL Coder generates Verilog or VHDL code from the Simulink models or from Matlab functions. It provides a workflow advisor (see Fig. 2) to automate the process of FPGA programming. The Simulink HDL Coder makes the task of

programming the FPGA easier for the freshman/sophomore students, who usually do not have any prior experience in hardware description languages.

4.2 Robotic arm

We use the SG5-UT robotic arm [4] in this class (Fig. 1b). The robotic arm comes with 5 RC servos, and therefore provides 5 degrees of freedom. The servos are called Base, Bicep, Elbow, Wrist, and Gripper. Each servo can operate through 180 degree when given the appropriate Pulse Width Modulation (PWM) signal, ranging from 600 μs to 2400 μs with the period of 20 ms. The students learn the basics of motor control with PWM signals, generating these signals in both software and hardware. First they write a C program to generate the appropriate signal using a sleep function to determine the duty cycle. Next, they use Simulink to design a digital circuit to generate the same PWM signal in hardware.

4.3 Wiimote

We use the Wiimote (Fig. 1d) as a Human Interface Device to communicate with the ZedBoard. The students learn how to establish a Bluetooth connection in Linux to connect to the Wiimote. They write C programs to receive and decode the data packets from the Wiimote. They use both the Accelerometer values and the buttons on the Wiimote to control the LEDs on the ZedBoard and also the robotic arm.

5. HANDS-ON ASSIGNMENTS

This class has 11 hands-on assignments that are carried out during class time. Each assignment provides students with hands-on experience on an embedded platform. The exercises are designed to follow the theory elements being covered, and provides open-ended design experience. Each assignment moves the design one step closer to the final project (see Fig. 3), where students control the robotic arm with the Wiimote using the ARM processor and the FPGA on the ZedBoard. The exercises are divided into 5 categories (Table 2). Each exercise is explained next.

5.1 Introduction to the ZedBoard

The first four assignments are designed to introduce the students to the ZedBoard and Linux. The first assignment shows the students how to use secure shell to login to the ZedBoard's ARM CPU, to practice some Linux utilities, and to write, compile, and run a couple of C programs on the ZedBoard. In the second assignment, the students start writing more sophisticated C programs to explore data types in C. The third assignment is focused on debugging skillsets, using gdb in Linux on the ZedBoard. The fourth assignment is an introduction to addressing and memory mapping. The students learn how to use general purpose memory-mapped I/O on the ZedBoard. They learn how to read/write values from/to memory-mapped device addresses and see how to interface with LEDs, switches, and push buttons on the ZedBoard.

In the final project, the students design a digital circuit to generate PWM signals to control the RC servos. They also need to control the speed of changing the duty cycle of the PWM signals. They learn how to use counters as the main component of their digital design for PWM generation. This experience complements the introduction of sequential logic that is presented in lecture. They get to experiment with the counters and speed control mechanisms. The last part of the fourth assignment is to design a counter in software. They read the value of switches on the ZedBoard to specify the initial value of the counter. They also use the LEDs on the ZedBoard to display the value of the counter. They are also learning about two's complement, and learn how negative numbers are represented. The push buttons are also used to change the direction and the speed of counting. In the later assignments, they will design the same counter in hardware.

5.2 Networking

A single assignment is devoted to networking, providing an introduction to Bluetooth and our Human Interface Device (HID) (the Wiimote). The students learn how to establish a Bluetooth connection between the HID and the ZedBoard. They also learn how to use software to carry out communications between the HID and the platform, and how to read data from the Wiimote and communicate with it on the ZedBoard. They write a C program that displays the X-axis acceleration on the LEDs.

5.3 Digital Design

In the next three assignments the students learn how to develop digital designs. In the first assignment of this section, they use Simulink to develop digital designs, including decimal to binary converters, binary to decimal converters, full-adders, 8-bit adders and multipliers. They also become familiar with comparators, which will be used in the final PWM generator design.

In the second digital design assignment, they learn how to use the Mathworks Simulink HDL Coder to program the FPGA on the ZedBoard, and run their digital designs on the actual platform. They also become familiar with the push buttons bouncing issue, and design a digital logic for de-bouncing.

In the third digital design assignment the students are ready to design an 8-bit counter in hardware. Similar to the earlier assignment, they use the LEDs on the ZedBoard to display the value of the counter. The switches feed in the initial value, and the push buttons control the speed of counting as well as the direction.

5.4 PWM signals to Control the RC servos

Two assignments are devoted to show the students how to control the robotic arm using Pulse Width Modulation (PWM) signals. In the first assignement, they use the ARM microprocessor on the ZedBoard, while in the second they use the FPGA. In the first assignment, they learn the basics of Pulse Width Modulation (PWM), as well as GPIO interfacing in Linux. They write a C program to send PWM signals to the robotic arm using a Pmod connector on the ZedBoard. In the second assignment the students design digital logic to control the RC servos of the robotic arm. The digital logic will be implemented in the FPGA on the ZedBoard. The FPGA generates PWM signals, which control the RC servos in the robotic arm. The students use the switches to select the RC servos, and push buttons to move them to the right or left.

5.5 Hardware/Software co-design

The final assignment provided ties together all of the elements the students have developed in previous labs, and combines them in a hardware/software co-designed imple-

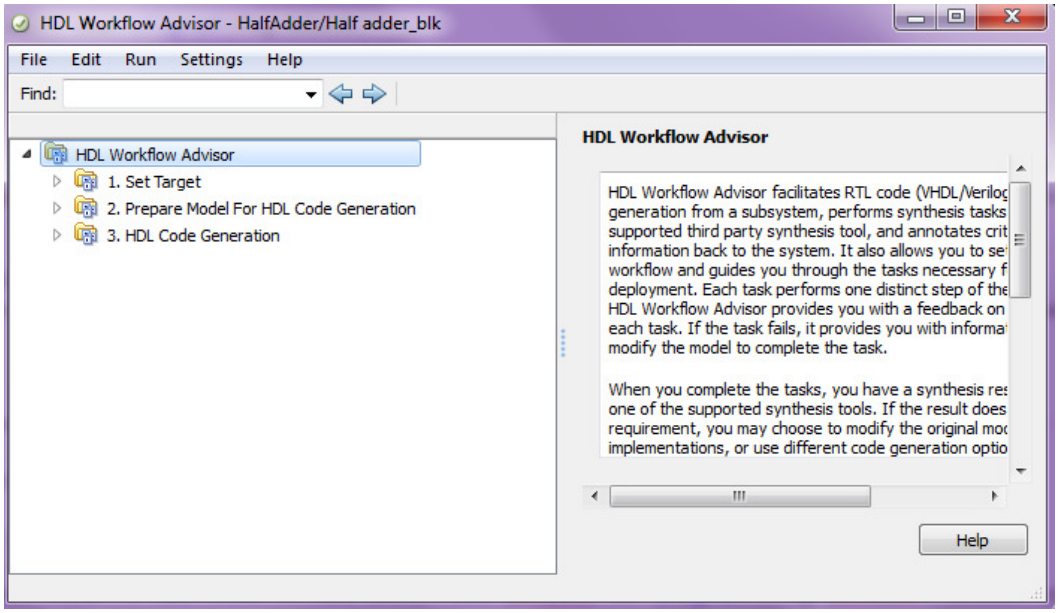


Figure 2: The Simulink HDL Workflow Advisor.

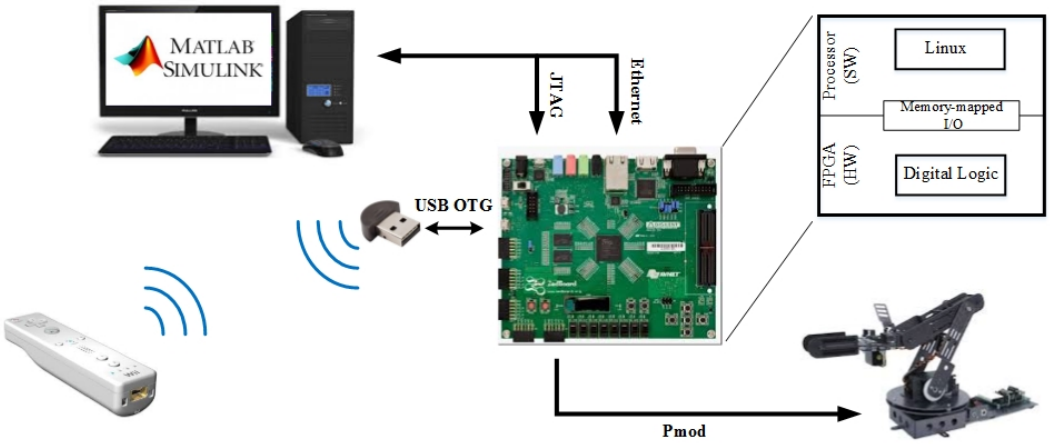


Figure 3: The final project in the Embedded Design class.

Table 2: Assignments description:

Section	Description	# of labs
Introduction	Connect to the ZedBoard, Linux commands, Write simple C programs	4
Networking	Introducing the Wiimote as a HID, establish a Bluetooth connection in Linux	1
Digital Design	Using Simulink to develop digital designs, and program the ZedBoard	3
PWM signals to control RC servos	Using software and hardware to generate PWM signal and control RC servos	2
Hardware/Software co-design	Final project	1

mentation. In their designs, they must control the robotic arm using the FPGA, and use the ARM processor on the ZedBoard to receive commands from the Wiimote and control the FPGA. They write a C program that receives signals from the Wiimote connected via Bluetooth and sends commands to the FPGA to carry out a specific movement by the robotic arm. The FPGA is used to generate the necessary PWM signals to control the robotic arm to carry out a specific action involving multiple servo motors simultaneously.

6. FEEDBACK

This course was extremely well received among the students. While the majority of the students admit that it was indeed a very challenging course, they all agreed that this course was very beneficial and helped them build a strong foundation in their Computer Engineering principles. Pop-

ular aspects of the class range from the use of the Linux environment to the use of a real robotic arm in practice. Moreover, we found that student most frequently noted the tight integration between the lectures and the labs as their favorite element of the class. Students get to immediately apply the theory they learned in lecture.

One frequent feedback on this class received during and after the class was that the material and experience prepared students for interviews during their first co-op job search, given they had the opportunity to learn a fairly wide range of Computer Engineering concepts.

7. CONCLUSION

In this paper we have presented our motivation, design and delivery of Embedded Design: Enabling Robotics, that was recently added to the undergraduate ECE/CS curriculum at Northeastern University. This course is designed to be taken by ECE students during the first semester of their second year. After a first semester of success with this course with the ECE students, it is now offered as a required class in Computer Science, providing our students with hands-on embedded systems/software knowledge and digital design experience.

The students completing this class are prepared and trained in many fundamental concepts in Computer Engineering. The laboratory assignments are well integrated and intermixed with the theory during the semester. The open-ended laboratory components challenge the students to think in terms of both hardware and software solutions in their embedded design projects. Based on student feedback, we feel that this course has provided a solid foundation for an education in Computer Engineering fundamentals.

Acknowledgment

The authors would like to thank the David House Foundation, Analog Devices, Xilinx and The Mathworks for their support of this project.

8. REFERENCES

[1] `http://sicp-s3.mit.edu/tutor/6.01`.

[2] `https://www.ece.cmu.edu/courses/items/18100.html`.

[3] `http://xillybus.com/xillinux`.

[4] `http://www.crustcrawler.com/products/arm5.php?prod=0`.

[5] Zedboard hardware user's guide. `http://zedboard.org/sites/default/files/ZedBoard_HW_UG_v1_1.pdf`.

[6] M. G. Benjamin, D. R. Kaeli, and R. Platcow. Experiences with the blackfin architecture in an embedded systems lab. In E. F. Gehringer, editor, *WCAE*, page 2. ACM, 2006.

[7] D. Brylow. An experimental laboratory environment for teaching embedded hardware systems. In *Proceedings of the 2007 Workshop on Computer Architecture Education, WCAE 2007, San Diego, California, USA, Saturday, June 9, 2007*, pages 44–51, 2007.

[8] D. Franklin and J. Seng. Experiences with the blackfin architecture for embedded systems education. In *Proceedings of the 2005 Workshop on Computer Architecture Education: Held in Conjunction with the 32Nd International Symposium on Computer Architecture*, WCAE '05, New York, NY, USA, 2005. ACM.

[9] A. F. MondragÃşn-Torres and J. W. Christman. A comprehensive embedded systems design course and laboratory. In *MSE*, pages 56–59. IEEE, 2013.

[10] Y. N. Patt and S. J. Patel. *Introduction to computing systems - from bits and gates to C and beyond (2. ed.)*. McGraw-Hill, 2004.

[11] K. G. Ricks, W. A. Stapleton, and D. J. Jackson. An embedded systems course and course sequence. In *Proceedings of the 2005 Workshop on Computer Architecture Education: Held in Conjunction with the 32Nd International Symposium on Computer Architecture*, WCAE '05, New York, NY, USA, 2005. ACM.

[12] D. Roggow, P. Uhing, P. Jones, and J. Zambreno. A project-based embedded systems design course using a reconfigurable soc platform. In *Proceedings of the International Conference on Microelectronic Systems Education (MSE)*, May 2015.