On Teaching Embedded Systems Design to Electrical Engineering Students

Stephan Wong, Sorin Cotofana
Computer Engineering Laboratory,
Electrical Engineering Department,
Faculty of Information Technology and Systems,
Delft University of Technology,
The Netherlands
{Stephan, Sorin}@CE.ET.TUDelft.NL

Contacting author:

Stephan Wong
Computer Engineering Laboratory
Faculty of Technology and Systems
Mekelweg 4
2628 CD Delft
The Netherlands

Email: stephan@ce.et.tudelft.nl
Phone: +31 15 278 6168
Fax : +31 15 278 4898

Abstract

This paper describes the some relevant issues encountered by the authors during the development and teaching of an embedded system (ES) course meant for third year Electrical Engineering Bachelor students at the Faculty of Information Technology and Systems, Delft University of Technology, The Netherlands. Given that we were addressing Electrical Engineering students, we deviated from the "traditional" way to teach ES design, i.e., focus on embedded software only. Consequently, we decided to cover theoretical as well as practical ES design aspects both from hardware and software prospective. We organized the course as a combination of lectures and a lab assignment. The main topics discussed in the lectures include embedded systems characteristics and requirements, unified modeling language (UML), instruction set architectures, compilation techniques, power-aware design, hardware accelerators, embedded system networks, and design methodologies. The current lab assignment encompasses the design of a spectrum analyzer for audio analogue signals that displays the frequency spectrum on an LCD display. The lab is meant to give the students an opportunity to solve a real life ES design problem by exercising hardware/software co-design in simulated company conditions.

On Teaching Embedded Systems Design to Electrical Engineering Students

Stephan Wong, Sorin Cotofana
Computer Engineering Laboratory,
Electrical Engineering Department,
Delft University of Technology,
The Netherlands
{Stephan, Sorin}@CE.ET.TUDelft.NL

Abstract

This paper describes the some relevant issues encountered by the authors during the development and teaching of an embedded system (ES) course meant for third year Electrical Engineering Bachelor students at the Faculty of Information Technology and Systems, Delft University of Technology, The Netherlands. Given that we were addressing Electrical Engineering students, we deviated from the "traditional" way to teach ES design, i.e., focus on embedded software only. Consequently, we decided to cover theoretical as well as practical ES design aspects both from hardware and software prospective. We organized the course as a combination of lectures and a lab assignment. The main topics discussed in the lectures include embedded systems characteristics and requirements, unified modeling language (UML), instruction set architectures, compilation techniques, power-aware design, hardware accelerators, embedded system networks, and design methodologies. The current lab assignment encompasses the design of a spectrum analyzer for audio analogue signals that displays the frequency spectrum on an LCD display. The lab is meant to give the students an opportunity to solve a real life ES design problem by exercising hardware/software co-design in simulated company conditions.

1 Introduction

In recent years, embedded systems are becoming increasingly more important due to their widespread utilization in every aspect of our lives. Embedded systems are affecting our lives in many ways while at the same time we are usually not aware of it. Examples can be found in alarm clocks, automobiles, mobile phones, personal digital assistants, etc.. An important development that sparked this widespread utilization must have been the advent of the microprocessor in the embedded systems design. In addition, this development is being fueled by the continuous advances of semiconductor technologies allowing smaller and faster chips to be manufactured. As a result, embedded systems are becoming smaller and are able to perform more functions at the same time. The utilization of microprocessors, called embedded processors in this setting, has greatly contributed to the digital data processing performance of many embedded systems. However, the embedded system design also entails many analogue components that must not be overlooked, e.g., analogue-to-digital converters.

Given the importance of embedded systems, we have developed a new course for third year Bachelor students studying Electrical Engineering at the Faculty of Information Technology and Systems, Delft University of Technology, The Netherlands. The course is entitled "Embedded Systems" and it is intended to introduce students to various aspects of embedded systems in the lectures and to familiarize students with the hardware/software co-design methodology in a lab assignment. The course is built such that the students, after successfully finishing the course, must be able to design an embedded system. This means that given a certain application the student must be able to:

- select an embedded system platform, consisting of hardware and software components, that is able to fulfill the requirements of the targeted application;
- realize a software-only implementation, analyze it, and identify its performance bottlenecks;
- overcome the identified performance bottlenecks by moving the bottleneck functions in question towards a hardware implementation.

Since this course is intended for Electrical Engineering students, the focus of the course in not only on embedded software. Instead, we focus on theoretical and practical embedded system design aspects from a hardware and software point of view. First, we introduce an embedded system specification and the various hardware and software components. Second, we introduce the hardware/software co-design methodology that encompasses the simultaneous design of hardware and software components. Third, we discuss the implementation in either hardware or software and their related code generation or synthesis, respectively. Finally, we highlight operating systems for embedded systems and network of embedded systems.

The practical aspect of the course is embodied in a lab assignment in which students have to design a frequency spectrum analyzer. Due to practical restrictions, the students are presented with a fixed hardware platform on which they must implement the system. In order to simulate a real industrial design setting, the students are handed no more than the assignment, the manuals of the platform, and the appropriate development tools. Complementing the simulated industrial setting, two teaching assistants were appointed to act as chief designers. The lab assignment incorporates various topics discussed in the lectures and involves the design stages from specification to implementation to verification.

This paper is organized as follows. In Section 2, we discuss the organization of the course into lectures, the lab assignment, and the grading system. In Section 3, we discuss the topics being discussed during the lectures. In Section 4, we describe the lab assignment and the lab setting and explain how the lab is linked to the lecture topics. Finally, in Section 5 we summarize our experiences in teaching the course, the feedback we received from students, and some future directions we foresee in this course.

2 Course organization

The embedded systems course is intended for third year Electrical Engineering Bachelor students who have background knowledge on "Digital Systems" (a first year course) and "Computer Architecture and Systems" (a second year course). From the student's perspective, the course is divided into three parts, namely:

- **Lectures** (28 hours total) The theory of this course is presented over fourteen lectures with each one taking two hours. The lectures are not mandatory.
- **Lab Assignment** (28 hours total) The practical part of this course entails an extensive lab assignment. The students must finish the lab assignment within seven labs (four hours each). During this time, the students have the opportunity to work on the presented hardware platform, implement their software and hardware solutions, and they may receive (minimal) guidance from two teaching assistants. Furthermore, it is obligatory for the students to attend the labs.
- **Self study** (64 hours total) Apart from the non-obligatory lectures and the obligatory labs, the student is expected to read the book and make preparations before attending the lectures, labs, intermediate tests, and the final exam.

The hours for self study are derived from the fact that the course is a three credit points course[1] (120 hours). The communication between the teaching team and the students happens via a well-known system called Blackboard [1]. Via Blackboard, we post announcements, lectures slides, intermediate test results and answers, and final exam results and answers. Next to the organizational part of the course, the students partaking in this course need to be graded. For this purpose, we have come up with the the following grading system:

- Three intermediate tests (T1, T2, T3) of which the second test (T2) is obligatory.
- Final exam (FE) which is obligatory and covers all the theory discussed in the course.
- All tests and final exam are *open book* meaning that students can bring any material that they deem necessary to bring the tests or exam to a good end.
- The grades for all the tests and the final exam range from 0 to 10.
- Students with the best design in the lab are also rewarded with bonus point (P) of 1.

The grade is based on the obligatory test T2 and the final exam as follows:

$$Grade = FE \times 0.8 + T2 \times 0.2$$

We felt that students who study regularly should be rewarded. Consequently, we have taken the non-obligatory tests T1 and T2 and converted them into bonus points which are calculated as follows:

[1]In the Delft University of Technology, an academic year consists of courses that add up to 42 credit points. Each credit point means that the student has to spend (on average) 40 hours on the course in question.

$$Bonus = \sum_{i=1,3}(Ti \times 0.1) + P \qquad\qquad only\ when\ T1, 3 \geq 6$$

This means that the intermediate tests T1 and T3 only count for one tenth of the final grade and only when a six or higher is achieved for the test. Taking into account the grade and the bonus points, the final grade can exceed the highest mark (10). Therefore, whenever the final grade exceeds 10, it is rounded to 10 as follows:

$$Final\ \ Grade = min(Grade + Bonus, 10)$$

We have to add that the final grade only counts after the students have successfully finished the lab assignment.

3 Lecture topics

As mentioned earlier, the course "Embedded Systems" is intended for third year Electrical Engineering Bachelor students. Therefore, the focus of the course is not only on the software development for embedded systems, but also on many hardware aspects of embedded systems. This section discusses the theory being taught in the fourteen lectures. The theory is based on the book "Computers as Components: Principles of Embedded Computing System Design" by Wayne Wolf [3]. The course starts with the presentation of our one sentence definition of what embedded systems are: "*Embedded systems are (inexpensive) mass-produced elements of a larger system providing a dedicated, possibly time-constrained, service to that system*". We are aware that no generally accepted definition of embedded systems exists. In most literature, only the key characteristic of embedded systems stating that they provide a dedicated service[2] to a larger (embedding) system is used to define embedded systems. The remaining topics in the lectures are described in the following:

- We discuss the requirements, challenges, and design methodologies of embedded systems followed by an introduction to the unified modeling language (UML) used to describe embedded systems.
- We introduce two types of embedded processor instruction set architectures, namely the ARM architecture and the SHARC architecture. In addition, we discuss subjects like interrupts, traps, exceptions, and a memory system incorporating caches.
- We discuss a typical CPU bus and their related subjects like DMA controller, I/O devices, A/D and D/A converters, and memory devices. Furthermore, we discuss the design, development and verification processes for both hardware and software components.
- We highlight software development possibilities and basic compilation techniques followed by an analysis and optimization of embedded software programs in order to decrease execution times, to save energy and power, and to reduce program size.

[2]The nature of the service is not relevant in this context.

- We present processes and operating systems, hardware accelerators, embedded system networks, and system design techniques.

The topics of the lectures are tightly coupled to the mentioned course book. Therefore, we have also utilized the slides that accompany this book as a basis for the final slides for this course. Finally, extensive examples were used (outside the mentioned book) to explain the mentioned topics.

4 Lab assignment

In this section, we first describe the lab assignment and the lab setting. Consecutively, we briefly highlight the relation that the lab assignment has with the theory that is being taught in the lectures (see previous section).

The lab assignment entails the design of a frequency spectrum analyzer that converts an incoming analogue audio signal into its corresponding frequency spectrum. We have to stress that only a (detailed) behavioral description (in words) is given to the students. The utilized description of the intended frequency spectrum analyzer's functionality is the following:

- Read N samples of the analogue audio signal and convert them into the digital domain;
- Compute the Fast Fourier Transform (FFT) over the N samples;
- Group the frequency components into frequency bands and compute, for each and every band, the average amplitude of all the frequency components within the frequency band;
- Display the average amplitude bands.

The lab assignment is performed by students in groups of two and it is intended to allow students to apply the theoretical knowledge gained during the lectures in an industrial setting. It is our intention to give to the students as much freedom as possible in the design of the spectrum analyzer. However, due to practical reasons, they were not allowed to choose their own hardware platform. The hardware platform presented to the students consists of an 80C552 microcontroller board, an alphanumerical LCD display, and a UP1 board containing an Altera FLEX10K20 field-programmable array (FPGA) [2].

First, assuming this hardware platform students are required to program the application software and compile it to the 80C552 microcontroller followed by an analysis of the performance. Then, the students must profile the code in order to determine the performance bottlenecks. Such bottlenecks do indeed exist since we have purposely chosen a slow microcontroller. Subsequently, the students must design hardware accelerators in order to increase the performance. This is performed by writing VHDL code and synthesizing it to the UP1 Altera FPGA board. In this way, next to software and hardware design, students must also take care of hardware/software interfacing and both hardware and software testing of their designs. Afterwards, the performance of the new design including hardware accelerators must be re-evaluated in order to determine whether more accelerators are needed in order to obtain closer to real-time performance. Furthermore, given that we want to simulate as close as possible an industrial setting of the lab, we only provided manuals of the microcontroller board, the FPGA board, and the development software tools. To complete the industrial setting, two teaching assistants were appointed to act as chief engineers and to only provide limited help.

The lab assignment has many links to the theory taught in the lectures. The main points are summarized in the following:

- **Program compiling, assembling, and linking** All these three steps in generating an executable from a high-level programming must be understood by the students. In relation, the utilized microcontroller is slow and has very limited instruction memory space. Thus, students are forced to apply optimizations in order to increase performance and reduce code size.

- **Timers and interrupts** For the analogue-to-digital (A/D) converter, the students are required to utilize a timer to start the A/D converter and display the converted digital values. In order to do this, students have to understand how interrupts work and they have to program the interrupt handlers.

- **Hardware/Software co-design** In the current lab setting, the students will certainly find out that only utilizing the microcontroller is not fast enough to perform both the FFT and the band-processing functions. Therefore, they have to identify the most time-consuming and compute-intensive functions and implement them as specialized hardware units on the FPGA board.

- **Device interfacing** As mentioned in the previous, it is required that certain functions are to be implemented in the FPGA board and thereby requiring it to communicate with the microcontroller and the program running on the microcontroller. The students must come up with a communication protocol to handle this communication.

In the next section, we will conclude this paper by briefly highlighting our experiences with both the lectures and the lab and we will also mention some feedback from the students. Furthermore, some ideas on future plans for the described embedded systems course are presented.

5 Conclusions

In this paper, we have shortly introduced the importance of embedded systems and thereby argued for the need for an embedded systems course. The “Embedded Systems” course described in this paper is intended for third year Bachelor students studying Electrical Engineering at the Faculty of Information Technology and System, Delft University of Technology, The Netherlands. The course is divided into two parts, namely a theoretical part which is taught in fourteen lectures and a practical part which consists of lab assignment. In teaching the course, we noticed that it is helpful to discuss more extensive examples in order to explain certain concepts. For example, when discussing signals on a CPU bus the utilization of a timing diagram is more insightful than only mentioning the signals. Furthermore, during the lab we noticed that only some students apply code optimizations discussed during the lectures in order to increase performance.

From the feedback we gathered from students, we found out that the lectures were useful due to the focus on both the hardware and software aspects of embedded systems. Furthermore, students experienced the lab as very difficult since it was not a “walk-through” lab assignment. However, due to the fact that students had find the information from the presented documentation by themselves, they found the lab very educative. In addition, students who regularly attended the lectures found the lab easier than their counterparts that only occasionally attended the lectures. Also, the students were positive towards the discussed grading system since it allowed them to gain bonus points and encouraged them to study regularly.

Based on our own experiences and feedback from the students, we are considering two future directions of the course presented in this paper. First, the hardware components utilized in the current lab setting are showing their age and are in need of substitution for more up-to-date components. This will allow a more complex and elaborate lab assignments, e.g., a complete MP3 player, to be implemented and possibly covering more embedded system topics, e.g. networks. A second possible future direction is to slightly change the lab assignment(s) such that the final grade can be based solely on the lab since it can possibly incorporate all the theoretical knowledge presented during the lectures.

References

[1] http://www.blackboard.com.

[2] Altera University Program: Design Laboratory Package. http://www.altera.com/education/univ/unv-kits.html.

[3] Wayne Wolf. *Computers as Components: Principles of Embedded Computing System Design*. Morgan Kaufmann, 2001.