Introducing Arduino Platform to Sophomore's using an apt recipe

Chirag Parikh
School of Engineering
Grand Valley State University
Grand Rapids, MI, 49504
Email: parikhc@gvsu.edu

Abstract

Microcontroller-based sensing and control has become an important part of many engineering disciplines. Engineering students at Grand Valley State University are introduced to microcontroller-based control in their introductory Digital Systems course which they take in their sophomore level. In this course, students learn to use the Arduino platform to design embedded systems. The Arduino board and programming language is an inexpensive way for faculty to teach embedded system design in introductory courses. However, if Arduino platform is not taught in the right context to the students they may not benefit from using this open-source prototyping platform to learn the basic fundamentals of embedded system design.

The major contention in using any open source community such as Arduino is that they are usually focused on getting things to work as opposed to providing an insight into low-level technical aspects of embedded systems. In spite of this, the Arduino platform still provides students with enough experience and knowledge of some technical aspects of embedded system design by encapsulating them into custom functions. In this paper, we talk about an approach that we have taken so that the students can experience the best of both the worlds namely, Arduino built-in functions and low-level technical details to develop working and interesting designs.

I. Introduction

Microcontroller-based sensing and control has become inevitable in almost every field. An area that has traditionally been reserved for electrical or mechanical engineers is now multidisciplinary, integrating digital electronics, communications and computing with a variety of systems ranging from medical to biological to environmental. The new trend is to take this area beyond the traditional engineering setting and to make it accessible to students from various disciplines in a way that would foster their practical understanding and use of it. Such form of educational learning falls under the category of *Project Based Learning* (PBL) or *Hands-on Learning* curricula. Today, PBL has become a well-known standard implemented by many universities in their curriculum[1,2] to prepare students for the real-world. PBL approach really helps towards students learning as it is a dynamic approach to teaching in which students explore real-world problems and challenges. With this type of active and engaged learning, students are inspired to obtain a deeper knowledge of the subjects they are studying. One of the primary benefits of PBL is that it combines traditional classroom knowledge with real-world expertise to better prepare students for success.

Benefits that students gain from PBL are[3]:

- ***Ownership of Education***: PBL allows students to have a greater control on what and how they learn. This makes students feel more responsible for their work.
- ***Knowledge of real-world skills***: PBL teaches students about real-world skills such as teamwork, time management, communication and many more.
- ***Hands-on***: PBL allows students to become more hand-on and try things that are beyond the scope of traditional classroom learning.

PBL is not only beneficial for students but also for the faculty teaching the course. Benefits that faculty gain from PBL are:

- ***Student Interaction***: PBL in comparison to traditional classroom teaching puts faculty into a role of a mentor that allows for greater interaction with the students.
- ***Beyond the scope of classroom teaching***: PBL teaches students about real-world skills such as teamwork, time management, communication and many more.
- ***Hands-on***: With PBL, learning is not confined to what is covered in the classroom but it allows students to become actively involved in educating themselves.

Faculty in School of Engineering at Grand Valley has taken a step towards this by assimilating PBL using microcontroller platform such as Arduino in their sophomore course *EGR 226: Introduction to Digital Systems*.

This paper describes our approach in providing an apt recipe in teaching the Arduino platform to the sophomore students in our introductory digital system's course. The paper is organized as follows: the next section describes the Arduino platform that we introduce to the students in our Digital System course. We then discuss the format and the structure of the introductory digital system's course followed by our recipe of teaching the introductory course to sophomore students. Then, we describe the development of our laboratory exercises and how they build towards the final project that we assign to students. We finally end the paper with our concluding remarks.

II. Arduino Platform

In order to blend the PBL with the microcontroller, the students were required to purchase a custom-designed microcontroller board as a learning platform in this course. This board was designed at Grand Valley State University to be inexpensive yet robust for educational use. Over time several support materials for this development platform, including course notes, laboratories, a web site with programming resources, etc. was developed. This microcontroller platform was intended to be useful throughout our engineering curriculum and it became a very popular platform to work with. This custom board was economically priced for the student to purchase and use for a variety of projects in their future courses.

But with the popularity of the custom board, came the challenge of providing support to the students who wanted to use the microcontroller board beyond the introductory Digital Systems course in variety of ways. This lead to many questions regarding effective operation and debugging. These questions did not always have easy answers and therefore strained our resources. As a final challenge, the popularity of the microcontroller board also meant that the School of Engineering was responsible for manufacturing hundreds of these boards every year. This in itself is a manufacturing and logistics challenge which further strained our resources and, due to lack of economies of scale, caused the custom board cost to become higher than similar commercially available products such as Arduino.

Arduino was introduced in Italy in around 2005 and started gaining popularity due to its broad *ecosystem*. As they say[4]: "*Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments*". Arduino platform could be used by anyone ranging from high-school kids to novice programmers to experts. A large investment has been made in the board's "ecosystem" (development tools, sample code, add-on hardware products) and that has caused the Arduino to become very popular throughout the world, and is the centerpiece of what has been called the *Maker Movement*[5].

Arduino platform comprises of a microcontroller with various peripheral interfaces that is programmed by an existing software platform written in Java and is based, mainly, on *Processing*[6] (a language developed mainly for artists). Arduino comes in a variety of flavors such as Uno, Nano, LilyPad, Fio and many more. For our Introductory Digital Systems course, we require students to purchase the Arduino Uno board. Arduino Uno (as shown in Figure 1 below) consists of an ATMega328P microprocessor, a USB-to-serial chip, and an AC to DC power converter. Once can install the IDE on a machine and then can program the Uno over the USB. Arduino Uno is very cost effective in comparison to a textbook and it costs approximately $35.

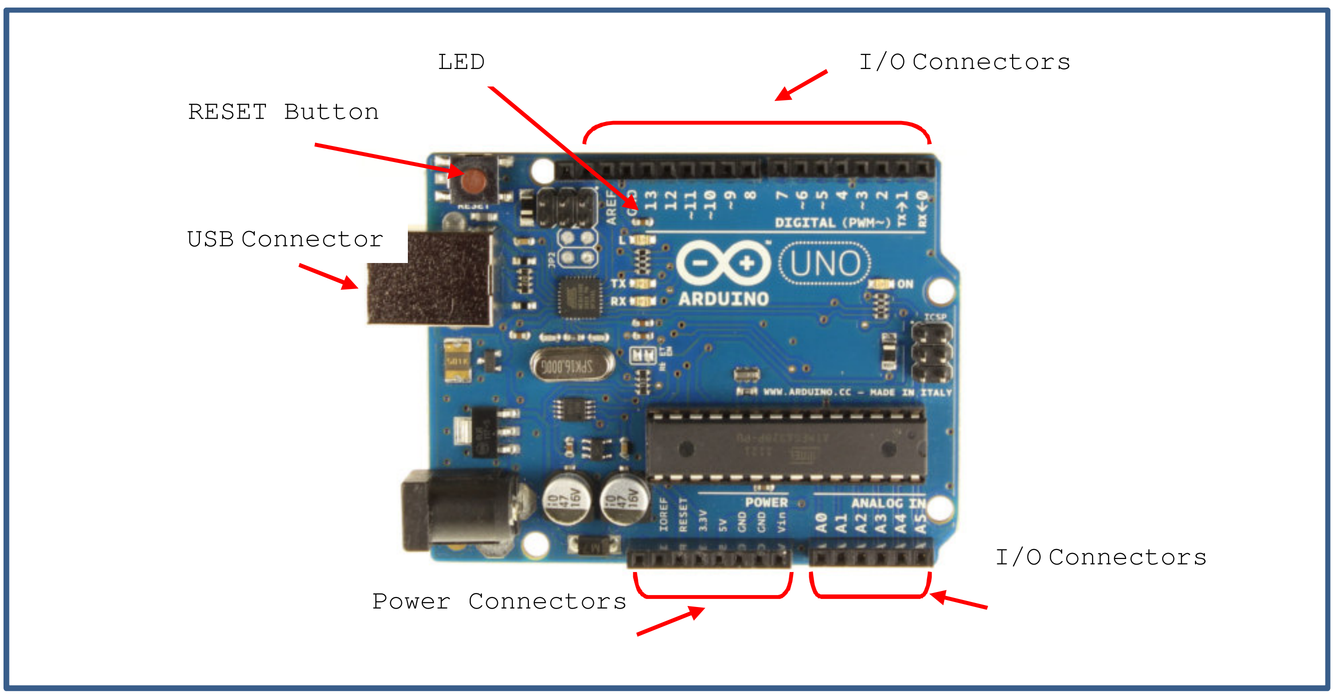

Figure 1: Photograph of the top of the Arduino Uno board.

We have identified Arduino platform to be very beneficial in an educational setting based on our observations as listed below:

- Arduino platform is very easy to setup (plug and play)
- Arduino IDE comes with numerous examples for controlling several peripherals
- Arduino platform is compatible with different OS (Windows, Mac and Linux)
- Arduino platform is very low cost (inexpensive hardware and free software)
- Students can prototype their design very quickly in various programming languages including C

On the other side, we have several concerns about the learning ability of the students when using such platforms:

- Are students really learning the fundamentals of microcontroller programming with the open-source platform?
- Is this approach low-level enough?
- How to assess the level of contributions that open-source projects have in student design?

The next section talks about the way our course has been organized.

III. Course Organization

EGR226: Introduction to Digital Systems is an introductory course offered in digital circuits and systems in the School of Engineering at Grand Valley State University that students take during their sophomore level[7]. This is a semester long course that includes approximately three hours of lecture per week along with a three hour lab session. In our ECE program, the dependence upon this introductory course is strong and all of the concepts are heavily used in various upper-division courses. As this course is a prerequisite for all engineering majors prior to secondary admission, the course focuses on the practical application of digital systems to solve engineering problems. During the laboratory portion of the course, students: design, build, and test various types of digital systems.

A unique feature of the course is its integration of digital system fundamentals, C programming language, and microcontroller interfacing. For the Mechanical and Manufacturing disciplines, this provides a strong foundation for the types of digital system applications with microcontroller interfacing that will be encountered in the upper level courses and senior projects. For the Electrical and Computer disciplines, the depth of content is developed in subsequent courses. The course begins with the fundamentals of digital systems and progresses toward the development of a general-purpose computer. During this development, the basic building blocks of a microprocessor are covered, such as adders, registers, and counters. One of the key outcomes of the development of the microprocessor is the shift from hardware design to software design. The end of the course focuses on the programming of a microcontroller and the development of software to solve simple engineering applications.

Some of the main goals of the course are (1) to teach students the general structure and operation of a microprocessor-based system and (2) to teach students to write software for a microcontroller to control an electromechanical system. Throughout the second half of the course, basic concepts of a microprocessor based design are introduced by way of examples that involve working with several peripherals of a microcontroller. During the microcontroller aspect of the course, we introduce students to various topics such as basic architecture of a microcontroller, general purpose Input/Output (GPIO), timers, interrupt mechanism and A/D conversion. Students also learn about hierarchical design where lower-level blocks are designed first and instantiated as per the need to form a complete system. For example, students learn to write software to control timers and interrupts and then use this knowledge to control the speed/direction of a motor using pulse-width modulation.

The laboratory portion of the course is divided into two types of exercises: Laboratory exercises and Projects. The laboratory exercises are intended to familiarize the students with the microcontroller platform (Arduino) and its peripherals as discussed in lecture. Conversely, the projects are fairly open ended. The projects build upon the earlier laboratory exercises and then expand these systems. The result is a more sophisticated system with a practical application. The next section presents on our approach to answer some of the concerns (mentioned in Section II) by presenting our recipe of introducing Arduino platform to the sophomore students.

IV. Our Recipe

Though Arduino platform has captured hearts of many enthusiasts, our concern is that *Should Arduino and its ecosystem of open source projects and resources be used as a platform to teach introductory embedded system design?* In order to be able to answer this question, we first had to identify the concepts that students should learn in our introductory course. We figured that students should have a working understanding of the architecture of the microcontroller along with its basic peripherals such as GPIO, timers, interrupts and A/D conversion. Students should be able to setup and interact with these peripherals independent of the microcontroller platform. Arduino platform has a rich database of several pre-built high level functions that would hide the low-level complexity from the users. This would defeat the purpose of one of our goals mentioned in the previous section.

Our approach was as follows:

- Direct students to the pre-built functions available in the Arduino database
- Teach students to write their own custom functions using low-level registers that has the same functionality as the pre-built functions
- Make students use their custom functions instead

We feel that this approach not only teaches students the basic functionality of the microcontroller and its peripherals but it also makes student not depend on the database of the functions available for them to use. We feel that with this approach students will experience a large majority of basic concepts with the added benefit of building working and interesting designs. Also, having seen the database of pre-built functions students can use them in their upper division courses to target complex projects. In the next section, we talk about the sample laboratory exercises and kinds of projects that students undertake.

V. Sample Laboratory Assignments and Projects

The course laboratory exercises are meant to help students design and understand the use of microcontroller in an electro-mechanical system. The course laboratories are structured as four assignments and one project. They cover the following topics: introduction to the microcontroller platform; general purpose input output; timer peripherals; interrupt mechanism and analog-to-digital conversion. These laboratories expose the students to become familiar with the microcontroller platform and learn to program a microcontroller to perform the desired functions. Each laboratory has a set of deliverables; typically including the successful demonstration of a working design, source code with comments, and a brief report.

The following is an outline of the four laboratory assignments that we offer in the course:

- **Laboratory #1**: The purpose of the laboratory was to introduce students to the microcontroller platform that they use in the course. The laboratory teaches students to write and upload C programs for the microcontroller platform. The laboratory also provides an insight into some of the pre-built functions that is available through the open source database.

- **Laboratory #2**: The purpose of the laboratory was to teach students to work with GPIO of microcontroller and communicate with Arduino over serial interface. To be able to do this we provide students with an add-on shield (Gadget shield as shown in Figure 2) that has several peripherals and sensors that students can take advantage of. In this laboratory, students are required to display the least significant 4-bit representation of the character user enters on the terminal window on the Gadget shield LED's.

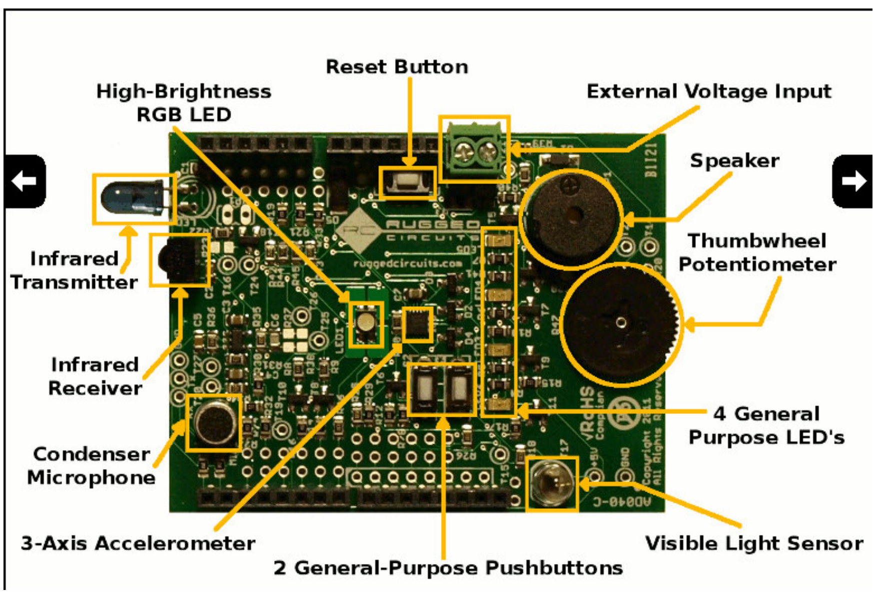

Figure 2: Sensors and actuators on the Gadget Shield

- **Laboratory #3**: The purpose of the laboratory was to teach students to interface Arduino with the LCD. Again we provide students with the LCD module that fits on the Arduino board. In this laboratory, students are required to design Rock-Paper-Scissors game that the user plays with the microcontroller. For this laboratory, we allow students to use some of the pre-built functions for random number generation. Sample output of the game is shown in Figure 3 below.

Figure 3: Sample output for Rock-Paper-Scissors game

- **<u>Laboratory #4</u>**: The purpose of the laboratory was to teach students the principle of pulse-width modulation (PWM). In this laboratory, students were required to design a menu-driven LED dimmer application that varies the intensity of the LED on the Gadget shield based on user input.

- **<u>Final Project</u>**: The final project is spread over two weeks where students learn about interrupt mechanism and analog-to-digital (A/D) conversion and apply those concepts during the first week of the project. In the second week, students integrate everything together (knowledge about GPIO's, timers, PWM, interrupts, A/D conversion) to create a working application. The students are given two weeks to design and implement the project by using pre-designed components from the laboratory assignments and/or creating new components from scratch. Sample projects that the students have designed in the past are Music Synthesizer, DC motor speed/direction controller, and may more.

VI. Conclusion

In this paper, we have described our approach to teaching introductory microcontroller concepts to sophomores by integrating the open source platform such as Arduino with the low-level details of microcontroller based design. We demonstrate to the students the benefits of using open source platform for a more advanced design but still instill in them the low-level approach of interfacing with the microcontroller. Our assumption is that if they know how to work with the microcontroller at a low level, they should be able to work with any microcontroller without being dependent on the open source platform to guide them. We do not discourage students from using the open source reference but we encourage them to use it for more challenging projects that they would undertake in their upper division courses.

We also describe the details of our course and show how individual laboratory exercises build towards the project. The majority of the laboratory activities in the course are organized around the PBL pedagogy that provides students with adequate practice with real-world systems at an elementary level. We conclude that by transitioning from using custom designed board to a more open source development board has provided students to work with more creative projects due to the availability of numerous add-on shields that can sit on top of the Arduino board.

Bibliography

1. C. L. Dym, A. M. Agogino, D. D. Frey, and L. J. Leifer, "*Engineering design thinking, teaching, and learning*," Journal of Engineering Education, vol. 94, pp. 103–120, 2005.

2. J. Macias-Guarasa, J. Montero, R. San-Segundo, A. Araujo, and O. Nieto-Taladriz, "*A project-based learning approach to design electronic systems curricula*," Education, IEEE Transactions on, vol. 49, no. 3, pp. 389 – 397, 2006.

3. J. E. Mills, and D. F. Treagust, "*Engineering Education—Is Problem-Based or Project-Based Learning the Answer?*" Australian Journal of Engineering Education, *http://www.aaee.com.au/journal/2003/mills_treagust03.pdf*.

4. Arduino, Available at http://www.arduino.cc, 2010.

5. Wired Magazine Article ("Epicenter"), August 6, 2011, last accessed March 18, 2012 available at *http://www.wired.com/epicenter/2011/08/big-diy/all/1*,.

6. C. Reas, B. Fry, and J. Maeda, "*Processing: A Programming Handbook for Visual Designers and Artists*," The MIT Press, 2007.

7. A. J. Blauch, and A. Sterian, "*A Practical Application Digital Systems Course For All Engineering Majors*", Proc. of ASEE 2002 Annual Conference, Montreal, Quebec, Canada, June 2002.

CHIRAG PARIKH

Chirag Parikh is currently an Associate Professor in the Padnos College of Engineering and Computing at Grand Valley State University. He received his B.S. in Computer Engineering from Fr. Agnel College of Engineering, University of Mumbai, M.S. in Electrical and Computer Engineering from University of Texas at San Antonio, and Ph.D. in Electrical and Computer Engineering from University of Texas at San Antonio. He has taught courses on integrated freshman engineering, structured programming, digital systems, microcontrollers, FPGA and cryptography.
http://claymore.engineer.gvsu.edu/~parikhc