An Application-Based Freshman Introductory Programming Course using the Raspberry Pi

W. Lawson, S. Secules, and S. Bhattacharyya
The University of Maryland, College Park, lawson@umd.edu, secules@umd.edu, ssb@umd.edu

***Abstract* - An innovative approach to teaching an introductory C programming course to freshman electrical engineering students has been developed. The innovation stems from the use of electrical engineering applications and projects to motivate students to master language syntax and implement key programming concepts and best practices. Two lectures per week cover programming concepts, introduce hardware and discuss applications. Weekly laboratory sessions center around writing C code on a Raspberry Pi (RPi) computer to interact with a variety of sensors, actuators, and electronic components and achieve laboratory goals. The laboratory experience culminates with two multi-week hardware projects designed to challenge the students' new knowledge and skills.**

The new course has been run in parallel with a traditional introductory C class. Program evaluation has been conducted by a research team which operates separately from but advises the team of instructors about course improvements. Results show that students in the alternative course find it more collaborative, less competitive, and having a greater sense of community than students in the traditional class.

Index Terms – application-driven, C language, freshman programming, Raspberry Pi.

INTRODUCTION

For several decades now there has been an increasing emphasis on using active-learning in freshman engineering courses [1-3]. A central feature of active-learning settings is the affordances for collaborative settings and student-centered instruction, which have been shown to have cognitive, affective, and persistence advantages for students [4]. While a large number of these efforts have focused on freshman design courses, there has been some effort to shift the emphasis to introductory programming courses. A standalone computational platform in the form of a micro-processor is often used as the "brain" of a design project; likewise, a microprocessor can be necessary when moving programming instruction from didactic, lecture-based, and professor-centered settings to spaces where students have more agency to explore programming applications in real time and relevant settings.

The principle goal of this project is the development and delivery of an application-based programming course for freshman that emphasizes both software and hardware components that are typical of simple embedded applications. The course assumes no programming experience and no introductory orientation to the electrical engineering (EE) profession. It is offered as a 3-credit alternative to a traditional software-only 2-credit introductory C programming course. The credit difference between the two courses allows the lecture time needed to introduce the hardware segment of the course.

The course requires both individual software-only programming homework and application-driven assignments in which the students write code to interact with hardware. Virtually all programming assignments have a connection to the EE discipline. This project-driven course involves two hours of lecture and one three-hour lab session each week. In addition to mastering the student learning outcomes of our traditional introductory programming course, students in our course are introduced to many concepts from the electrical engineering discipline, including elements of circuit theory, electromagnetics, communications, and control systems.

The research component of this proposal is designed to measure whether or not, and to what extent, the course achieves the student learning outcomes. It will also contribute to basic research on how students' epistemological stances towards programming influence their actions during programming.

COURSE CONTENT AND STUDENT LEARNING OUTCOMES

The course lectures are divided into 18 modules of varying lengths. The module titles are given in Table I. The ten programming modules are presented in order and cover the same material as in the traditional C programming course with the exception of a unit on Unix. The first module touches on most features of the language in a relatively superficial way and the remaining modules explore each topic in greater depth. The eight hardware modules are on average much shorter than the programming modules and are inserted into the lectures as needed for the students to be successful in the laboratory.

The Student Learning Objectives are as follows. All students who pass this course will have an:

- Operational familiarity with elementary programming concepts: program flow, data types, arrays and memory, logic and arithmetic operations, input/output and functions.
- Ability to utilize good programming practices to write efficient, clear, and maintainable code.

- Ability to use an IDE to write, debug, load and run code to solve engineering problems, perform basic calculations, and input and output meaningful data.
- Appreciation for the enabling role of programmable devices in technological systems and applications.
- Understanding of the operation of basic electronic components, sensors and actuators.
- Ability to work effectively in teams.
- Ability to communicate effectively in written and oral formats.

TABLE I
THE NOVEL C PROGRAMMING COURSE MODULES.

1	A crash course in C programming	10	Introduction to Unix
2	Data types	11	The Raspberry Pi and the GPIO
3	Operators	12	Introduction to basic circuit components
4	Program selection	13	Introduction to sensors
5	Repetition	14	Introduction to op-amps, diodes and transistors
6	Functions	15	The SPI interface
7	Arrays	16	Introduction to A/D converters
8	Input / output formatting	17	The I^2C interface
9	File input / output	18	Introduction to mux/demux chips

COURSE STRUCTURE AND FINAL PROJECTS

Each semester there are nine labs and two final projects. About one-third of the labs are designated individual labs and the other two-thirds are designed to be done in groups of two. While some of the labs can be finished in three hours, many are to be completed outside of regular lab time, and students carry their micro SD cards to and from the lab for continuity. A summary of the lab goals is given in Table II.

The group project is designed for groups of 3 to 4 students. Groups are assigned after about 1/3 of the semester and have various preparatory tasks to perform in the middle third of the semester before the project begins in earnest the final third of the semester. The final project involves an autonomous vehicle using sensors to navigate a simple maze from start to finish, and then a return to the start of the course without using sensors. For the first three offerings of the course, we used a modified RC tank as the vehicle for the project (see Fig. 1.)

The final individual project has been to detect and interpret a Morse code signal from an LED. A test code randomly selects up to four sentences to transmit. The student must build the hardware to detect the LED output and write the code to translate the code.

RESEARCH METHODOLOGY

Course evaluation has been conducted by a research team which operates separately from but advises the instructional team about course improvements. Overall sentiments and experiences are fed back in regular meetings. The mixed research methods include student surveys, classroom observation, and student interviews

TABLE II
THE PRINCIPLE LAB GOALS FOR THE NOVEL C PROGRAMMING CLASS

Lab	Content/Goals
1	Assemble RPi Kit and write simple code to output message
2	Generate a code that allows you to type in a sentence and then have an LED blink the sentence in Morse code
3	Write a code that will turns lights (LEDs) on when lights are off and keep track of where (e.g. in a house) the lights are on
4	Learn to use the MCP3008 A/D converter. Write codes to (a) get data from analog temperature sensors, (b) calibrate an IR distance sensor, and (3) use a calibrated IR distance sensor to measure distances to objects.
5	Generate two codes for a 3-axis analog accelerometer. The first code is used to calibrate the sensor. The second code is to measure and record accelerometer data with a calibrated sensor and attempt to discern velocity and distance.
6	Generate two codes for a 3-axis digital magnetic sensor. The first code is used to null and calibrate the sensor. The second code is to measure and record magnetic field data with a calibrated sensor.
7	Generate a code that interprets the data from an acoustic distance sensor to estimate distance to objects and to identify and ignore outliers in the data.
8	Generate a code that utilizes a servo motor and a magnetic sensor to track a moving permanent magnet
9	Write a code to use two digital magnetic sensors and a mux/demux chip to make a magnetic gradiometer. Write a code to calibrate this device.

During a pilot offering of the course, we conducted semi-structured interviews with students in both the traditional and novel course offerings. Interview questions addressed student perceptions of a few key areas: group work versus individual work, skills needed for programming, and identity/belief/efficacy related to programming. Our analysis of these interview transcripts helped formulate a survey instrument which highlighted some of the key cognitive, affective, and experiential differences emerging from the traditional and novel course student populations.

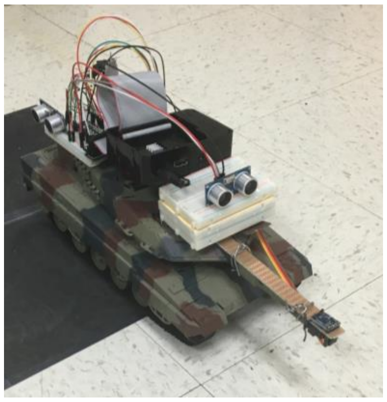
FIGURE 1
A TYPICAL HARDWARE PRODUCT FOR THE FINAL GROUP PROJECT

In each of the three terms that the course has been offered, we administered surveys to compare student responses in the novel and traditional courses. Students were surveyed at the beginning of the semester, at the end of the semester, and in the subsequent semester after taking the class.

RESEARCH STUDY RESULTS AND DISCUSSION

Sample survey results are shown in Table III. A score of 1 means complete disagreement, a 4 means neutral, and a 7 signifies complete agreement.

TABLE III
SURVEY RESULTS FROM THE NOVEL COURSE FOR THE IDENTITY/BELIEF EFFICACY SECTION. (* INDICATES STATISTICALLY SIGNIFICANT DIFFERENCES.)

Survey question:	Mean	Standard Deviation
PRE- I feel like I fit in as an electrical engineer.	5.6*	1.1
POST- I feel like I fit in as an electrical engineer.	6.4*	0.5
PRE- Programming is not “real engineering”	2.5	1.3
POST- Programming is not “real engineering”	2.1	1.2
PRE- I want to take more programming classes beyond this class, even if they weren’t required.	5.6	1.5
POST- I want to take more programming classes beyond this class, even if they aren’t required.	5.4	1.4
PRE- I’m excited about the electrical engineering major.	6.0	1.4
POST- I’m excited about the electrical engineering major.	6.5	0.5
PRE- Coming into this class, I feel confident that I can learn coding.	6.1	1.1
POST- Going into <my next class>, I feel confident that I can learn coding.	6.5	0.8

The following statements summarize our survey results from Fall 2014 and Fall 2015 course cohorts.

There were gains in self-efficacy and identity measures for novel course offering pre- to post-, with statistically significant ($\alpha = 0.05$) gains in matched samples t-test on the statement “I feel like I fit in as an electrical engineer.” This tentatively contrasts with initial data from the 2015 cohort that the first semester of the traditional programming course can decrease average scores on identity and self-efficacy measures.

The novel course offering produces a higher appreciation for, and enjoyment of, group work than the traditional course, and a statistically significant gain in terms of recognizing its importance in the students’ future professional programming activities (Independent samples t-test, $\alpha = 0.05$) .

Upon reflection, in a lecture-based programming course the subsequent term, students from the two course offerings were asked to rank their first term course (traditional or novel) or the subsequent course (same traditional course) on a few key dimensions. Students from the novel course rated it as more collaborative, less competitive, more like real world engineering, and having a stronger feeling of community than their current traditional programming classes. The measures of collaborative and real-world engineering were statistically significant on an independent samples t-test ($\alpha = 0.05$) when compared with traditional course students.

Perhaps this final contrast (that students identify the novel course as more like “real world engineering”) is the most promising finding. Even in spite of work which is difficult and taxing, a student who desires to be an engineering major may find some level of comfort in believing their effort is put towards an authentic engineering challenge, as opposed to arbitrary and difficult tasks disconnected from their intended professional practice.

SUMMARY AND FUTURE WORK

Our novel application-based introductory C course has had a total of about sixty students over three semesters, limited by resources. Our retention rate has been about 95%, with one student failing, one student leaving because he decided his previous programming instruction made this course unnecessary, and a third student leaving early in the semester. All other students successfully completed the course.

We expect to offer this course indefinitely in the future in parallel to the traditional course as an alternative choice for incoming freshman who would prefer a more application-driven course. The course will be continuously updated and improved. For example, in Fall 2016 the RPi 3 (with faster processing and integrated Wi-Fi) will be used in the course, tanks will be replaced with a robot car chassis, and an off-the-shelf motor shield will replace the custom PCB interface.

ACKNOWLEDGMENT

This work is supported by the National Science Foundation under grant DUE- 1245745.

REFERENCES

[1] Dally, J. W., and G. M. Zhang, “A Freshman Engineering Design Course,” *Journal of Engineering Education*, pp. 83-91, April 1993.

[2] Meade, J., “Change is in the Wind”, *ASEE PRISM*, 2, May 1993, pp. 20-24.

[3] Parker, J., D. Cordes, and J. Richardson, “Engineering design in the freshman year at the University of Alabama-Foundation Coalition program,” *Frontiers in Education Conference, 1995.* Proceedings. 1995 (Atlanta, GA), pp. 4d2.5 - 4d2.8 vol.2

[4] Prince, M. (2004). “Does Active Learning Work? A Review of the Research.” *Journal of Engineering Education*, 93(3), 223–231. http://doi.org/10.1002/j.2168-9830.2004.tb00809.x

AUTHOR INFORMATION

W. Lawson Professor, University of Maryland, College Park, lawson@umd.edu

S. Secules Doctoral Candidate, University of Maryland, College Park, secules@umd.edu

S. Bhattacharyya Professor, University of Maryland, College Park, ssb@umd.edu