Embedded computing reinforces and integrates concepts across ECE curriculum

Dr. Harry Courtney Powell, University of Virginia

Harry Powell received the B.S. in Electrical Engineering the University of Virginia in 1978, a M.S. in Electrical Engineering in 2006, and the Ph.D. in Electrical Engineering in 2011. Dr. Powell spent over 20 years in industry designing computer controlled automated systems before returning to academia in 2001. He was appointed to the faculty in 2013, and teaches courses in electric and electronic circuit analysis, electromagnetic energy conversion, embedded computing, and the 4th year Major Design Experience. Joanne Bechta Dugan was awarded the B.A. degree in Mathematics and Computer Science from La Salle University, Philadelphia, PA in 1980, and the M.S. and PhD degrees in Electrical Engineering from Duke University, Durham, NC in 1982 and 1984, respectively. She has performed and directed research on the development and application of techniques for the analysis of computer systems which are designed to tolerate hardware and software faults. Dr. Dugan is an IEEE Fellow. She was Associate Editor of the IEEE Transactions on Reliability for 10 years, and is currently Associate Editor of the IEEE Transactions on Software Engineering. She served on the National Research Council Committee on Application of Digital Instrumentation and Control Systems to Nuclear Power Plant Operations and Safety. She is also a member of Eta Kappa Nu, and Phi Beta Kappa. Previously, she taught at Duke University and worked as a visiting scientist at the Research Triangle Institute.

Prof. Joanne Bechta Dugan, University of Virginia

Joanne Bechta Dugan is Professor of Electrical and Computer Engineering and the Director of the Computer Engineering Programs at the University of Virginia. Her research focuses on probabilistic assessment of the dependability of computer-based systems. She has developed the dynamic fault tree model, which extends the applicability of fault tree analysis to computer systems. Current work focuses on the development of new technologies and engineering approaches to evaluate and improve engineering education, both in traditional classroom setting and in non-traditional on-line settings. Dugan holds a B.A. degree in Mathematics and Computer Science from La Salle University, and M.S. and PhD degrees in Electrical Engineering from Duke University.

Embedded Computing Reinforces and Integrates Concepts Across the ECE Curriculum

Introduction

An examination of the Electrical and Computer Engineering curriculum in many programs across the United States reveals that there is a large commonality in course content and program requirements. Many ECE curricula begin with linear circuits and digital logic followed by electronics, signals and systems, electromagnetic fields, and so forth. While there have been pedagogical advances, i.e. "flipped" classrooms[1], problem-oriented instruction or interactive course content, there has been much less work in combining concepts across the entire spectrum of the curriculum[2]. We have observed that many students have a strong tendency to put the concepts from each course into its own "box", and consider them to be unrelated to the concepts and materials in other "boxes".

Contrast this state of affairs with the modern engineering environment, especially for electrical and computer engineers. Virtually any non-trivial system - an electronic automotive engine control unit for example - requires an understanding of concepts from across the entire spectrum of ECE. Furthermore the central component that integrates these concepts is an embedded computer. As a means of reducing the tendency to "boxing", we use our introductory course in embedded computing to motivate and contextualize concepts from across our curriculum.

Many ECE curricula have a course in microcontrollers or embedded computing, covering topics such as digital input/output (I/0), analog to digital conversion (A/D) or the use of timers. While this is certainly prerequisite knowledge, little is done to relate these concepts to other parts of an ECE curriculum[3]. In our approach, we perform an explicit link with embedded computing concepts to subject material from elsewhere in the curriculum that would directly employ the topic at hand[4].

Our class is targeted at 3rd year students. At this point in their curriculum they will have had introductory courses such as *Digital Logic Design*, *Linear Circuits*, and *Electronics*; *Signals and Systems* is co-requisite and electrical engineering majors will also be taking *Electromagnetic Fields*. As they move from this course into the 4th year, they may choose to take elective courses in controls, communications, wireless design, or networking.

Course Structure

Although this course includes both a lecture and laboratory component, laboratory work is the heart of this course. It has been shown that problem-based learning and experiential techniques are effective at cementing concepts across engineering disciplines, including embedded computing[5]. Our laboratory experiments also include a strong design-based element and are not simply "cookbook" exercises. This is crucial to developing the critical reasoning skills required of an engineer and is most effectively introduced before the required 4th year *Capstone* design sequence[6].

Each lecture is tightly integrated with subsequent laboratory work, and studio techniques are often employed, wherein the laboratory and lecture occur within the same physical space and overlap with each other. Furthermore we have designed low cost hardware based on industry-standard components that enables students to own virtually all of the required course material.

This encourages experimentation outside of the traditional laboratory environment, especially since students have 24/7 access to the laboratory space and equipment.

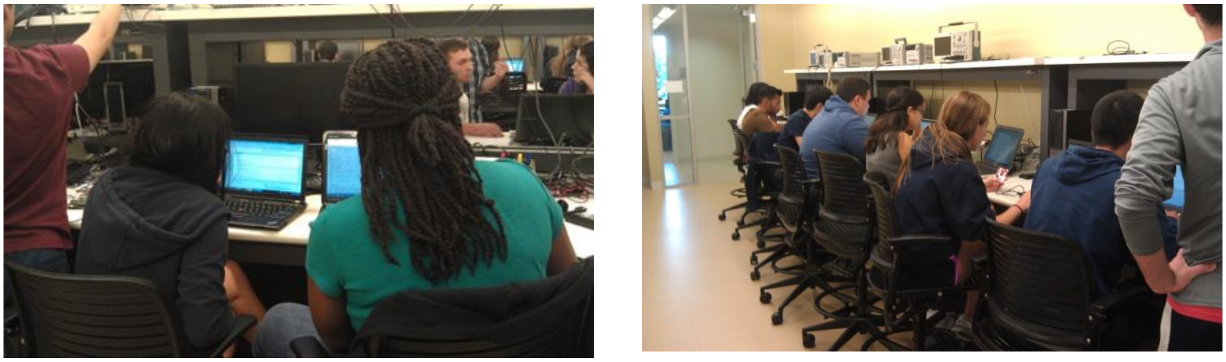

Figure 1. Students working and learning in the lab

The class is structured with a weekly assignment which consists of 2 components: an in-lab experiment and a larger project. The Monday lecture reviews last week's experiment and project, typically beginning with a brief on-line quiz aimed at a summary assessment of the previous week's activities. This provides us with immediate feedback and enables us to clear up misconceptions while the subject matter is still fresh with the students. We then introduce the current experiment/project and provide background on its relevance to other aspects of their course work. To the extent possible, we include in-class activities to facilitate learning.

The laboratory meets Tuesday & Thursday mornings (each with half the class) and often begins with a short discussion/presentation of concepts related to the specific lab assignment. While the Monday lecture may dwell more on the "big picture" background material, the in-lab lecture is brief and usually focuses on more specific material, i.e. timer programming, which would be required for the assignment at hand. The lab assignment, which must be completed during the assigned laboratory time, is a subset of or is closely related to the week's project. The project is completed outside of the normal laboratory meeting time and the total weekly assignment is due Sunday midnight.   Friday afternoon office hours are held in the lab and are well attended.

Each of the weekly assignments has baseline requirements and optional challenges; the challenges are intended to provide a deeper level of understanding and are used for bonus points. It should be noted that those who complete the challenges rarely need the extra points! Students who meet the challenges, demonstrate good debugging skills and submit high quality reports and documentation are invited to serve as (paid) teaching assistants (TA's) in later semesters. We have informally noticed that having capable undergraduate assistants tends to inspire some students; "if they can do it maybe I can too". This results in a net very positive feedback in the overall class experience.

Each experiment is designed with two goals in mind. The first is to understand and employ an embedded computing concept, i.e. analog to digital conversion, within the context of a programming assignment with functional, performance and documentation requirements.   The second equally important goal is to require the use of concepts from elsewhere in the curriculum in order to complete the experiment successfully. Some concepts have been seen in other courses and reviewed briefly; other concepts are new and require an introduction and guidance.   For example, one of our experiments involves signal acquisition and filtering to reduce the effects of interference. As part of this sequence, students review sampling concepts from signals and systems, and efficient arithmetic manipulations from digital logic design. Throughout the course, concepts merged with embedded computing include power electronics, feedback and control, and communications theory as well as computing concepts such as algorithm analysis, dataflow, state machines, testing and validation. Unfamiliar concepts are treated superficially; however, the instructors in the later courses revisit the embedded experiment as part of their more thorough concept exploration. Figure 2 illustrates the relationship between embedded computing concepts and other courses and topics across the curriculum.

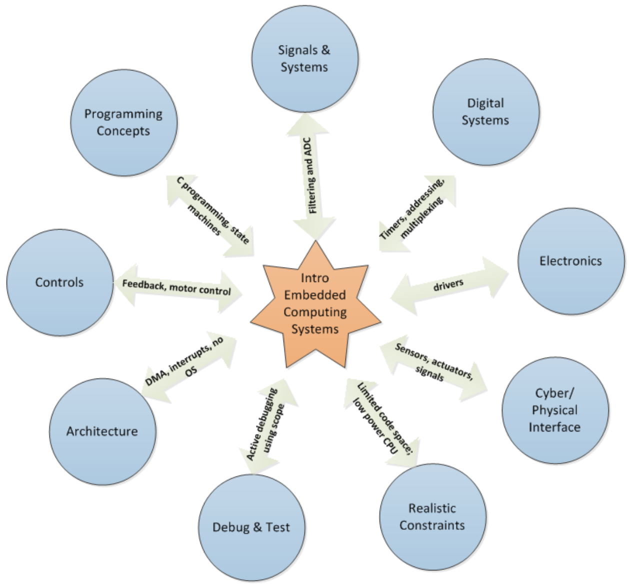

Figure 2 Embedded nexus of ECE curriculum

Course goals

Our primary motivation for adding the course was to incorporate this important area into our (somewhat dated) curriculum. The change was not driven by unsatisfactory assessment results or by employer, alumni or advisory board feedback. Even though we are a large public university with a mandated ratio of two in-state students for each out-of-state student, our students are well-prepared for the rigors of an engineering curriculum and have consistently met our targets for student outcomes. They are high-achievers who succeed in the public and private sector and in academia, so our assessment data is unlikely to be able to demonstrate success. In fact, since our goal was to add a new required course, we could simply assert that we met the goal since the course is now required for both electrical and computer engineers. Of course we wish to evaluate the results and continue to improve the course and the integration with the rest of the curriculum.

Despite consistently meeting/exceeding our student learning outcome targets, our exit surveys have been calling for more hands-on learning. Comments such as “more projects,” “more experiential learning,” “more hands-on learning” are the most common free-text response to the summary question “What specific suggestions can you offer to improve our curriculum?” in the exit surveys administered in 2010, 2011, 2012 and 2013. In fact, such comments comprise just more than half of the suggestions in each year. Some comments in 2013 specifically refer to the new embedded computing course in a call for more such courses. About a third of the 2013 graduates would have taken the new course; most of the 2014 graduates in both programs will have taken it. It's been offered in both the spring and fall semesters in 2012 and 2013 to about 175 students total. We expect the exit data from 2014 to provide some evaluative feedback from the students' perspective. Where we expect to see more impact is in the major design experience class (*ECE Capstone*) and later alumni surveys.

Artisanal hardware

As described in more detail in[4] our embedded computing course uses custom-designed and locally manufactured header boards interfaced to the Texas Instruments *MSP430 LaunchPad* [7]. Programs are written in C and are developed using TI's *Code Composer Studio*[8].

Many introductory classes in embedded computing use components that are designed for easy integration and hide the details. Indeed many of our students have used such introductory robotic systems in high school, first-year intro classes or in extracurricular activities. Our approach in this class is instead designed to confront the details to gain a more complete understanding of interfaces and signaling. We also want students to work directly with industry-standard parts and components and to gain experience with datasheets and other raw documentation. Furthermore, each of the boards was designed from a pedagogical standpoint to facilitate the use of concepts across the entire ECE spectrum rather than simply providing a flashy project.

To achieve these goals, the first author has designed a set of custom circuits that connect neatly to the *MPS430 Launchpad*. Figure 3 shows one example header board that includes a 3-axis accelerometer, an LED driver, a ring of 8 LEDs and numerous test points for attaching clips and leads. Using this header board requires students to learn to interface the accelerometer to the A/D converter on the LaunchPad, calibrate the readings, derive an output (to light the LEDs) and use SPI to send the output to the LED driver. Six such header boards have been developed by the first author of this paper; we work with a local electronics firm to manufacture the boards. We believe that using "artisanal" boards demonstrates what a small group of designers can achieve and encourages entrepreneurship by example.

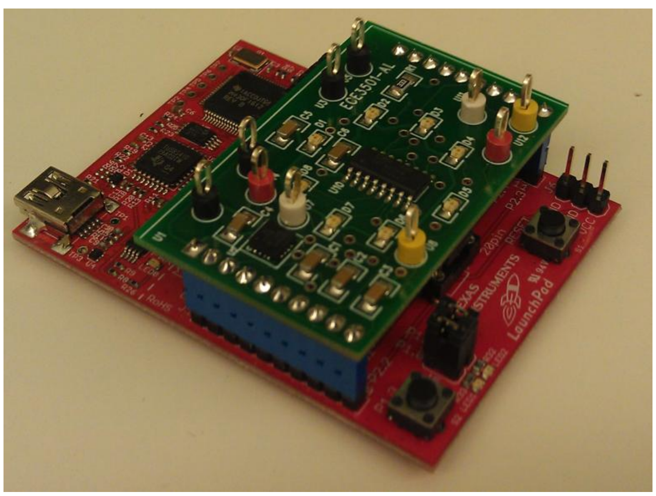

Figure 3 MSP430 Launchpad with custom header board

Experiments teach concepts

Each experiment is designed with several goals in mind. The first is to provide an understanding of how to design a program that employs a basic embedded computing technique such as analog to digital conversion or switch debouncing. The second equally important goal is to use this experiment as a means of illustrating relevance to the concepts from the overall ECE program. Below, we describe several of these experiments.

1. Pulse-width modulation (PWM)

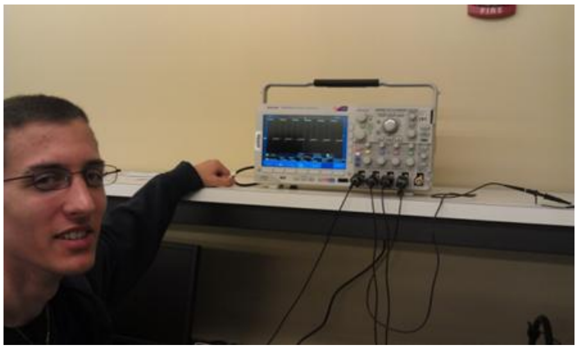

Figure 4. PWM Signal

Pulse-width modulation is a topic that introduces and reinforces several concepts. In *Digital Logic Design*, a prerequisite course, students learn to build a timer system from a register, comparator and counter, and they see how a timer is used to achieve simple PWM. In the embedded course we revisit this timer system (since it was modeled on the timer in the MSP430 microcontroller) and students write a program to modulate the brightness of an on-board LED. They use interrupts to effect the control and use the scope to visualize their output signal; they find the visualization on the scope especially satisfying (see Figure 4).

What can be learned from such a simple experiment? At first glance it appears to be a relatively simple exercise and in a typical embedded computing course it might be left at that[9]. However we can expand upon how the experiment is approached to elicit deeper understanding, not only within embedded computing, but across the general body of knowledge required to be an effective engineer. From the standpoint of embedded programming, the students learn how to manipulate timer subsystems to best advantage, minimizing the overall load on the processor. Also, the impact of how PWM frequency affects processor resources and flexibility of programming approaches is considered. However, we also relate this to concepts from basic circuits and electronics. The students observe that the end goal is to change the brightness of a LED and we show how doing it employing PWM uses less electrical energy than using a variable resistor to change LED current. The simple calculus to prove this is revisited and serves as a reminder that the concepts from college level mathematics are indeed relevant elsewhere. We also revisit the use of transistors as switching elements, a core concept from electronics.

2. Quadrature rotary encoder

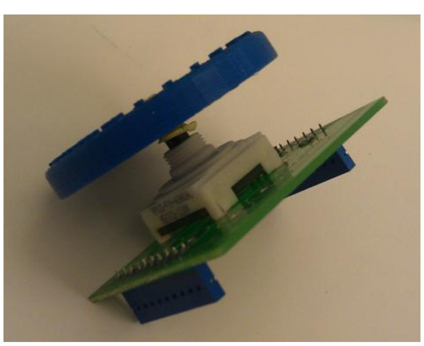

Figure 5 Mechanical quadrature encoder

Quadrature rotary encoders are one of the most commonly used data input devices, both in laboratory bench equipment and as an industrial feedback element for motion control and positioning systems. Experiments in reading values from a rotary encoder (Figure 5) build on a previously developed switch debounce experiment code and reinforce state machine concepts. Although students have seen state machines in a discrete math course and in logic design, they gain a greater appreciation when using them to describe desired program behavior. A typical experiment in this sequence is to develop an electronic combination lock, i.e. a combination of "left" and "right" clicks are used to unlock the system whose success is indicated by illuminating a LED.

Considered in isolation, this experiment might be considered to be entirely embedded computing centric. However, we employ it in a transitional sense, and to reinforce several concepts that will be vital in later more complicated projects. The most important is the concept of code reuse as the students are informed that this code will also be employed in later experiments in which the encoder is used as a feedback element in a control system.

3. Motor control

In a subsequent experiment we use PWM to control a motor. (Figure 6) In this experiment we are able to expand upon the earlier LED experiment, while retaining the essential embedded concepts previously developed.

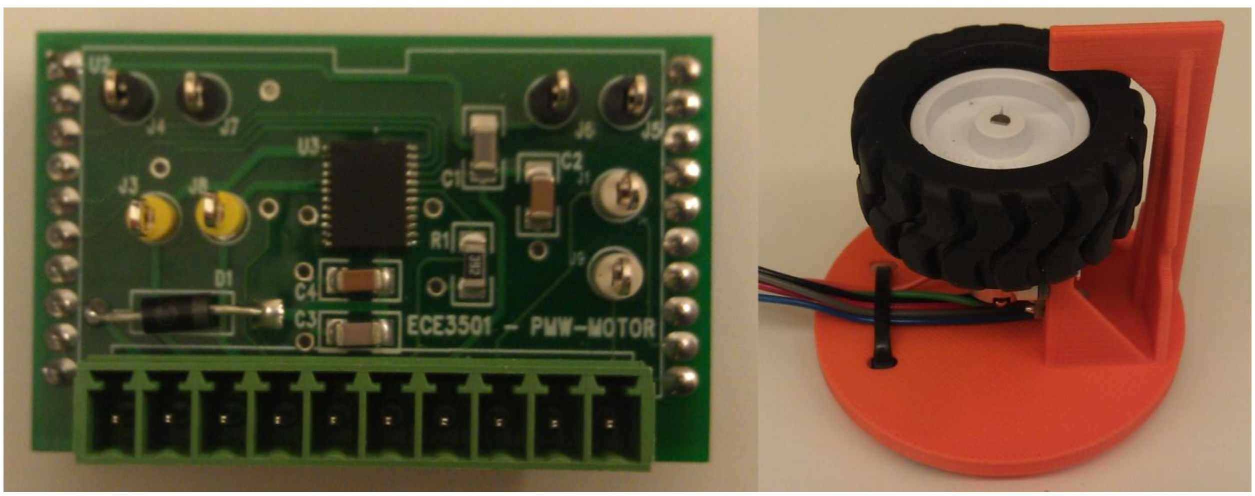

Figure 6 H Bridge control and motor

The PWM signal is generated by the microcontroller and applied to an "H Bridge" controller to drive the actual motor. The motor has an integral quadrature encoder built in that is employed as a feedback element. A typical experiment involves starting the motor at zero speed, ramping speeds to accelerate to a desired target rate, then decelerating to stop at a predetermined position.

Our embedded pedagogy is such that we may concentrate on the interesting parts of this problem, rather than the PWM and quadrature encoder topics, which consists of code that is largely reused from earlier experiments. We focus on techniques that ensure numerical stability and efficient methods for transition through PWM values as the motor accelerates and decelerates. Concepts from elsewhere in the curriculum include a basic introduction to direct current motor theory, relating torque to current and speed to voltage, with an simple model of losses and their impact on performance. Although most students will not have had *Controls* at this point, we do set forth an elementary foundation for them to work from, and the actual control technique employed in the experiment is adaptive "bang-bang" which is a very simple one for them to grasp. We allow them to experiment with the adaptation rate, which provides an introductory and intuitive sense of stability.

4. RF communications

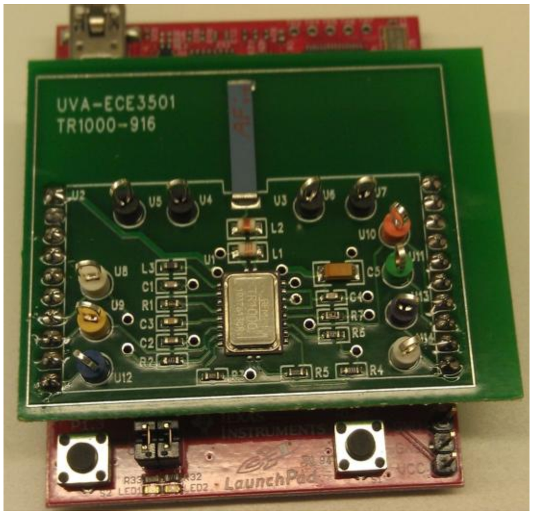

Figure 7 916 MHZ transceiver

Two experiments explore inter-processor communications, one with wires, and the other without. In one experiment we explore communications concepts with Manchester phase encoding. Students are given the receiver code to decipher and then they write the corresponding encoder. The two pieces together achieve simple full-duplex communication. A later experiment uses our custom-designed wireless boards, shown in Figure 7, in lieu of wires. Again the lessons of code reuse are available, while the wireless communication using "On-Off Keying" (OOK) allows students to confront errors and missed signals. This experiment builds upon the state machine architecture developed earlier in the course. Our approach emphasizes systematic methods for turning a state machine sketch into an efficient program, which is vital for achieving a functional project. Additionally, the integrated circuit employed is very simple and the students must actually encode and decode the actual "1's and 0's" of the message at the very lowest level.

For most students, this represents their first exposure to wireless systems and communications concepts. We expose the students to error detection techniques at both the bit and message level, and introduce very rudimentary methods of implementing a messaging protocol. Additionally, although it is rather ad hoc, the students develop a sense of the directionality of the on-board antenna, and gain insight as to how noise affects achievable communications rates. Although they will not have had a formal communications course at this point, knowledge gained from experience with such issues will provide a firm motivation when they encounter them again in a more rigorous and mathematical sense later on.

While this list of experiments has not been exhaustive, they represent a sampling of the size and scope of our projects and give a sense of the pedagogy that we employ to reinforce and motivate material from across the rest of the curriculum.

Example student project

In the Fall 2013 semester, after a sequence of 12 weekly experiments, the final assignment was the student's choice. The requirements for this assignment were that the students use at least 2 concepts from the course in an application of their own choosing and to have some fun. The expectations were that this final assignment would be comparable to the weekly experiments (it was worth a similar number of points) and they had only 2 weeks to complete it. The project was due at the scheduled time for the final exam. These third-year students were also taking other courses with traditional final exams (*Signals & Systems, Electromagnetic Fields,* and *Computer Networks* for example) so our expectations were modest. Several projects were variations on the Simon game[10] using an accelerometer board with a ring of LEDs. Other projects used the rotary encoder board (which has 2 7-segment display digits) for memory games or combined the accelerometer board with the motor controller board, using tilt to define the speed and direction for turning the motor (to which a wheel is attached). Students were asked to prepare video demonstrations of their projects, most of which were rather fun to watch.

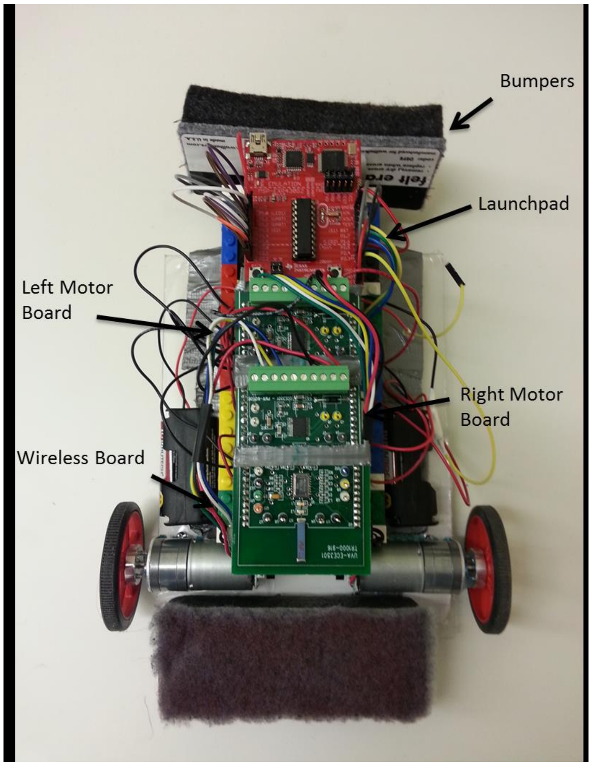

Figure 8 Remotely controlled car

One group produced a remote-control car (see Figure 8**Error! Reference source not found.**) using several header boards and MSP *430 LaunchPads*. This project was particularly challenging since it used the newly-designed wireless boards (which they had not used before). Two motor control boards drive the wheels, the tilt of an accelerometer board determines speed and direction. There is wireless connection (using the new boards) between the controller board and the car.

The five students who developed this project were admittedly well above average. They were a diverse group of students: two electrical engineers, two computer engineers and one computer science major (see). All are 3rd year students. They were self-driven and self-regulating, defining several workable stopping points along the way since a working system was a requirement of the project. They were told that getting something to work was worth more than a more ambitious design that didn't work.

Outcomes

Figure 9 Students Enjoying Chocolate after successful project

Student feedback has been overwhelmingly positive. Students routinely express enthusiasm for the course and the instructors have often heard "I love this course!" Several students have noted how material from the embedded class has helped them on interviews for coveted internships at Google, Microsoft, Intel and Amazon.com, as well as smaller more local companies; indeed, several have obtained employment at Amazon.com and National Instruments. Furthermore informal comments from instructors of the 4th year *Capstone* have been positive. It has been noted that since the inception of this course students are more comfortable with the design process, and more readily integrate the concepts required for completion. The following is the full complete set of comments from the course evaluations in the fall 2013 semester.

Great class! Very technical and hands-on; a welcome break from other courses that were centered around lecture and homework sets.

The workload by the latter half of the semester took a long time, but deadlines were extended when needed, so it was reasonable. I liked how most of the work was hands-on and lab based so we had more experience with practical work.

Embedded computer systems is my favorite class, it has taught me new methods of debugging by using the oscilloscope as a debugging tool. I have learned a great deal in this class about embedded computer systems. Thank you Professor Powell and Dugan!

Using various header boards was an amazing experience. Additionally, having the labs build on each other was very useful and well organized. The only thing I didn't like was having to work with a group that was not so good because many other ones were full. Once I finally got to work with a great group, I started to learn a lot more and really enjoyed the class and projects.

I was pleasantly surprised about the fact that this class is basically ALL CODE. It made me happy, because I'm CPE, but I know that EE majors struggled a bit. You have been warned.

Some of the weekly assignments at the end of the course got to be a little big... I had to put in 20+ hours on 4 or 5 of them. And that's working with a partner, who also put that much time in...

THIS COURSE DEFINES CPE!!!!! HAVE MORE COURSES LIKE IT!!!! It actually separates CPE from CS and EE.

One suggestion, please teach pointers 1/2 way through the semester. I haven't taken 2150 yet *and so my coding skills were not as honed as others. I just learned a ton about code and coding structure from my teammates.*

Probably one of the most enjoyable courses I have taken at UVA.

Embedded is among the best-designed courses I've taken, but I found the effort required to be greater than any of my other courses, including 4-credit courses.

The lecture was helpful in most cases, but too much emphasis was placed on background information as opposed to more technical information to really help students complete the lab.

I learned more in this class than most every other class at UVA. This class teaches a skill that I'm going to use in most projects I do from now on.

Future work

We are currently working on an adapter to allow our custom header boards to also interface with the National Instruments myRIO[11]. We envision that this will facilitate development of follow-on course which would examine embedded computing concepts with a higher level of abstraction using LabVIEW programming[12].

Summary and conclusions

We have outlined our approach to teaching embedded computing, and view this as a first step in a much needed curriculum reform process at our university. We address concerns, not only from embedded computing, but across the spectrum of ECE topics, and in many cases across much of the broader field of engineering.

As the expansion of technical is proceeding at an exponential rate, the demands on the skill set of engineers is also increasing. A modern engineer must be proficient across a range of topics and develop the ability to reason through new types of problems, recognizing the implications and requirements that exist outside of a particular specialty. We envision the approach outlined in this paper as a viable pathway for educating future engineers.

Bibliography

1. Mason GS, Shuman TR, Cook KE. Comparing the Effectiveness of an Inverted Classroom to a Traditional Classroom in an Upper-Division Engineering Course. IEEE Trans Educ. 2013;56(4):430–5.

2. Rodriguez-Andina JJ, Gomes L, Bogosyan S. Current Trends in Industrial Electronics Education. IEEE Trans Ind Electron. 2010;57(10):3245–52.

3. Stapleton WA. Microcomputer Fundamentals for Embedded Systems Education. Frontiers in Education Conference, 36th Annual. 2006. p. 6–10.

4. Harry C. Powell, Joanne Bechta Dugan. Development of Embedded Computing Courseware for Undergraduates in Electrical and Computer Engineering. Proceedings WESE2013 (ACM). Montreal, Canada; 2013.

5. Hsu RC, Liu W-C. Project based learning as a pedagogical tool for embedded system education. 3rd International Conference on Information Technology: Research and Education, 2005 ITRE 2005. 2005. p. 362–6.

6. Radaideh A, Hitt GW, Khalaf K, Balawi S. Engineering Design EDUCATION: When, What, And HOW. [cited 2014 Jan 4]; Available from: http://advances.asee.org/?publication=engineering-design-education-when-what-and-how

7. MSP430 LaunchPad Value Line Development kit - MSP-EXP430G2 - TI Tool Folder [Internet]. [cited 2014 Jan 4]. Available from: http://www.ti.com/tool/msp-exp430g2

8. Code Composer Studio (CCStudio) Integrated Development Environment (IDE) v5 - CCSTUDIO - TI Tool Folder [Internet]. [cited 2014 Jan 4]. Available from: http://www.ti.com/tool/ccstudio

9. Uttarkar NK, Kanchi RR. Design and development of a low-cost embedded system laboratory using TI MSP430F149. 2013 International Conference on Communications and Signal Processing (ICCSP). 2013. p. 165–75.

10. Simon (game) - Wikipedia, the free encyclopedia [Internet]. [cited 2014 Jan 5]. Available from: http://en.wikipedia.org/wiki/Simon_%28game%29

11. NI myRIO - National Instruments [Internet]. [cited 2014 Jan 4]. Available from: http://www.ni.com/myrio/

12. NI LabVIEW - Improving the Productivity of Engineers and Scientists - National Instruments [Internet]. [cited 2014 Jan 4]. Available from: http://www.ni.com/labview/