Adding Hardware Experiments to a First-Year Engineering Computing Course

Dr. Kathleen A. Ossman, University of Cincinnati

Dr. Kathleen A. Ossman is an Associate Professor in the Department of Engineering Education at the University of Cincinnati. She teaches primarily first-year students with a focus on programming and problem solving. Dr. Ossman is interested in active learning, flipped classrooms, and other strategies that help students become self-directed learners.

Dr. Gregory Warren Bucks, University of Cincinnati

Gregory Bucks joined the Department of Engineering Education in 2012. He received his BSEE from the Pennsylvania State University in 2004, his MSECE from Purdue University in 2006, and his PhD in Engineering Education in 2010, also from Purdue University. After completing his PhD, he taught for two years at Ohio Northern University in the Electrical and Computer Engineering and Computer Science department, before making the transition to the University of Cincinnati. He has taught a variety of classes ranging introductory programming and first-year engineering design courses to introductory and advanced courses in electronic circuits. Recently, he has been working to develop graduate level courses in Engineering Education to support current graduate students interested in entering academia. He is a member of ASEE, IEEE, and ACM.

Adding Hardware Experiments to a First-Year Engineering Computing Course

Introduction

At the University of Cincinnati, all first-year engineering students are required to take a two-semester sequence, Engineering Models I and II, which introduces students to the computational software package, MATLAB®, and shows how computing can be used as a tool for solving engineering problems. Course materials are designed to show the importance of computing in engineering and to tie together calculus, chemistry, and physics concepts within an engineering context. The focus of the first course is on developing computing, problem-solving, and logical thinking skills.

As established in the literature, hands-on engineering projects have a positive impact on both student engagement and student learning.[1-2] Several programs have introduced hands-on projects for first and second year engineering students. At the Colorado School of Mines, mechanical engineering sophomores (about 150 split into three sections), have two group projects interfacing software and hardware using the SparkFun RedBoards and MATLAB®.[3] Northern Essex Community College has a first year course offered to a small group of engineering students.[4] Several interesting software/hardware experiments such as control of a stepper motor are performed in a well-equipped lab with oscilloscopes, spectrometers, and cameras. Results of pre and post Self-Assessment indicated that students felt their programming and debugging skills had improved significantly as well as their understanding of electrical components and experimental applications due to the introduction of the hardware component. At Portland State University, Dr. G.W. Recktenwald recently completed a comprehensive six year study of a one year sequence of courses called Living with the Lab (LWTL) offered to first year engineering students interest in Mechanical Engineering.[5] Students felt the course improved their interest in and confidence in their ability to succeed in engineering. Interestingly, although female students indicated less previous experience with hands on experiments, they rated the hands-on work in the sequence higher than their male counterparts did. At Purdue[6], traditional instruction in a first-year engineering course sequence was modified to utilize the graphical programming language LabVIEW as well as the National Instruments myDAQ as a precursor to MATLAB in order to both provide students with a visual representation of the foundational computing concepts and to provide a hands-on component so that students could interact with their programs apart from what was represented strictly within the computer. The course culminated with an open-ended design project requiring the students to use LabVIEW and the myDAQ to create something that was both fun and challenging. Overall, students expressed that the inclusion of LabVIEW and the myDAQ enhanced their experience in the course and prepared them to learn MATLAB in the subsequent semester.

Beginning with the 2013-2014 academic year, two hardware/software labs were introduced in Engineering Models I as a way to increase the hands-on nature of the course. There were several motivating reasons why hardware was chosen for this purpose. First, utilizing hardware in a computing course illustrates how software and hardware can interact to automate the collection and analysis of data or to create an automatic control system. Most complex engineering

systems involve some combination of hardware and software, so this provides students with some basic knowledge of how these types of system work.

Additionally, many students struggle with the abstract nature of programming. Modern learning theories highlight the importance of prior knowledge in the learning process, namely that new knowledge is either assimilated into one's current model of the world or the person's current framework is altered in order to accommodate the new knowledge.[7] In computing, "there are no everyday intellectual activities that can form the basis for spontaneous construction of mental models of programming concepts such as recursion or variables, in contrast to such notions as numbers or velocity."[8] As a result, students often attempt to apply models from natural language or other closely related fields as the terms used in most programming languages come from English.[9-10] However, there are distinct differences between how a computer and a human interpret certain words.[11] As an example, the word while has a very different meaning when used in English versus computing. Both involve the idea of repetition, in that something will be completed until some event occurs. However, in English, once the event occurs, repetition terminates immediately. In computing, all actions being repeated will continue to execute until the condition for the event checked, resulting in a delay between the event and termination of the process. This slight difference can cause significant issues for students. Therefore, it has been argued that students need to be shown explicit models of different constructs in order to develop a firm understanding.[12]

The use of hardware helps to address this issue by providing a more concrete way to experience the execution of code. The hardware aspect provides a physical representation of the actions the computer is performing and can help students develop a better understanding of what is happening.

The focus of the paper will be on the challenges and lessons learned when implementing these experiments with a very large number of students: 1374 students were enrolled in twenty-three sections in the fall semester of 2016. Student feedback on these labs from the end of course survey will be discussed as well as feedback from the undergraduate Teaching Assistants.

Hardware

In the first year, the Digilent Analog Discovery was used as the data acquisition device (DAQ) to interface between MATLAB® and some simple circuitry. There were a lot of connectivity issues with this device and MATLAB®, so it was replaced the following year with the National Instruments' myDAQ. To use the myDAQ, students need to download a large file with drivers and other software then run a setup file which takes some time. Many students have difficulty with this process and are not successful with the installation. Therefore, one of the first lessons learned was to have students come to lab the week before the first hardware experiment with the drivers downloaded and installed so the Teaching Assistants could plug in the myDAQ and check to make sure everything was working properly.

Experiments

Over the past four years, with the exception of switching hardware platforms following the first year, the number and structure of the hardware labs used has remained constant from year to year. However, slight variations have been made to the labs each year. One purpose for these

variations is to discourage students from utilizing code or solutions from previous semesters. More importantly, however, is that this course is taken by all first-year students in the College of Engineering and Applied Science, which encompasses 14 different majors. In order to cater to the wide variety of interests of the students, different labs have been developed, primarily from the standpoint of the problem the students are trying to solve. The hardware itself is fairly similar among the different versions created.

In the first hardware lab (week 4 of the semester), students are introduced to the idea of a hardware/software system and equipped with the basic skills needed to interface their MATLAB® code with the myDAQ. In one of the labs developed for this first experience, students explore using pulse width modulation (PWM) to control the brightness of an LED. They wire a series circuit with an LED and a resistor. In the MATLAB® code, they prompt the user for the desired frequency, duty cycle, and duration of the square wave. They then create the square wave in MATLAB® and apply the square wave as a voltage signal across the circuit. Students first vary frequency and observe the light blinking at the lower frequencies and appearing to remain on at higher frequencies. Students then vary the duty cycle of the square wave and observe the effect on the brightness of the LED.

For the second hardware lab (week 9 of the semester), we have developed three different experiments which are rotated each year. Each experiment involves feedback control within a while loop based on a sensor measurement.

In the Automatic Night Light lab, students control the brightness of an LED based on the ambient light level as measured by a photocell. They wire a resistor in series with an LED and a second resistor in series with a photocell as shown in Figure 1. They first measure the voltage across the resistor in series with the photocell in light and in dark to determine a voltage range for their sensor. In their code, they create a while loop to read the voltage across a resistor in series with the photocell and, based on that reading, adjust the applied voltage to the LED.

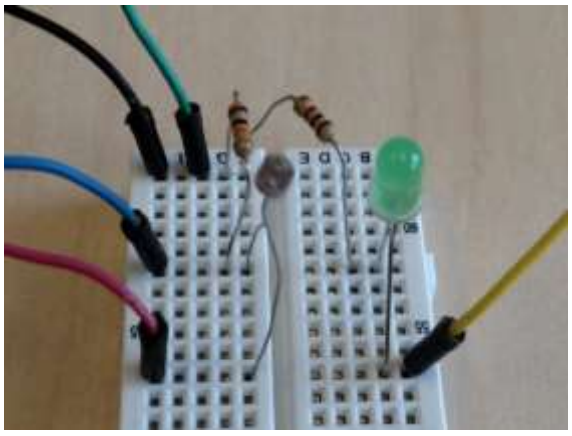

Figure 1: Automatic Night Light

In the Thermostat lab, students create a thermostat of sorts which uses input from a temperature sensor. The schematic is shown in Figure 2. Student first calibrate the temperature sensor (LM335) by using the voltage reading across the sensor at room temperature and the actual temperature in the room (based on the room's thermostat). On the software side, students

prompt the user for a desired temperature and an acceptable range around the desired temperature. The temperature range is converted to an acceptable voltage range for the temperature sensor. The sensor voltage is monitored in a while loop. When the sensor voltage is in range, neither LED is lit. When the sensor voltage is too high, the green LED is lit (air conditioning) and remains lit until the voltage drops down to the desired temperature voltage level. Similarly, when the sensor voltage is too low, the red LED is lit (heater) and remains lit until the voltage increases to the desired temperature voltage level. The temperature sensors were coated so they could be dipped into cold water.

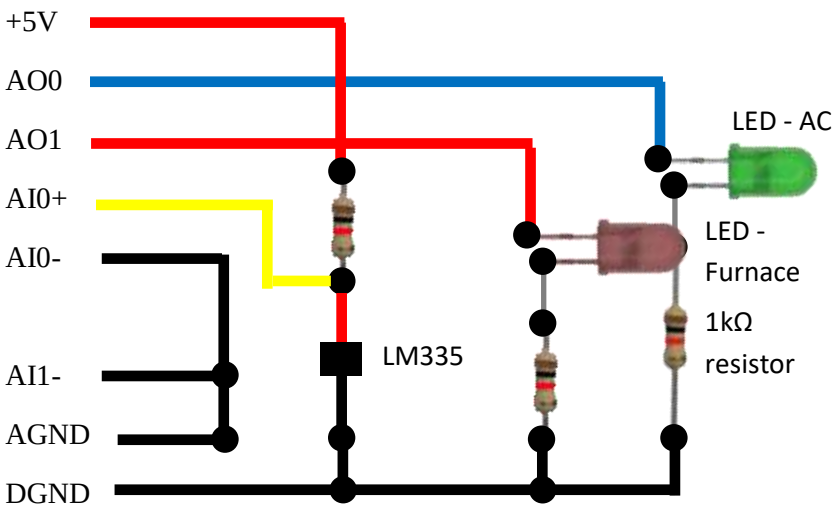

Figure 2: Thermostat Schematic

In the Refrigerator Door Alarm lab, an LED simulating the light in the refrigerator turns on when the door is opened and a buzzer sounds when the door is open too long and goes silent when the door is closed. The schematic is shown in Figure 3. The blue wire represents the door. When it is connected to the 5V line the door is open and when connected to AGND, the door is closed. Students write code to check the voltage level of AI1 every second within a while loop. When it goes high (door open), a command is sent to light the LED and a counter is started. If the door remains open for 10 seconds, a square wave is sent to the buzzer. When the door is closed, the LED is turned off and the buzzer is silenced.

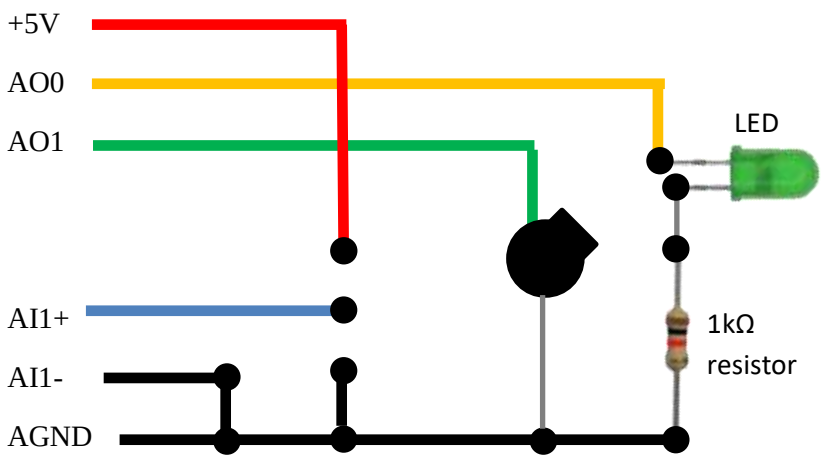

Figure 3: Refrigerator Door Alarm

Challenges and Lessons Learned

In the fall semester 2016, 1409 students were originally enrolled in Engineering Models I and only 35 withdrew during the semester, leaving 1374 students who participated in the hardware/software activities. There were 13 different instructors covering 23 sections of the course with an average of sixty students per section. Students meet for recitation (lab) once per week for two hours and each lab session has six Teaching Assistants (upper level undergraduate students). Obviously, the large number of students taking the course presents a challenge. In addition, some instructors and teaching assistants are unfamiliar with the hardware and with wiring circuits on a breadboard. We also have no dedicated lab space. Recitation is taught in the same two rooms that are used for lecture and for other courses and one of the two rooms is a computer lab.

Keys to running these experiments successfully are:

- Keep the circuit simple
- Develop a good introductory PowerPoint presentation for the instructor
- Set up the experiment a week ahead for instructors and teaching assistants
- Develop a series of verification steps that will allow students to distinguish between hardware and software problems

The majority of our first-year students have no experience with wiring a circuit on a breadboard. It is also apparent that several of our students don't have experience using a screwdriver, which has resulted in set screws being popped out of the myDAQ and often lost. The circuit needs to be simple (few components) and detailed directions and illustrations in the lab document are important. An additional consideration is that students need to be able to complete the lab within the two hour time period since they cannot take the components or the myDAQ home with them.

Instructors need to be provided with material that will allow them to explain the concepts behind the lab. They also need to be encouraged to actually present this material to their students rather than just assuming students will go through the presentation on their own.

The experiments are set up a week ahead so that instructors and teaching assistants that are not familiar with the equipment can see the set-up and run the code in order to understand how the lab is supposed to work. Some instructors and teaching assistants take advantage of this but some go to lab unprepared.

In a system that involves both hardware and software, it is often difficult to determine whether the problem is with the software or with the hardware. In these labs, students always begin by checking connectivity between the myDAQ and MATLAB®. Once connectivity has been established, they wire the circuit. After wiring the circuit (before writing any code of their own), they send signals to the circuit and make measurements using commands in the MATLAB® command window. If an LED doesn't light or they don't get a voltage reading from a temperature sensor or photocell, then they know they have a wiring problem that needs to be addressed. Once the wiring is verified, then students write the software.

Student Feedback

Three Likert-scale questions about the hardware experiments were included on the end of course survey followed by a short essay question:

1. The lab assignments which used the myDAQ connected to a circuit improved my understanding of how a software program can be used to control a hardware device.
2. Lab #4 - creating a square wave with various duty cycles and frequencies to light an LED improved my understanding of the frequency and period of waveforms such as square waves and sine waves.
3. Lab #9 - controlling the light level of an LED using a photocell to detect light level improved my understanding of conditional statements and while loops.
4. Please comment on what you liked and/or did not like about the hardware labs with the myDAQs.

The survey had a response rate of 78.8%. Student responses to the first three questions are shown in Figure 4. About 75% of the students strongly agreed or agreed that the experiments improved understanding of programming concepts and how software can be used to control hardware. Roughly 8.5% of students strongly disagreed or disagreed with the statements. Also, the second experiment was perceived as improving understanding a bit more that the first which may be because students already had some experience with the myDAQ and wiring on a breadboard for the second experiment.

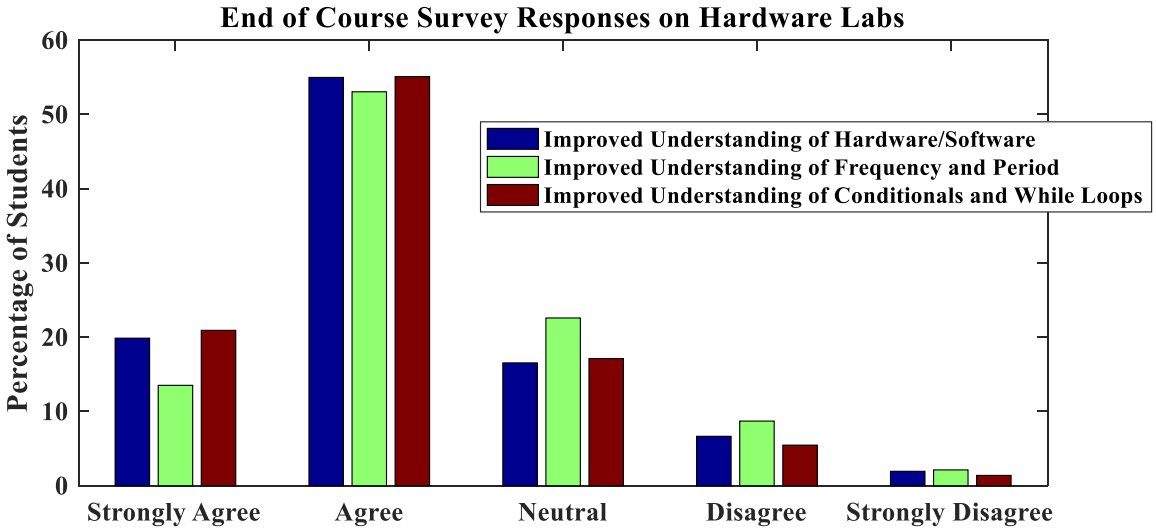

Figure 4: End of Course Survey Responses

When asked to comment on the hardware labs, 59% of students had positive comments, 22% had negative comments, and the remaining 19% had either neutral or mixed (some good and some bad) comments.

The vast majority of the positive comments were about actually getting to see something happen as a result of a software programs. Several students also commented that the hardware labs were easier to complete within the two hours than the software labs and working in teams was enjoyable. Samples of positive student comments:

- *I enjoyed the hardware labs. I think it is important to see the physical connection that software can have with hardware, and the labs provided this experience. They were very relevant to my chosen major, Electrical Engineering.*
- *Those labs were my favorite because i got to see physical results and see what applications MATLAB® could be used for. Plus it helped me further understand what i was learning in engineering foundations.*
- *The concept of connecting software and hardware was extremely helpful for someone like me with very little coding experience. It really helped to connect some influential points.*
- *I thought these labs were very interesting and helpful at giving me a better understanding of how hardware and software come together.*
- *I thought it was cool to work in teams. It would be better for the class, possibly, if there were more group assignments.*
- *I had always wondered how coding works for hardware, so it was nice to be able to manipulate it ourselves and see it in action.*
- *I liked them! Cool way to show how computers control stuff*
- *I liked the MyDAQ labs since they are hands on and helped me learn better since I am a more hands on person. There should be more experiments like these ones. If all experiments were like this it would make a lot more sense to me.*
- *These were honestly some of my favorite labs on account of us being able to see how a code we write doesn't have to be contained to a computer.*

The negative comments were about the difficulty of downloading the software, difficulty of connecting to the device, not understanding what the provided code was doing, not interested in electronics, working in teams, and not challenging enough. Several students with MAC computers were really unhappy that they couldn't use their computers for the hardware labs because the MathWorks Support Package for the NI myDAQ does not work for MACs. Samples of negative student comments:

- *The MyDAQ software was just a very confusing process to download, install, and operate. It was hard to understand why the software didn't work after the installation was done. The whole process was just not very user friendly.*
- *The software needed to run the hardware wouldn't download properly for many people, so if there was a direct download link it might help.*
- *IT does help with seeing real life application a bit, but when you don't know what most of the pre-written code is doing, it makes understanding what's going on diffficult.*
- *I felt like the MyDAQ labs were not very effective, most of the code was given to us, and so I did not understand what the code was doing. I learned a little about the topics they were meant to teach about, but that was from reading the lab instructions, not from actually doing the lab.*
- *I found the MyDAQ mostly uninterested because I don't really plan on using anything like that in my discipline of engineering. I felt as though it was geared more towards electrical engineers and computer science/engineering majors.*
- *I didn't really like using a breadboard to make a circuit. Part of that was due to my limited understanding of circuits and how breadboards work.*
- *The hardware labs were difficult because they required teamwork which is hard to do with code. The number of hardware sets available made it required to work in a group but it would have improved learning if the labs were indiviudal.*
- *I liked to be able to see the connection between a software program and a hardware device, but since only one computer could be used and multiple to people were working on the DAQ device at once, it was difficult to understand fully what was happening.*
- *I felt that doing the assignment with a group would often result in only one person coding, another person setting up the hardware, and the last one would just sit there and look pretty.*

Students work in teams of three for the hardware labs but complete the software labs individually. Of the open-ended responses, 68 of them addressed the teamwork aspect of the lab activities. The responses about working on teams were mixed. 36 of the 68 indicated that they liked working in teams because it was fun and helped them to understand and complete the lab. 32 of the 68 indicated that they did not like working in teams. The majority of these students indicated that the team size should be reduced from three students to two so that everyone in the group could participate. A few students indicated that they would prefer to work alone.

Teaching Assistant Feedback
An e-mail was sent to the teaching assistants with two questions:

1. Do you feel that the hardware experiments were worthwhile doing in Models I? Why or why not?
2. What were the challenges as a T.A. in helping students with these labs?

Nine of the 45 teaching assistants from the fall 2016 semester responded. Seven thought that the hardware experiments were worth doing and two felt that they were not. Some of the teaching assistants commented on how the students enjoyed seeing the physical results of their code while others commented on the importance of the experiments based on their own co-op experiences. Teaching assistants that were not enthralled with the experiments felt that they were applicable for only a small group of engineers and much too hard to debug. A sample of the responses:

- *I felt that the hardware experiments were very worthwhile. Immediately when you mentioned the hardware labs I thought of an experience I had that was unique to those labs: when I was helping a group of students in my section this fall, they were having trouble with the code for a while, but when they finally got it, you could tell how excited they were by the look on their faces. You could tell that they were impressed that they could use a computer to make something physical do something, rather than just making a computer do something on the computer, if that makes sense. And of course they were excited that they finally solved their problem, but it seemed like they were even more excited than normal when they produced something they could touch and physically interact with. In other words, I felt that the hardware labs were especially useful because students can feel like they are creating something tangible, which provides an additional platform for students to experience engineering and learning. Different students learn in different ways and are interested in different topics, and I believe the hardware experiments add an important type of variety to the course's assignments.*
- *I do feel that the hardware experiments were worth doing in Models 1, because I think it bridged the gap for many students between learning to write code and how that code could actually apply to the real world. Many of the students that I worked with started to understand how important coding is and how it can be used as a tool after utilizing it to manipulate the LED with the circuit board and MyDAQ. I would say it really brought home simple things like that code runs the smartphones in their pockets.*
- *I think the hardware labs were beneficial. When I took Basic Electric Circuits, we also worked on boards from Texas Instruments. The brief exposure I had during my Models 1 labs made me confident working with the more complicated NI Elvis boards. I've also used some of what I learned about MATLAB hardware programming in a Neurology lab research project I am working on at Cincinnati Children's Hospital. From only a few weeks on co-op at BASF, I already see how complex the interactions between software and hardware can be in an industrial setting. While the software is not the same, the labs in Models 1 at least plant the idea of how software and hardware must work in unison.*
- *I think that the hardware experiments are definitely worthwhile. They are a bit clunky at first getting laptops set up with the drivers and getting the DAQs to actually connect to MATLAB, but I do believe that this is a great introduction to the NI Devices and how they*

work. I am currently co-oping at Northrop Grumman in West Chester, and I use NI-DAQs with MATLAB on a weekly basis for various types of hardware testing (I actually have about five of them scattered around my cube right now). Although these labs will not be applicable in the future for every kid that goes through this class, I strongly believe that showing kids how easy it can be to have software and hardware working together is definitely a worthwhile learning experience, and I can personally say I felt much more confident in my abilities as an engineer at work because of my early exposure to the DAQs in your labs.

- *I feel that the hardware experiments were not very worthwhile to the majority of students, at least compared to the non-hardware assignments. The skills used for the hardware assignments are quite specific, and I don't think many students would find them useful later on, unless they are electrical engineers or something. The coding assignments, however, are much more worthwhile as they exercise a skill that is very flexible and used in many fields, to some extent.*
- *I don't feel the hardware experiment is worth doing. I don't think it prepares them for anything. Its awkward and frustrating for students and TAs. The little bit of code they actually write is not worth the time spent working with the hardware.*

When asked about the challenges as a T.A. in helping students, most all cited the frustration in installing the software properly and the difficulty in finding wiring errors.

Future Plans

There are two sections of Engineering Models I running during the Spring 2017 semester. In one section, the NI myDAQ is being replaced by a system utilizing an Arduino Uno while the other section will not use hardware. A lab activity has been developed that requires the students to develop a coding solution to a problem, with one version utilizing the Arduino hardware and a simple circuit and the other version focusing solely on a simulation. Students will be polled before and after the activity to determine their level of understanding of the concepts covered by the activity and their experience in solving the problem. The goal is to determine whether the use of the hands-on hardware component has an educational benefit in the learning of computing concepts and what those benefits might be.

The motivation for looking at the Arduino is that it could eliminate some of the negatives that students and teaching assistants associate with the hardware experiments. The software to communicate with the Arduino is extremely easy to install because it is an Add On in MATLAB®. The code to set up the input and output channels is simpler to understand so students would hopefully not be frustrated by a bunch of pre-written code that seemed mysterious. The wiring would be simpler since students would not have to use a screwdriver to secure wires into a terminal. The Arduino is significantly cheaper than the myDAQ so it would be possible to purchase additional units, allowing students to work in groups of two rather than three.

Bibliography

1. M. J. Prince, “Does active learning work? A review of the research,” Journal of Engineering Education, 93: 223–231, 2004.
2. S. Freeman, S. L. Eddy, M. McDonough, M. K. Smith, N. Okoroafor, H. Jordt, and M. P. Wenderoth, “Active learning increases student performance in science, engineering, and mathematics,” Proceedings of the National Academy of Sciences of the United States, 111(23):8410–8415, May 2014. doi: 10.1073/pnas.1319030111.
3. D. Derrick Rodriguez, J. Blacklock, and J. M. Bach, “Letting students learn through making mistakes: Teaching hardware and software early in an academic career,” Proceedings of the ASEE Conference, Seattle, WA (2015).
4. S. W. McKnight, J. E. Pelletier, and P. G. Leventman, “Introduction to Engineering Course At a Community College Using Hands-On MATLAB Experiment Control,” Proceedings of the ASEE Conference, San Antonio, TX (2012).
5. G. W. Recktenwald, “Six Year of Living with the Lab,” Proceedings of the ASEE Conference, New Orleans, LA (2016).
6. G. W. Bucks and W. C. Oakes, “Enhancing the Experience in a First-Year Engineering Course Through the Incorporation of Graphical Programming and Data Acquisition Technology,” Proceedings of the ASEE Conference, San Antonio, TX (2012).
7. McLeod, S. A. Jean Piaget. Retrieved from www.simplypsychology.org/piaget.html (2015).
8. Rogalski, J. and R. Samurcay. Acquisition of programming knowledge and skills. Psychology of Programming. J. M. Hoc, T. R. G. Green, R. Samurcay and D. J. Gilmore. San Diego, CA, Academic Press: 157-174 (1990).
9. Ma, L., J. D. Ferguson, et al. "Investigating the viability of mental models held by novice programmers." SIGCSE Bull. 39(1): 499-503 (2007).
10. Ma, L., J. D. Ferguson, et al. Using cognitive conflict and visualisation to improve mental models held by novice programmers. Proceedings of the 39th SIGCSE technical symposium on Computer science education. Portland, OR, USA, ACM (2008).
11. Soloway, E., J. Bonar, et al. "Cognitive strategies and looping constructs: an empirical study." Commun. ACM 26(11): 853-860 (1983).
12. Bayman, P. and R. E. Mayer. "A diagnosis of beginning programmers' misconceptions of BASIC programming statements." Communications of the ACM 26(9): 677-679 (1983).