Paper ID #18439

Introducing Coding in Freshman Physics Laboratories using Arduinos

Dr. Carl K Frederickson, University of Central Arkansas

Dr. Frederickson has taught physics at UCA for 22 years. He is the current department chair and is leading the development of a new Engineering Physics degree program.

Introducing Coding in Freshman Physics Laboratories using Arduinos

Abstract

During the fall semester 2015 Arduino microprocessors were introduced into the second semester calculus based physics laboratory. The Arduino was used as the data acquisition platform. One reason for this change is to introduce coding into the lab activities as part of the data collection/analysis process. Lab activities are designed to be more open-ended and require students to determine what data needs to be taken and how it needs to be analyzed. The focus is to have students design and carry out measurements with a goal in mind. This change required time to be spent learning the Arduino hardware and software development environment. This measure has now been used in both the first and second semester sections of the calculus-based course. During the first semester, which focuses on mechanics, the Arduinos were used in conjunction with Vernier Software data collection hardware. This allowed more of a focus on the coding of the microprocessor. The majority of students have no coding experience prior to this course. Students do not seem to have an issue with coding the Arduinos. There were a few instances where the Arduino proved to not be the best platform. Student example work and attitudes will be presented. The effect on student assessment performance will be discussed.

1 Introduction

The ability to read, create, and interpret computer code is an important skill for an engineer or a physics major to have. Many programs, including the program at the University of Central Arkansas (UCA), require students to complete one or more computer science courses for their degree. While these courses provide a solid introduction to programming in general, they do not always provide broad opportunities to apply this knowledge in an engineering or physics environment. This is similar to the required writing courses that students complete as part of their freshman general education core. While these courses teach the mechanics of writing, it is in a broad spectrum and not always applicable directly to writing in a technical field.

Students in the traditional physics and applied physics tracks at UCA are required to complete a 3-hour introduction to programming course offered in the Department of Computer Science. This course currently uses Python as the programming language of choice. Students are required to develop code for an end-of-semester project of their choosing. While the course has many physics and pre-engineering students, it will also have many management information systems majors as well. As with freshman writing courses it has to present coding to a broad audience. Thus students do not always see the connection to their chosen field of study.

The Department of Physics and Astronomy is introducing more coding across its curriculum to better prepare our students for careers and graduate study. The focus of many of these efforts in our department and around the country is on adding modeling calculations to courses[1,2,3]. As a new component in this effort, a pilot program was started in the fall semester of 2015. During this semester Arduinos were introduced in the laboratory component of the second semester calculus-based freshman physics course. They were used primarily for data collection instead of Vernier LabPro interfaces. One goal of this change was to provide a platform to introduce coding as a component of the collection and analysis in laboratory activities.

The project has now been conducted for three semesters in the off sequence calculus based course; twice in the second semester and once in the first semester. Arduinos are currently being used for the second time in the first semester course. This paper will focus on the group of students that used the Arduinos for both semesters. In this class 27 of the 37 students completed both the first and second semesters in the spring and fall of 2016. An additional, 5 students completed the second semester only, in the fall of 2016. Both the Force Concept Inventory (FCI), during spring 2016, and the Brief Electricity and Magnetism Assessment (BEMA), during fall 2016, pre and post-tests were used to measure student knowledge gain. The initial goal is to introduce the coding skills. However, this does require spending part of laboratory time on the coding as opposed to the physics. The FCI and BEMA are being used to make sure this time effort does not hurt student mastery of the physics.

This paper will first discuss the choice of platform. It will then present 4 of the projects that students are asked to complete. Two projects will be presented from the first semester and two from the second. During the first semester the Arduino serves as the link between Vernier equipment and the computer. Circuits are introduced at the beginning of the second semester allowing more flexibility in the projects. The results of a students survey presented as part of the end of semester course evaluation during the second semester will be discussed.

2 Choice of Platform

The goal of this project is to provide opportunities for students to design code for data collection and analysis. To facilitate this, one aspect of the hardware chosen is that it require the least amount of ”circuit building” to accomplish the data acquisition. Two platforms were considered, the Arduino Uno and the Raspberry Pi R3. Both of these platforms are inexpensive and readily available. The Raspberry Pi has the advantage of providing a stand-alone computer interface. The Pi also uses Python as its standard programming language that is more in line with coding efforts being implemented in the curriculum. However, the Pi does not include analog input channels. External circuitry would be required to record analog signals making it less useful for data acquisition.

The Arduino Uno includes 14 digital I/O channels and 6 analog input channels making it more useful as a data acquisition platform. Secondly, students are able to connect currently available data acquisition equipment using either an analog or a digital breadboard adapter sold by Vernier.

This allows students to use a Vernier Force Sensor, Motion Detector, or any other equipment. Using the adapters, students are not required to build any circuits during the first semester course which focuses on mechanics. Data collection can be controlled using the Arduino and this hardware with a simple 4-wire connection between the digital or analog adapters and the Arduino. Two of these wires are the 5V and Ground connections which students do not have any issues understanding.

There are two drawbacks to the Arduino. First, it does not use Python as the default programming language. This does not reinforce the use of Python in latter classes. The second is that the analog inputs only accept voltages from 0V to +5V. They do not read negative voltages. This is not really an issue until late in the second semester when discussing oscillatory circuits. Both of these issues keep the Arduino from being ideal but are out weighed by its other attributes.

A library of starter code for controlling different Vernier data acquisition equipment is also available[4]. This code can be used to demonstrate how the equipment can be controlled and used for data acquisition. It provides a starting point for students to begin their experimental design. The ability to measure analog signals without external circuitry and the availability of the library of starter code associated with currently held Vernier data acquisition equipment made the Arduino the preferred platform.

3 Overview of Laboratory Activities

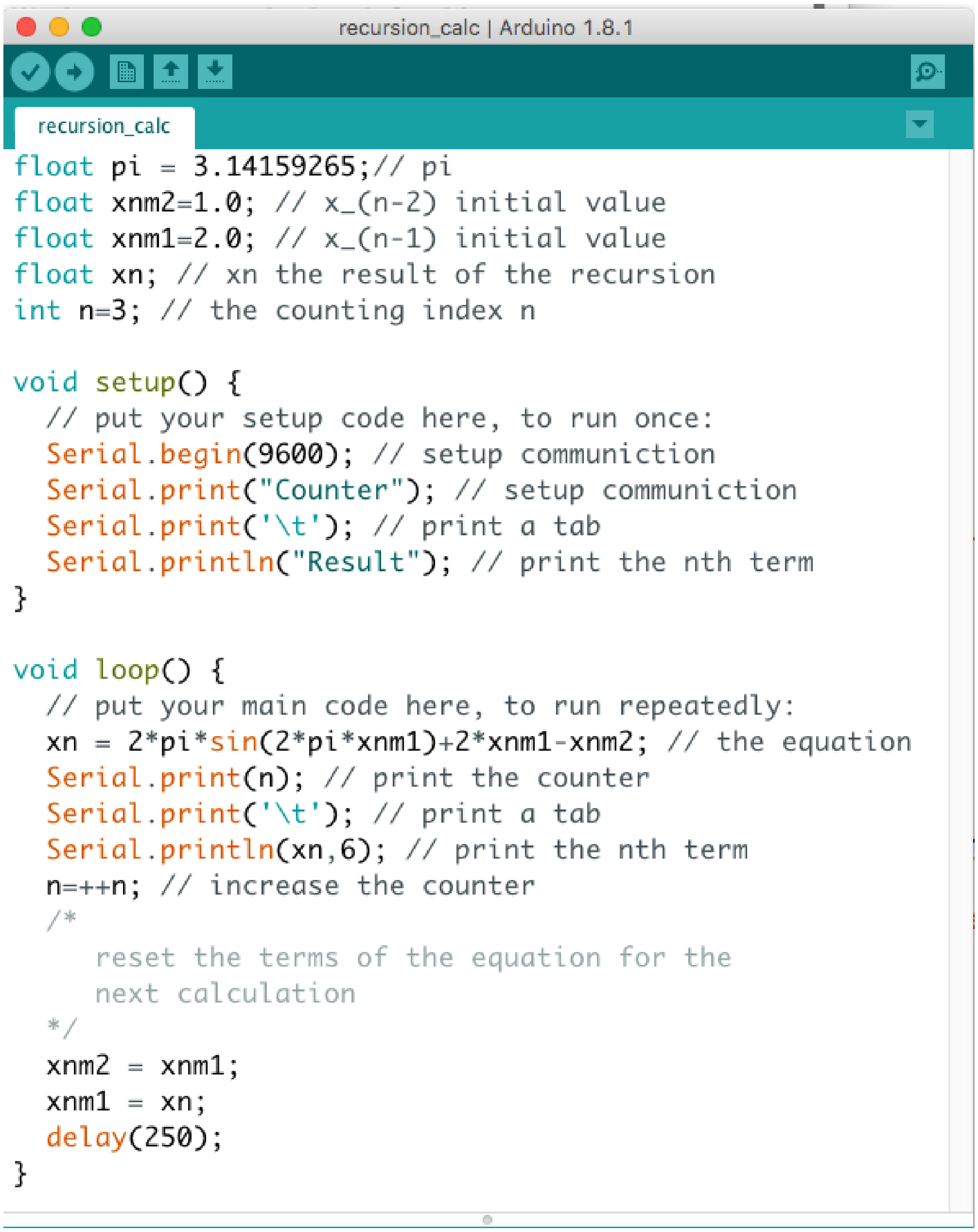
Use of the Arduino platform is designed to introduce students to some of the behind the scenes activities that occur when they make a measurement. It is important to battle the ”that’s what the computer said” attitude of many students. To this end, the activities are left as open-ended questions. Students are given instruction on how the data acquisition equipment works, sample code for controlling the equipment, and then asked to complete a task. No prescribed procedure is used. Students are asked to design their own data acquisitions procedures and are required to defend them. This provides an environment where the requisite physics concept can be reinforced while allowing the student to be more self-directed[5].

3.1 First Coding Activity

The initial activity does not involve any data collection. It is designed to provide students with a healthy skepticism of the results that the Arduino (or any other computer) produces. Students are given Eq. (1) and asked to evaluate by hand (pencil and paper) terms 3 - 15 starting with x_1 = 1 and x_2 = 2.

$$x_i = 2\pi \sin(2\pi x_{i-1}) + 2x_{i-1} - x_{i-2} \qquad (1)$$

This equation evaluates to give the integers one after another (x_1=1, x_2=2, x_3=3, x_4=4, ...). The $sin(2\pi x_i)$ term evaluates to zero because every x_i is an integer. After determining the first 15 terms in the recursion by hand, students are asked to code this equation in the Arduino. Figure (1) shows an example sketch for this calculation. The results and code are shown in Fig. (2). Note that the Arduino is only able to complete this calculation accurately through the 6th term due to round off errors in the sin term. The results are no better when you use Excel or Google Sheets. This is often the first time that students have seen a calculation like this done on a computer. It offers an opportunity to generate some understanding of the care that needs to be used in computer calculations.

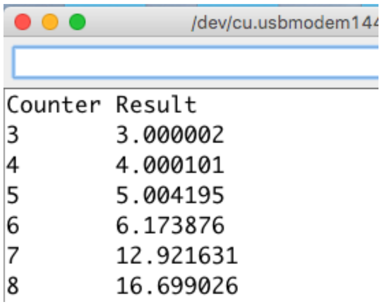


```
float pi = 3.14159265;// pi
float xnm2=1.0; // x_(n-2) initial value
float xnm1=2.0; // x_(n-1) initial value
float xn; // xn the result of the recursion
int n=3; // the counting index n

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600); // setup communiction
  Serial.print("Counter"); // setup communiction
  Serial.print('\t'); // print a tab
  Serial.println("Result"); // print the nth term
}

void loop() {
  // put your main code here, to run repeatedly:
  xn = 2*pi*sin(2*pi*xnm1)+2*xnm1-xnm2; // the equation
  Serial.print(n); // print the counter
  Serial.print('\t'); // print a tab
  Serial.println(xn,6); // print the nth term
  n=++n; // increase the counter
  /*
     reset the terms of the equation for the
     next calculation
  */
  xnm2 = xnm1;
  xnm1 = xn;
  delay(250);
}
```

Figure 1: A sketch that calculates the recursion equation.



```
Counter Result
3       3.000002
4       4.000101
5       5.004195
6       6.173876
7       12.921631
8       16.699026
```

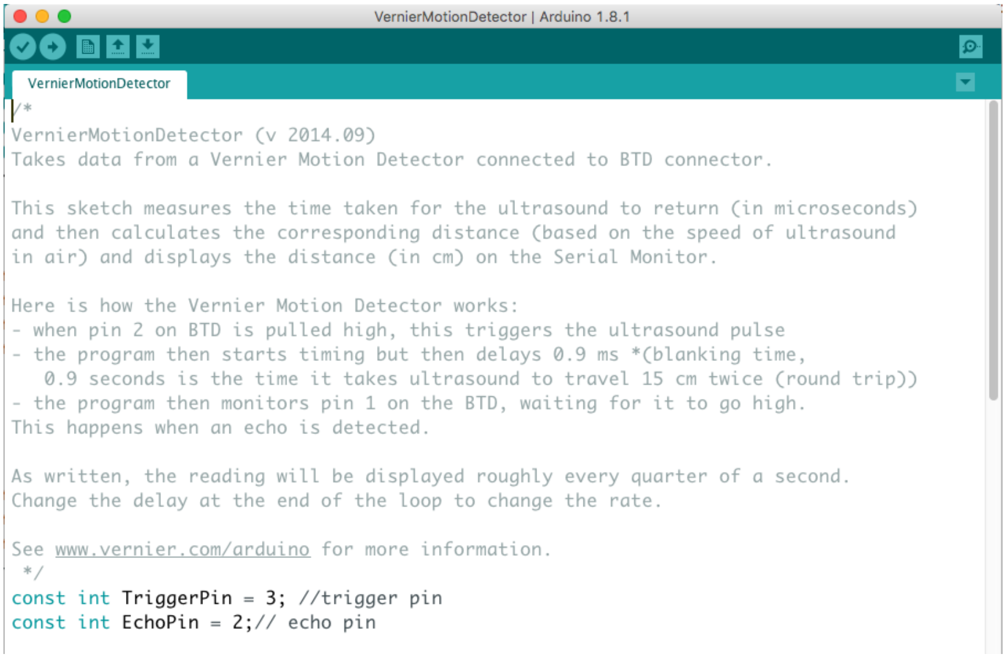
Figure 2: Results of the recursion calculation using the Arduino.

3.2 Second Coding Activity

The second activity is a familiar laboratory exercise. Students measure linear motion using a sonic motion detector to collect position vs. time data. The data is used to determine the speed of the cart from a graph of the position vs. time. Most students will have used computer data acquisition techniques before and are very comfortable with accepting the computer results. Many of them have used similar systems in high school courses and other science laboratories in conjunction with software that reports the data in a convenient format. The data output by these systems is often pre-scaled and processed to provide for quick analysis. A good example is the data that is automatically displayed by the LoggerPro software from Vernier when a sonic motion detector is connected. The software automatically displays position, velocity, and acceleration vs. time graphs. Using the Arduino, students are required to develop all of this data analysis on their own. The use of the Arduino requires students to invest more into the details of the measurement[6].

Using a BTD adapter available from Vernier, students can easily interface the motion detector with the Arduino. A starter sketch is available from Vernier that allows students to quickly access the motion detector signal. Figure (3) shows the header comments from the starter sketch that explain how the motion detector should be connected and how the sketch works. The sketch illustrates the usefulness of defining variables using names associated with their function. The Arduino digital IO pins used to trigger the detector and listen for the echo indicator are defined using the ”TriggerPin” and ”EchoPin” variables.

As a first check students will move the cart on the track to confirm that the position data reported is accurate. Once they are comfortable that the position data is correct they are asked to use the code to collect position and time data for a cart moving on a horizontal track. The carts are fairly low friction and students expect to find a linear relationship between the position and time. This project has been run with 4 laboratory sections (20 different lab groups of 3-4 students) over two spring semesters. Each year one lab group has used the 0.25 s delay between acquisitions as the Δt to build their position vs. time graphs. This produces very good results for a cart that is given a speed of around 0.5 m/s. The rest of the lab groups have relied on the default time variable, *Duration*, output by the starter sketch, see Fig. (4). This output produces a plot of position vs. time that indicates that the cart was traveling at a speed of 0.017 cm/μs. Most groups will use the raw data as reported in centimeters and microseconds. When asked, students are satisfied with the small number that they determine for the speed. It is considered as a reasonable result until it is



```
/*
VernierMotionDetector (v 2014.09)
Takes data from a Vernier Motion Detector connected to BTD connector.

This sketch measures the time taken for the ultrasound to return (in microseconds)
and then calculates the corresponding distance (based on the speed of ultrasound
in air) and displays the distance (in cm) on the Serial Monitor.

Here is how the Vernier Motion Detector works:
- when pin 2 on BTD is pulled high, this triggers the ultrasound pulse
- the program then starts timing but then delays 0.9 ms *(blanking time,
   0.9 seconds is the time it takes ultrasound to travel 15 cm twice (round trip))
- the program then monitors pin 1 on the BTD, waiting for it to go high.
This happens when an echo is detected.

As written, the reading will be displayed roughly every quarter of a second.
Change the delay at the end of the loop to change the rate.

See www.vernier.com/arduino for more information.
 */
const int TriggerPin = 3; //trigger pin
const int EchoPin = 2;// echo pin
```

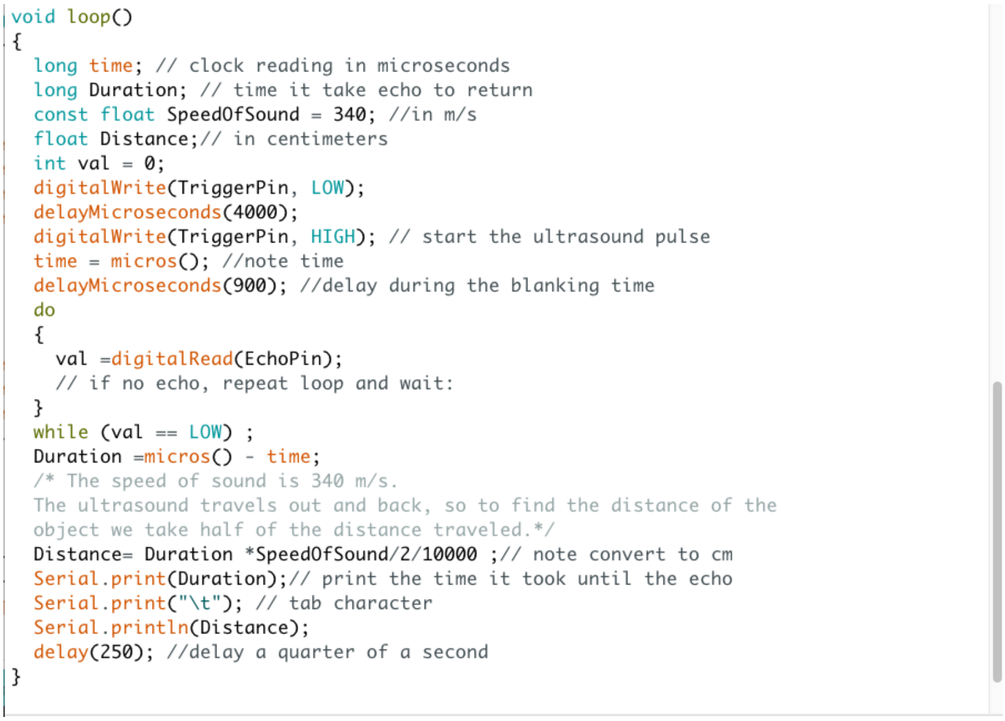
Figure 3: The header comment to the starter sketch controlling the Vernier Motion Detector.

converted to 170 m/s, or half the speed of sound used in the sketch. The fact the 90% of lab groups report this result without reservation initially reflects student willingness to accept the results that the computer reports with little critical evaluation. This leads to a discussion about how to modify the program to report the time that the cart is at each position and not the time it takes the sound to travel to the cart and back. This is the first modification that is associated with a physical measurement.

This exercise offers a number of opportunities for students to begin to understand the connection between physical variables and the code that controls the data acquisition and reporting. Students are also cautioned not to accept the output of any sketch without some form of verification similar to the position data confirmation mentioned earlier. Observing the lab groups, most realize that the code, in its original state, outputs the variable *Duration* in μs and not in s as stated in the output column header. The lab groups respond to this in different ways. Some will adjust the code to account for this discrepancy. Few are comfortable enough changing the code even in this small way at this early stage and will need instruction. Most will leave the output in μs leading to reports of speeds such as 0.017 cm/μs. A classroom discussion is used to identify ways to alter the code to output the time the cart is at each position as opposed to the time it takes sound to travel to the cart and back. This involves the correlating the physical events and the code.

3.3 Third Coding Activity

The first activity of the second semester course is to measure the internal resistance of a D-cell battery. DC circuits will have been discussed in lecture already so students are familiar with voltage drops across resistors. It is not uncommon for students to look to the internet for help



```
void loop()
{
  long time; // clock reading in microseconds
  long Duration; // time it take echo to return
  const float SpeedOfSound = 340; //in m/s
  float Distance;// in centimeters
  int val = 0;
  digitalWrite(TriggerPin, LOW);
  delayMicroseconds(4000);
  digitalWrite(TriggerPin, HIGH); // start the ultrasound pulse
  time = micros(); //note time
  delayMicroseconds(900); //delay during the blanking time
  do
  {
    val =digitalRead(EchoPin);
    // if no echo, repeat loop and wait:
  }
  while (val == LOW) ;
  Duration =micros() - time;
  /* The speed of sound is 340 m/s.
  The ultrasound travels out and back, so to find the distance of the
  object we take half of the distance traveled.*/
  Distance= Duration *SpeedOfSound/2/10000 ;// note convert to cm
  Serial.print(Duration);// print the time it took until the echo
  Serial.print("\t"); // tab character
  Serial.println(Distance);
  delay(250); //delay a quarter of a second
}
```

Figure 4: The loop function of the starter sketch controlling the Vernier Motion Detector.

with this problem. By this time, they know how to measure voltages using the Arduino's analog inputs. Students are left to build their own sketch to complete this task. This is the first sketch that they must build from scratch themselves. There is a sketch available to measure voltage but not to complete the calculations required to determine the internal resistance.

3.4 Fourth Coding Activity

The fourth activity is to use the Arduino to measure the capacitance of an unknown capacitor. RC time constants will have been discussed in class at this point in the semester. Most of the groups used a measurement of the charging time for an RC circuit to complete this project. Students would measure voltage as a function of time then export their data for analysis in Excel. A few groups searched and found code online that can be used to have the Arduino determine the capacitance from the charge time of the capacitor. The code and an explanation can be found at the Arduino website[7]. Students can download a copy of the sketch provided and quickly read out the value of their capacitor. To counter this availability, student were required to present their code to the class along with a detailed explanation of how it worked. Secondly, they were asked to measure both the charge and discharge time constants to determine the capacitance.

	Freshman	Sophomore	Junior	Senior	
Class Rank	0	4	8	20	
	Computer Science	Mathematics	Physics	Chemistry	Health Sciences
Major	4	3	8	16	1

Table 1: Class Makeup, Fall Semester 2016

4 Student Demographic

Arduinos have been used for three semesters starting with the fall of 2015 in a second semester course. The spring-fall sequence of freshman physics sections are smaller than the sections that run fall-spring. The group of students in the spring 2016 and fall 2016 courses used the Arduino labs in both semesters. These students were surveyed during the fall 2016 semester on their opinions of these activities. The makeup of this class is given in the Table (1).

Students in the fall class were surveyed about their experience programming and working with Arduinos prior to this class. They were then asked about how confident they were that they could build a sketch (develop Arduino code) to collect data on their own. Less than half (42%) of the class claimed any prior experience programming. None of the students had any experience using Arduinos prior to this course. Student confidence coding the Arduinos is shown in Table (2). Over 60% of the class felt they could program the Arduino to measure a single voltage or voltage as a function of time. A concern is that students would not respond well to having to ”code” in their physics class. This is especially concerning as the majority of these students are chemistry majors. Students were asked about this. The results are shown in Table (3). Just over half of the class felt the Arduinos made lab harder while nearly 30% felt it made lab easier. An interesting result of the survey is the number of students who felt using the Arduinos allowed them to be more creative in lab. The activities being developed for the Arduinos are being designed as open questions for the students to answer using measurement and analysis. There are no prescribed procedures which may have led to students feeling more creative in their lab activities. A large portion of the class (71%) felt that using the Arduinos required them to think more about the measurements. This is

	Very confident	Somewhat confident	Not that confident	I don’t believe that I could
How confident are you that you could generate a small program to measure a single voltage using the Arduino?	10%	55%	32%	3%
How confident are you that you could generate a short program to measure voltages as a function of time using the Arduino?	10%	52%	35%	3%

Table 2: Student Opinions on Using Arduinos

Using the Arduino to take data:	
Required me to think more about what was being measured.	71%
Did not require me to think about the measurement more than normal.	13%
Confused me about what was being measured.	16%
Using the Arduino to collect data has:	
Made lab more difficult	52%
Not affected how hard lab is	19%
Made lab easier	29%
Using Arduinos in lab:	
has allowed me to be more creative during lab	45%
is no different than using other data acquisition systems	23%
has made lab more difficult	32%

Table 3: Student Confidence Coding Arduinos

one of the reasons Arduinos were introduced. During the first semester one of the students in this class when asked about the axes on a graph they had produced commented ”LoggerPro always does that for me.” One advantage of the Arduino labs is that students are required to consider more carefully what they are measuring and how the data should be displayed for analysis.

5 Conclusion

The main effort of this project is to introduce students to coding in scientific activities. This increases the time required for laboratory activities limiting the number of labs that are completed. There is concern that this could affect student mastery of the basic physics concepts. To monitor this, students in the spring 2016 (University Physics 1) and fall 2016 (University Physics 2) courses completed pre and post-tests using the FCI[8] and BEMA[9] exams respectively. The PhysPort Data Explorer[10] web page was used for analysis of both assessments. The average gain on the FCI for this spring class was 0.36 (N = 24) which is described as ”about average for interactive engagement class[10].” This is based on a small number (24) of students who completed the pre and post-test. It is an indicator that using the Arduino in lab is not a detriment. Student grades in the first semester were similar to previous semesters. Results for the second semester were not as encouraging. Students grades were higher on average than in previous years. However, the normalized gain on the BEMA pre and post-tests were very low. The average normalized gain was 0.16 (N=28) which is described as ”near the bottom range of traditional lecture classes[10].” The student performance in the class did not correlate well with their performance on this assessment. Both of these measures involve small numbers of students (24 in spring 2016 and 28 in fall 2016). More data will be needed to determine if the use of the Arduino in labs is a negative, neutral, or positive factor in student understanding of the physics concepts.

Students have in general been accepting of using the Arduino as a laboratory platform. This is encouraging given the lack of familiarity with the Arduino prior to taking this course.The majority of the students in this course are chemistry majors. The chemistry program does not put an emphasis on coding and many of these students are seniors who will not get further exposure. Physics majors in the class can be followed through their degree to see if this has been beneficial. While this is the last class that uses the Arduino, there are other upper division courses that are incorporating more coding. Most physics majors take the fall-spring sequence of freshman courses (University Physics 1 and 2). This provides a control group that can be compared to the 8 majors that completed the fall 2016 semester using the Arduino.

Activities that use the Arduino as the platform for data acquisition are still in development. Assessments that can measure improvement in coding skills also need to be developed. Longitudinal studies of these students as they progress will need to be carried out to determine if this is an effective method of improving student skills in research laboratories.

References

[1] Ruth Chabay and Bruce Sherwood. Computational physics in the introductory calculus-based course. *American Journal of Physics*, 76(4):307–313, 2008. doi: 10.1119/1.2835054. URL http://dx.doi.org/10.1119/1.2835054.

[2] Todd Timberlake and Javier E. Hasbun. Computation in classical mechanics. *American Journal of Physics*, 76 (4):334–339, April 2008.

[3] Marcos D. Caballero, Matthew A. Kohlmyer, and Michael F. Schatz1. Implementing and assessing computational modeling in introductory mechanics. *Phys. Rev. ST Phys. Educ.*, 8(2):020106–1, 2012.

[4] Vernier Software & Technology. https://www.vernier.com/engineering/arduino/.

[5] Amy Biegalski P.E., Kevin Kit, and Rachel McCord. Converting traditional engineering physics laboratories into self-designed student explorations. In *2016 ASEE Annual Conference & Exposition*, number 10.18260/p.26589, New Orleans, Louisiana, June 2016. ASEE Conferences.

[6] Brian Huang. Open-source hardware – microcontrollers and physics education – integrating diy sensors and data acquisition with arduino. In *2015 ASEE Annual Conference & Exposition*, number 10.18260/p.24542, Seattle, Washington, June 2015. ASEE Conferences.

[7] ARDUINO. https://www.arduino.cc/en/tutorial/capacitancemeter.

[8] D. Hestenes, M. Wells, and G. Swackhamer. Force concept inventory. *Physics Teacher*, 30(3):141, 1992.

[9] L. Ding, R. Chabay, B. Sherwood, and R. Beichner. Evaluating an electricity and magnetism assessment tool: Brief electricity and magnetism assessment. *Phys. Rev. ST Phys. Educ.*, 2(1):7, 2006.

[10] PhysPort AAPT. https://www.physport.org/dataexplorer/.