The Use of Arduino in Physics Laboratories

Gürcan UZAL
Tekirdag Namık Kemal University, Vocational School of Technical Sciences, Department of Electronics and Automation, 59030 Suleymanpasa-Tekirdag/Turkey
guzal@nku.edu.tr
ORCID ID: 0000-0002-2029-8612

ABSTRACT
Arduino is easy to use because it is a simple system. In addition, since Arduino has an open source code system, it is a system that is open to everyone's use, can be developed and can be easily implemented. Anyone who wants to use Arduino can buy and use the necessary parts for their application. It is a platform that can be very useful in the physics lab due to its low price and wide availability of sensors and transducers. In this article, the Arduino platform is briefly introduced, and by installing an RC circuit, the charge-discharge curve of the capacitor has been drawn on the serial plotter. In addition, distance measurement was calculated by using an ultrasonic distance sensor and time measurement between two sensor events was calculated with two infrared obstacle sensors. The measurement results are given in the serial monitor. As a result, several examples of what can be done in laboratories in physics experiments using Arduino and some sensors have been shown.
Keywords: Arduino, physics laboratory, RC charge and discharge, voltage measurement, distance measurement

INTRODUCTION

In education, especially in physics education, new trends in active learning are emerging. These trends are aimed at ensuring student-oriented learning. "Problem-Based Learning", "Project-Based Learning", "Collaborative Learning" and "Inquiry-Based Learning" are the most important strategies used to ensure that learners actively participate in the learning process (Celik, Senocak, Bayrakceken, Tashkesenligil, & Doymus, 2005, p. 1). The teacher can facilitate the learning of physics by using one or more of these strategies according to the attitudes of the students in the classroom towards physics classes, their readiness, and their interest in physics class.

Laboratory activities are important in physics teaching, because by arousing students' interest and by putting emphasis on the learning of physics, they teach individuals to ask questions, identify problems and seek out solutions by working in collaboration with those around them. Laboratory experiences enable students to understand the laws of physics, recognize and understand physics concepts, and improve their scientific skills (Darrah, Humbert, Finstein, Simon, & Hopkins, 2014, p. 1; Sari, Pektas, Celik, & Kirindi, 2019, p. 3). The limited number of tools used in the laboratory does not allow every student to conduct experiments at the same time. In this respect, the physics laboratory can be enriched by designing measurement tools and experimental setups with Arduino (Gungor Babaoglu, Durmaz, & Oztekin, 2020, p. 93; Organtini, 2018, p. 1). In this study, several practical examples are given of how teachers can use the Arduino platform in the physics experiments they will be conducting in the lab.

ARDUINO'S PLATFORM

The Arduino development team consists of Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino and David Mellis. The Arduino platform is an open-source system consisting of electronic cards, expansion cards, sensors and software development environment (Tasdemir, 2015, p. 5; Cobanoglu, 2017, p. 1). Its hardware and software are open source, and its cost is preferred and prevalent by large audiences due to its benefits, such as the redundancy of code samples. In addition, thanks to the Arduino compatible shield and sensor diversity, the application is very user-friendly with many advanced technologies (Wi-Fi, GSM, GPS, etc.) (Cobanoglu, 2017, p. 23).

Arduino has different boards with various features. The simplest and most widely used is Arduino Uno. All boards share the same programming environment and may differ in board size, memory size, number of ports, type, and speed. The use of Arduino UNO board as a physics laboratory tool is an excellent solution (Organtini, 2018, p. 2).

The Arduino UNO board (Figure 1) is a microcontroller board based on the Atmega328 family of microcontrollers. It has a total of 14 digital input output ports and 6 of them are used as Pulse Width Modulation (PWM) outputs.

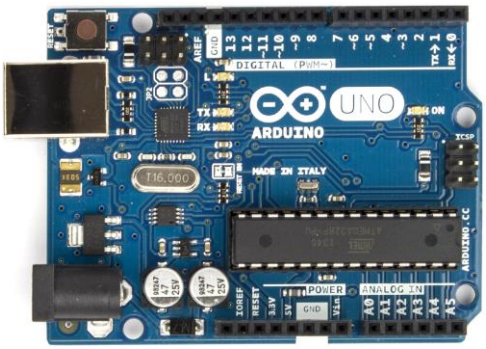

Figure 1: The Arduino UNO board.

Arduino Uno uses serial communication protocols such as UART, SPI, to communicate with environmental interfaces. ATmega328 UART TTL (5V) provides serial communication with RX and TX (pins 0 and 1). You can understand when the communication takes place from the blinking of the Rx, Tx leds on the Arduino board (Cobanoglu, 2017, p. 34).

ARDUINO PROGRAMMING

It is a platform that uses the C/C++ and Java grammar structure. Due to the programming environment, no in-depth object-oriented programming knowledge is required. Unlike computers, microcontrollers do not run an operating system, but only perform the only tasks that have been loaded into their memory (Organtini, 2018, p. 3).

The user can use the Integrated Development Environment or the Arduino Software (IDE), available free on the Arduino website, easily on a computer, to write a computer program. The IDE includes a text editor for writing code, a message box, a text console, a toolbar with buttons for commonly used functions, and a set of menus. Arduino platforms can easily be connected to any computer with Windows, MAC or Linux operation systems thanks to the USB interface, which can quickly be programmed and tested (Cobanoglu, 2017, p. 23). When the IDE is launched, it appears as in Figure 2.

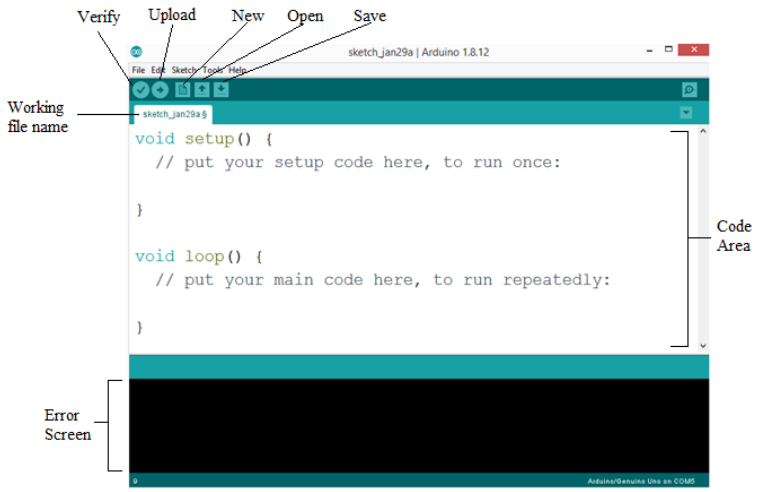

Figure 2: The Arduino IDE window.

Void setup() and void loop(), which are included in the Arduino code display in Figure 2, are the two main functions of the workspace. The void setup() part of the program is the part where the input – output pin modes are set, the task / variable definition taken from the library in the program is defined, and the first values of the defined variables are assigned. After the program is installed on the board, this part runs once, and then transfers the task to the void loop() section. In the void loop() section, the codes that your application wants to run continuously/indefinitely are written.

After the IDE is started, the codes of the work to be implemented are written in the void setup() and void loop() code area, compilation/checking is done, and if there are any errors or mistakes in the subsequent operations, the program reports errors in the error section at the bottom. This will continue until the code software errors are

fixed. After the arrangements are made, the necessary connections are made on the circuit and the program is installed on the board.

Serial Monitor

The Serial Monitor is a tool that we can view data sent via the serial port from the Arduino to the computer. The serial monitor is used to retrieve data from Arduino, view data for debugging, and send data (command) from PC to Arduino.

The PC and Arduino must be connected to each other with a USB cable for the serial monitor to function.

Figure 3: Arduino IDE window. Click the Serial Monitor Icon to open the Serial Monitor.

Items on the serial monitor

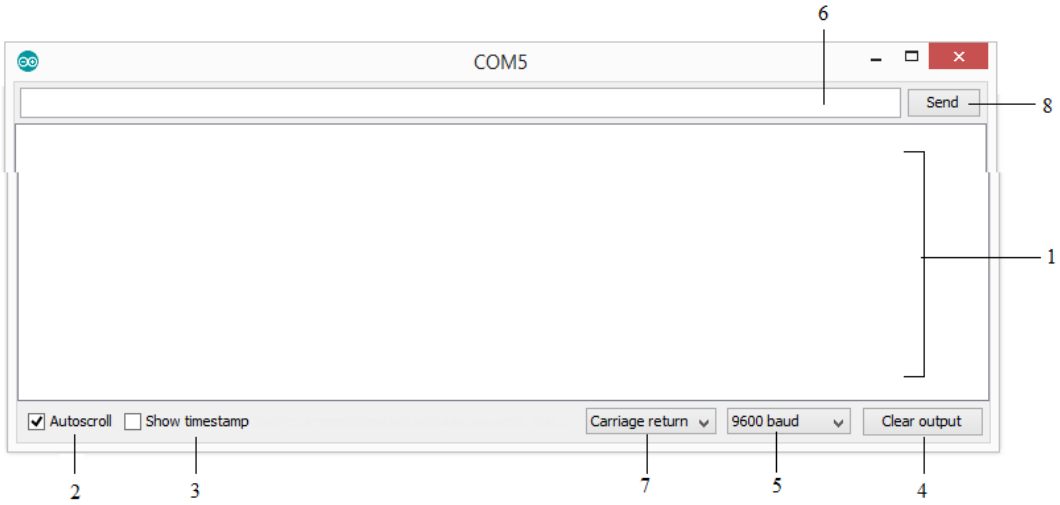

Figure 4: Serial Monitor Window.

1. Output console: Displays the data received from the Arduino.
2. Autoscroll checkbox: The option to choose between auto-scrolling and not scrolling.
3. Show timestamp: Option to view the time stamp before displaying the data on the serial monitor
4. Clear output: Clears all text in the output console.
5. Baud rate selection: The speed of the serial communication between Arduino and PC is selected.
6. Textbox: The user can enter the characters they want to send to the Arduino.
7. End selection: The ending character added to the data sent to the Arduino is selected.
8. Send button: When this button is clicked, the serial monitor sends the data in the text box and the ending character to the Arduino (Arduino-Serial Monitor, n.d.).

Serial Plotter

The Serial Plotter is a tool that can plot graphs of the data sent over the serial port of Arduino. The Serial Plotter can read temperature, pressure, humidity or any sensor data connected to Arduino analog input and visualize them as waveforms. It can plot graphs of the data it receives from more than one sensor at the same time.

Data communication between the serial plotter and Arduino is done with a USB cable. Therefore, Arduino and PC must be connected to each other with this cable. The baud rate of the Serial Plotter must be selected to be the same as the Arduino code.

x axis: Represents time. The axis has 500 sample points. The time between each point is usually equal to the time it takes to execute the code in the void loop() function section.
y-axis: Represents the values obtained from the sensor output. The y-axis automatically adjusts itself according to the increase or decrease of the value read from the sensor (Arduino-Serial Plotter, n.d.).

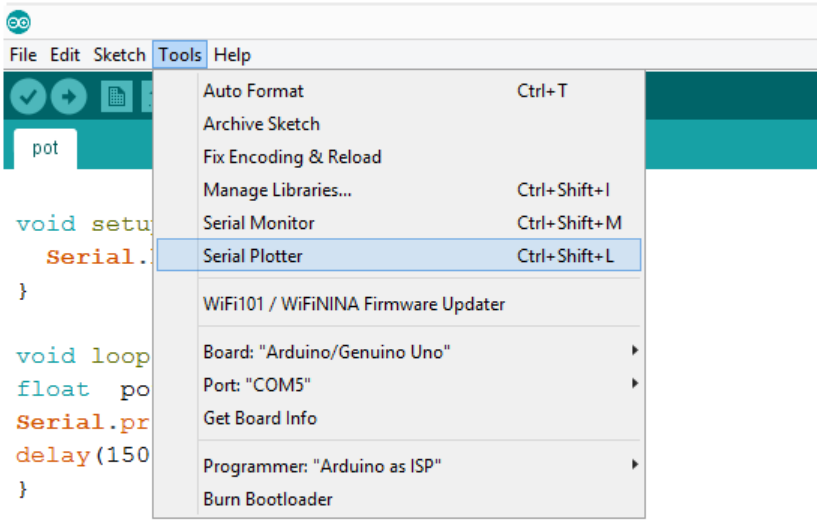

Figure 5: Opening the Serial Plotter.

Line charts
A line chart is used to show the changes in the current data that occur over a certain time period. It is possible to present these emerging data changes in an easier and more understandable form visually with the help of line graphs.

- **Drawing a single line graph**
  Printing a single chart can be done by sending the data and ending it with the character "\r\n". When writing code for it, the Serial.println (variable) function must be used. Serial.println() adds the characters "\r\n" automatically after the data (Arduino-Serial Plotter, n.d.).

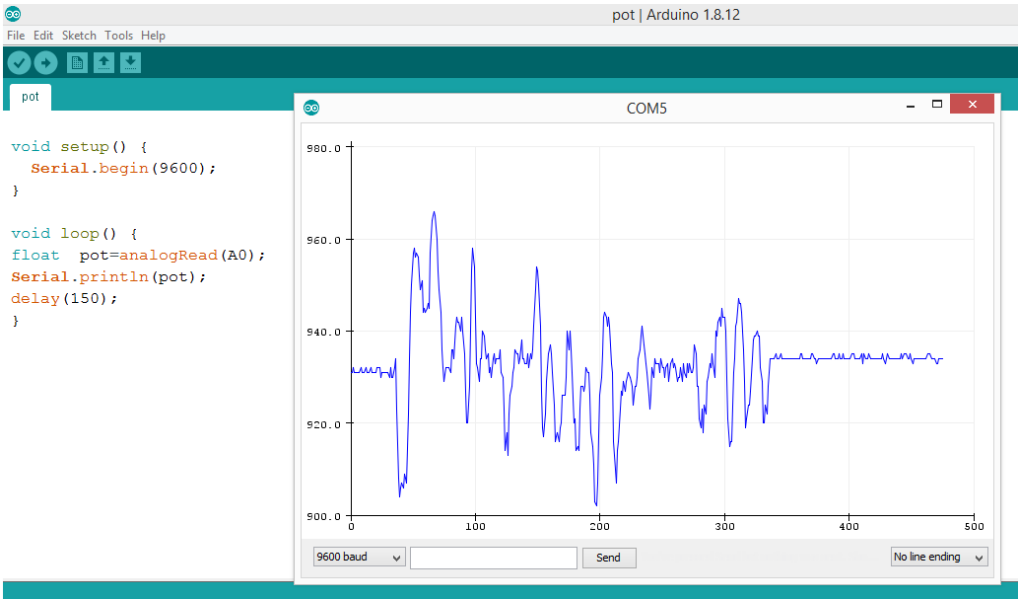

Figure 6: Drawing a single line graph of the values received from the analog pin A0 in the serial plotter.

- **Drawing multiple lines on the graph**
  When we want to plot multiple variables, we must separate the variables with the character "\t" or " ". The last value must be terminated with the character "\r\n" (Arduino-Serial Plotter, n.d.).

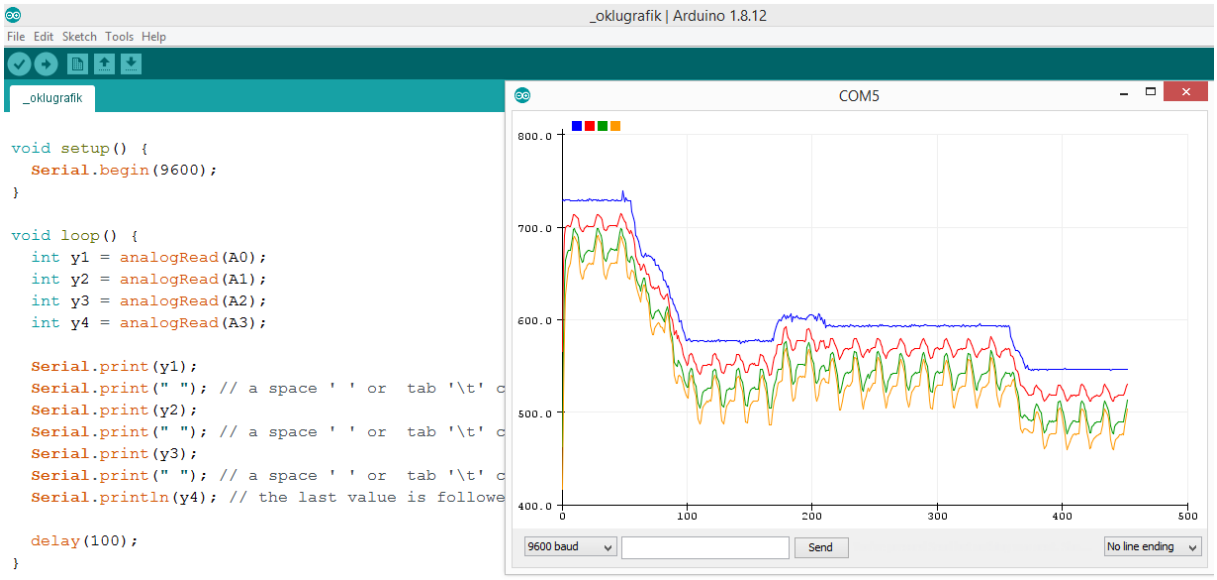

Figure 7: Plotting a multi-line graph of the values read from analog input pins A0, A1, A2, A3 and A4 on the serial plotter.

Using Analog Ports

Arduino Uno has six analog inputs from A0 to A5. Since each of these analog inputs has a resolution of 10 bits, it converts the input voltage into numbers from 0 to 1023. When the grounds of the circuit to be measured with the Arduino are connected to each other, the analog input pin allows electrical measurement with a resolution of 5/1024 ≈ 5mV. (Cobanoglu, 2017, p. 216). Voltage values greater than 5V can also be measured using voltage divider resistors.

Using Digital Ports

Each of the 14 digital pins on the Arduino Uno can be used as both an input and an output. Each works with 5V and draws 40mA current. Digital ports can be used for various operations, such as control and measurement. For example, interesting measurements in mechanics (one-dimensional position and time etc) can be made using ultrasonic sensors.

Digital pins can be used to represent data in binary form or for complex measurements using existing sensor modules (Organtini, 2018, p. 6).

ARDUINO SHIELDS AND SENSORS

Shield (Plug-in) cards are plug-ins that are installed on the Arduino board and provide special functions. With these add-ons, the capacity or functionality of the card can be increased. There are many low-cost shields, and their libraries are available. For example, the current capacities of the digital outputs on the Arduino microcontroller board are usually not enough to drive a servo or stepper motor, so a motor driver shield can be added to the Arduino board (Tasdemir, 2015, p. 15; Cobanoglu, 2017, p. 9). Physical values in the outside world can easily be detected using sensors with Arduino. Temperature, current, humidity, voltage, pressure, force, light, sound, acceleration, magnetic field, etc. sensors are available. These sensors allow the performance of many physics experiments. In addition, data collected using Arduino and sensors can be transferred to excel via the Parallax data Acquisition Tool (PLX-DAQ) program and easily graphed (Gungor Babaoglu, Durmaz, & Oztekin, 2020, p. 97).

MEASUREMENT EXAMPLES WITH ARDUINO

The Arduino pins are designed to measure voltages from 0 (zero) to 5V. Many measurements can be made provided that the physical changes obtained from the sensors are converted into voltage. When measuring with sensors, it is necessary to know the operating principles and limits of the sensors in order to prevent incorrect measurements or to keep the measurement errors under control.

Voltage Measurement

To obtain the value of the voltage in Volts, it is necessary to multiply the read value by a conversion factor, which is given as C = 5/1023 ≃ 0.00489. When writing code, it should be taken into account that the value returned by analogRead() is an integer and a line with int value = analogRead(A5)* 5/1023; (Organtini, 2016, p. 46). This type of measurement can be useful for all converters that provide voltage as their output, or for electrical measurements. Using this technique, for example, the time constant of an RC circuit can easily be measured. An RC circuit is a series of capacitors with capacitance C and a resistor with a resistance of R.

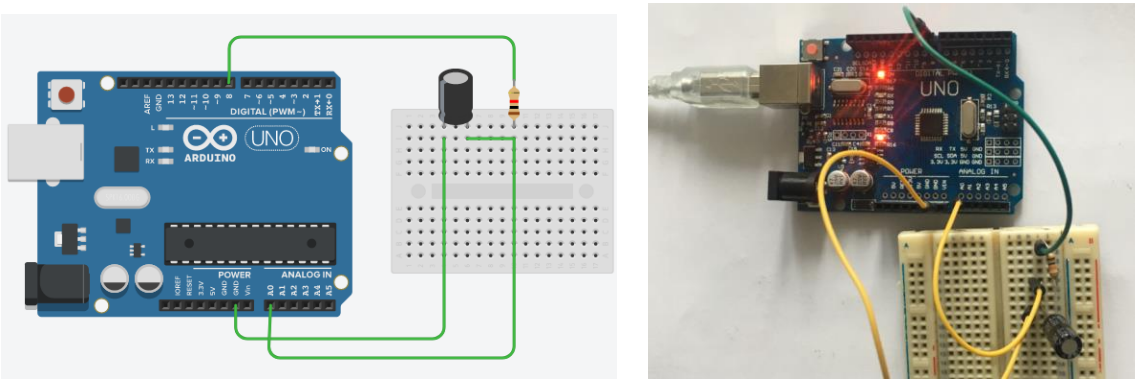

Figure 8: An RC measuring circuit.

The circuit in Figure 8 is connected. First of all, the Arduino's 8th digital pin is set to HIGH. The 470µF capacitor is charged via a 10KΩ resistor. When the capacitor is charged, the voltage reaches its final value exponentially, and this is indicated as $Vc(t) = Vo(1 - e^{-\frac{t}{RC}})$. The RC time constant indicates the fill rate.

Arduino's 8th digital pin is set to LOW and is ensured that the capacitor is discharged through the 10KΩ resistor. As the current flows, the voltage at the capacitor ends decreases. The Q charge is exhausted, and as a result, the circuit is also de-energized. This process creates an exponentially decreasing voltage, and this situation is denoted as $Vc(t) = Vo(1 - e^{-\frac{t}{RC}})$. As the RC time constant indicates the fill rate, here the discharge rate is indicated.

The charge and discharge voltage at the capacitor ends are obtained from the Arduino's analog pin A0 with 71 measurements at intervals of 30ms. Arduino Codes (Organtini, 2018, p. 5; Morresi & Piermarteri, 2020) and charge-discharge curves are shown in Figure 9.

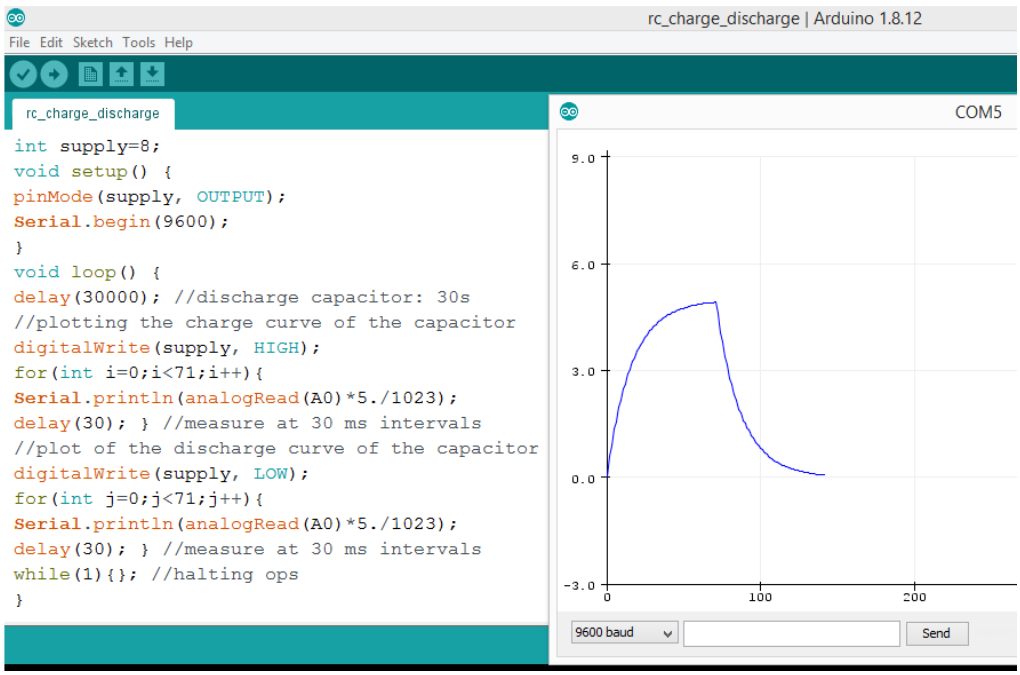

Figure 9: Plotting the charge and discharge curve of a capacitor with Arduino codes.

Distance Measurement

Distance measurement is a basic type of measurement. Converting a distance to a voltage is not such an easy task. But interesting measurements can be made in the field of mechanics with sensors that are easily available on the market, for example, ultrasonic sensors. Distance can be measured using an existing module consisting of two piezoelectric devices (such as the HC-SR04), one consisting of a transmitter and the other a receiver. Most of these devices use sound waves. They use ultrasound, which is a very short wave (or very high frequency) sound, to avoid false measurements and not harm people nearby. Sound waves move in air at a constant speed of about c ≃ 340 m/s. Ultrasonic sensors consist of a transmitter (speaker) and a receiver (microphone) for ultrasonic waves. The transmitter produces a series of waves that are reflected by any obstacle in front of it (if it is large enough) (Arun Francis, Arulselvan, Elangkumaran, Keerthivarman, & Vijaya Kumar, 2019, p. 207; Gabriel & Kuria, 2020, p. 937).

The time it takes for the reflected sound to reach the receiver,

$$t = 2\frac{d}{c}$$

t is the time taken for the reflected wave sequence to be detected by the receiver.
d, the distance between the receiver and the obstacle.
c, the speed of propagation of sound in the air.
2 the coefficient set for the pulses going from the transmitter to the obstacle and back to the receiver.

The sensor has an electronic circuit which measures the time and generates a pulse whose duration is proportional to the time. You can then trigger the device with the digital pins and read the duration of the measuring pulse with the pulseIn() function. The value read is proportional to t. It can be used to get the distance by reversing the above equation (Organtini, 2018, p. 6; Abdulkhaleq, Hasan, & Salih, 2020, p. 3).

$$d = \frac{c * t}{2}$$

There are two kinds of sensors on the market with three pins and four pins. See the datasheets for these sensors for details on how they work.

Figure 10: Three-and four-pin ultrasonic distance sensor.

You can easily measure distances with the ultrasonic distance sensor HC-SR04. In Figure 11, the pin labeled GND should be connected to the Arduino ground, the pin labeled Vcc should be connected to the Arduino's 5V pin, and the Trig and Echo should be connected to the digital and PWM pins, respectively. Also, the trigger pin should be defined as an output pin and the echo pin should be defined as an input pin.

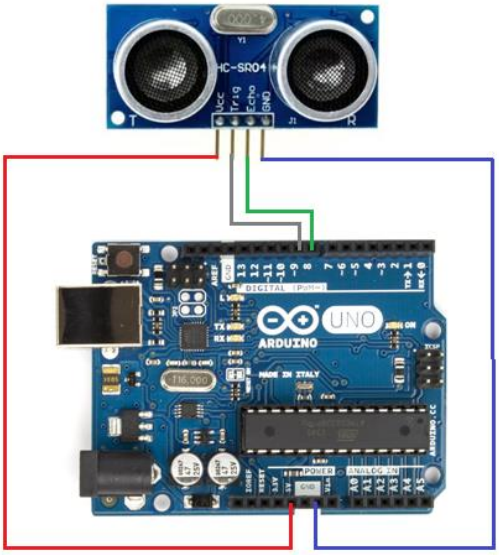

Figure 11: Connection diagram of HC-SR04 with Arduino.

Figure 12: The result of a measurement performed with the HC-SR04

distance measurement circuit.

```
#include <Ultrasonic.h>
const int TrigPin = 9;
const int EchoPin = 8;
long Duration;
int DistanceCm;

void setup() {
  Serial.begin(9600);
  pinMode(TrigPin, OUTPUT);
  pinMode(EchoPin, INPUT);
}
void loop() {
  digitalWrite(TrigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(TrigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(TrigPin, LOW);
  Duration = pulseIn(EchoPin, HIGH);
  DistanceCm = Duration * 0.034 / 2;
  Serial.print("Distance: ");
  Serial.print(DistanceCm); // Write the distance
                            //value of the sensor
  Serial.print(" cm");
  Serial.print("\n");
  delay(1000);// Wait 1 second
}
```

Figure 13: Arduino codes of distance measurement with HC-SR04 ultrasonic sensor.

The laws of physics are expressed in terms of elementary quantities with clear definitions. There are three basic quantities used in mechanics. These are length, time and mass. All other physical quantities, for example, velocity, acceleration, force, kinetic energy, etc. are expressed in terms of these basic quantities (Sarı, 2008, p.2). Time is one of the most precisely measurable quantities in physics. This quantity can be easily measured using the digital pins of the Arduino platform. For this, the millis() function of Arduino can be used. This function gives the time elapsed in milliseconds from the moment the current program starts running on the Arduino platform. Therefore, to calculate the time that a transaction takes, millis() can be called before and after the transaction and the difference of the two values can be taken. Sample application codes are given in figure 14.

```
void setup() {
// Enter your setup code here to run once:
Serial.begin(9600);
long int t1 = millis();
//task_whose_time_is_to_be_measured();
delay(1000); // processing time 1000 ms
long int t2 = millis();
Serial.print("Time range: "); Serial.print(t2-t1); Serial.println(" milliseconds");
}
void loop() {
// Enter your main code here to run it repeatedly:
}
```

Figure 14: Arduino codes that measure the time period of an operation (Sanghavi, 2021, para.1).

As mentioned above, the time between two sensor events can easily be measured. For this purpose, two infrared obstacle detection sensor (Obstacle detection IR sensor) modules can be used. The + pin of the sensors is connected to the 5V pin of the Arduino, the GND pin is connected to the GND pin of the Arduino, the out pin of the first infrared obstacle-detecting sensor module is connected to the 2nd pin of the Arduino and the out pin of the second infrared obstacle detecting sensor module must be connected to the 3rd pin of the Arduino. Also the 2nd pin and the 3rd pin must be defined as the input.

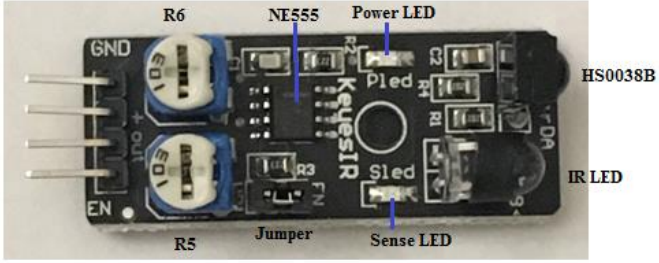

Figure 15: Keyes IR Obstacle Avoidance Sensor Module for Arduino.

This sensor is called Keyes, KeyesIR or Keyestudio KY-032. The sensor uses a four-pin connector with the pins labeled as follows: EN (Enable), Output (Out), + (Power), and GND (Ground). When the sensor detects an obstacle, its OUT output is LOW, and when no obstacle is detected, its output is HIGH. So, there is no need to install sensor related libraries. There are also two small potentiometers (variable resistors) and a jumper on the board. R6 is used to tune the 555 oscillator to exactly 38kHz. On the other hand, R5 limits and reduces the IR LED current when turned counter clockwise. These two settings together affect the sensitivity and range of the device. If you do not have an oscilloscope or frequency counter, it's best to keep the R6 as it's from the manufacturer or mid-range. Turning R5 clockwise will overload the 555 and end the process. Leaving the R5 in a full clockwise direction can lead to overheating and, eventually, to the device breaking down (IR Sensor for Obstacle Avoidance KY-032 (AD-032), n.d.).

Two key KeyesIR sensors are positioned so that there are 20 cm between them. The circuit measuring the time it takes for a toy car to cross a distance of 20 cm is shown in Figure 16.

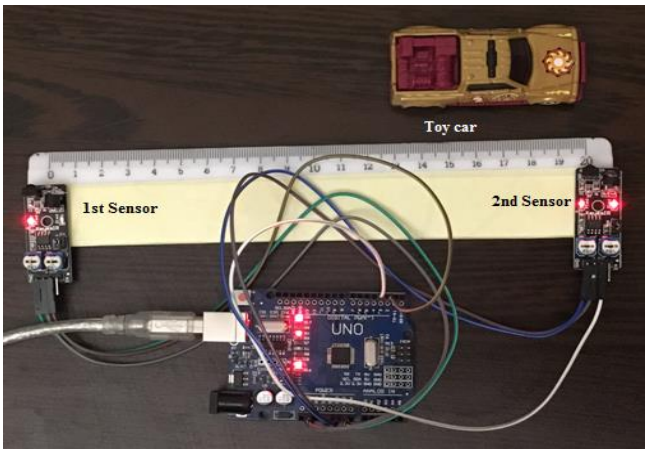

Figure 16: A circuit measuring the time interval with two IR sensors.

When the toy car comes in front of the 1st sensor, the time (t1) from the moment the program in Arduino starts running is measured as ms. When the toy car passes in front of the second sensor, the time passed is measured as (t2). The time (Δt=t2-t1) of the toy car to travel the distance between the two sensors is determined. This time is on the serial monitor in milliseconds and printed in the format of hours, minutes, seconds, and milliseconds. The codes to be loaded into the Arduino are given in Figure 17.

```
/*  Measuring the time interval  */

unsigned long t1, t2, elapsed;

void setup()
{
  Serial.begin(9600);
  pinMode(2, INPUT); // 1st sensor
  pinMode(3, INPUT); // 2nd sensor
  Serial.println("start the task to be
timed");
}

void displayResult()
{
  float h,m,s,ms;
            unsigned long over;
  elapsed=t2-t1;
  h=int(elapsed/3600000);//convert
milliseconds to
                          //hours
  over=elapsed%3600000;
```

```
Serial.println(elapsed);
  Serial.print("Elapsed time: ");
  Serial.print(h,0);
  Serial.print("h ");
  Serial.print(m,0);
  Serial.print("m ");
  Serial.print(s,0);
  Serial.print("s ");
  Serial.print(ms,0);
  Serial.println("ms");
  Serial.println();
}

void loop()
{
  if (digitalRead(2)==LOW)
  {
    t1=millis();
    delay(200); // for debounce
    Serial.println("Started...");
  }
  if (digitalRead(3)==LOW)
```

Figure 17: Arduino codes of time interval measurement (Boxall, 2013, p. 181).

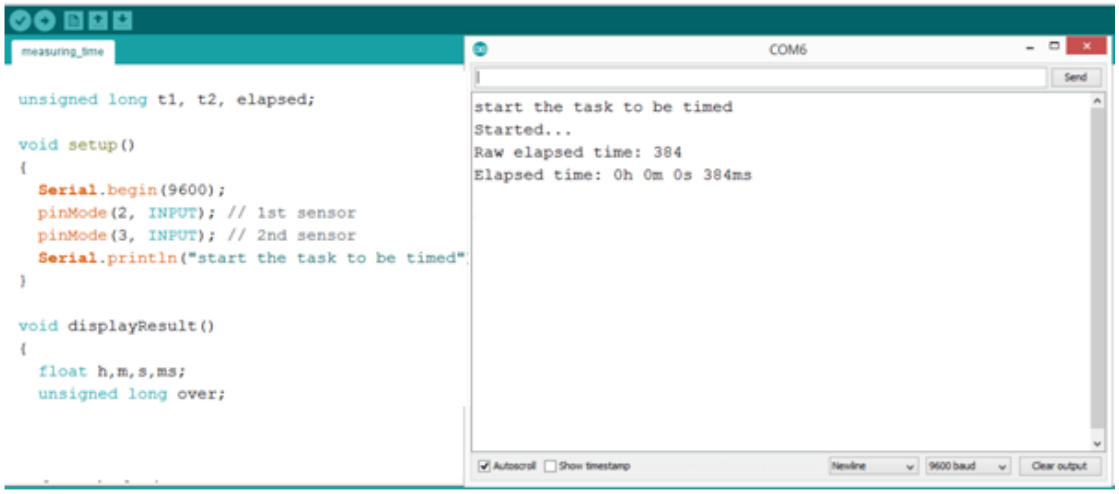

Figure 18: The result of a time interval measurement performed using two KeyesIR sensor modules.

CONCLUSIONS

In order to facilitate learning, a number of physical technologies are used in the field of educational technologies, such as computer hardware, software, a graphing calculator, sensor interfaces (LabQuest, Go!Link, Vernier Arduino Interface Shield, etc.), digital cameras, interactive whiteboard tools, and LCD projectors. Additionally, there are also technologies such as blogs, collaborative software, e-portfolios and virtual classes which consist of these technologies used together differently. (Wikipedia. (n.d.); Vernier. (n.d.)).

In a study conducted in recent years, it has been noted that the Arduino has been used primarily in physics education but also in areas such as chemistry, engineering, robotics, electronics and education. It has been stated that the Arduino has made application development possible in various fields from education to engineering. Moreover, it was emphasized that it contributes to students' learning of physics topics, concepts and principles by having them apply basic electrical and electronic knowledge, and to their practical realization of meaningful and permanent learning, through conducting experiments (Duman, 2019, p. 494).

Arduino can be used as an alternative for conducting experiments in physics, taking measurements and collecting data. Therefore, the physics laboratory can be enriched by designing measurement tools and experimental devices with Arduino. There are various studies with an introduction of sensors that can be used with Arduino along with the sizes to be measured. (Fisher & Gould, 2012, p. 8; Organtini, 2016, p. 45-46).

Likewise, there are studies related to the reorganization of electrical experiments carried out in physics laboratories with Arduino (Kirikkaya & Basaran, 2017, p. 351). In this context, it has been noted that projects could be developed to determine the electrical, mechanical and magnetic properties of substances (Bouquet, Bobroff, Fuchs-Gallezot, & Maurines, 2017, p. 3; Huang, 2015, p. 26.1205.2).

In relation to the use of Arduino in physics education; temperature dependency of electrical resistance (Sari & Kirindi, 2019, p. 688), investigation of simple harmonic motions within the scope of kinematic measurements (Music, 2017, p. 1; Tong-on, Saphet & Thepnurat, 2017, p. 1), have shown in various studies that magnetic field measurements can be performed (Organtini, 2016, p. 57).

In one of his studies, Nichols contributed to the development of measurement tools that can be used in the laboratory by showing that the data received from Arduino Uno could be transferred to programs such as Excel, LabVIEW, MATLAB (Nichols, 2017, p. 226).

In another study, a free fall experiment was examined by designing an Arduino-based experiment. This study showed that the gravitational acceleration of an object released from a certain height could be calculated using location and time data with the HC-SR04 ultrasonic distance sensor and Arduino (Moya, 2018, p. 1).

In this study, real-time monitoring of the values for physical quantities is provided with the voltage, distance and time interval measurements performed. The variety of sensors (acceleration, temperature, light, sound, magnetic field sensors, etc.) that can be used on the Arduino platform can provide many and interesting experiments in the

fields of Mechanics, Thermodynamics, Electromagnetism and Optics (Bouquet et al., 2017, p. 2; Oprea, 2018, p. 99).

As a result, due to the inexpensiveness of Arduino and its sensors, the variety of sensors to be used in physics experiments, and the ease of accessing various resources on this subject from the internet, it is thought that the use of Arduino in physics course laboratory activities will be beneficial to teachers, students and those who work as amateurs or professionals on this subject. In addition, since it is possible to design many measuring devices that can be used in physics laboratories with Arduino, it can be made easier and more enjoyable to learn physics in depth with these measuring tools.

REFERENCES

Abdulkhaleq, N. I., Hasan, I. J., & Salih, N. A. J. (2020). Investigating the resolution ability of the HC-SRO4 ultrasonic sensor. *IOP Conf. Series: Materials Science and Engineering.* 745, 012043. DOI:10.1088/1757-899X/745/1/012043

Arduino Serial Monitor. (n.d.). Arduino get started. Retrieved January 4, 2022 from https://arduino getstarted.com/tutorials/arduino-serial-monitor

Arduino Serial Plotter. (n.d.). Arduino get started. Retrieved January 14, 2022 from https://arduino getstarted.com/tutorials/arduino-serial-plotter

Arun Francis G, Arulselvan M, Elangkumaran P, Keerthivarman S., & Vijaya Kumar J. (2019). Object detection using ultrasonic sensor. *International Journal of Innovative Technology and Exploring Engineering* (IJITEE), *8*(6S), 207-209.

Bouquet, F., Bobroff, J, Fuchs-Gallezot, M., & Maurines, L. (2017). Project-based Physics labs using low-cost open-source hardware. *American Journal of Physics*, *85*(3), 216-222.

Boxall, J. (2013). *Arduino workshop* (1st ed.). San Francisco: No starch press.

Castro, L. H. M., Lago, B. L., & Mondaini, F. (2015). Damped harmonic oscillator with Arduino. *Journal of Applied Mathematics and Physics*, *3*, 631-636.

Celik, S., Senocak, E., Bayrakceken, S., Taskesenligil, Y., & Doymus, K. (2005). A compilation study on active learning strategies. *Journal of Kazim Karabekir Education Faculty*, *11*, 155-185.

Cobanoglu, B. (2017). *In-depth arduino. Istanbul: Abacus Book Publishing Distribution Services.*

Darrah, M., Humbert, R., Finstein, J., Simon, M., & Hopkins, J. (2014). Are virtual labs as effective as hands-on labs for undergraduate physics? A comparative study at two major universities. *Journal of Science Education and Technology*, *23*(6), 803-814.

Duman, O. (2019). Eğitimde arduino kullanımı ile ilgili yapılan çalışmalar. *XII. Uluslararası Eğitim Araştırmaları Kongresi*, April 25-28, 2019, Rize, Turkey.

Fisher, D. K., & Gould, P. J. (2012). Open-source hardware is a low-cost alternative for scientific instrumentation and research. *Modern Instrumentation*, *1*(2), 8-20.

Gabriel, M. M., & Kuria, K. P. (2020). Arduino uno, ultrasonic sensor HC-SR04 motion detector with display of distance in the LCD. *International Journal of Engineering Research&Technology* (IJERT), *9*(5), 936-942.

Gungor Babaoglu, M., Durmaz, K. K., & Oztekin, M. E. (2020). Calculation of gravitational acceleration with Arduino. *Journal of Science Education*, *8*(1), 92-100.

Huang, B. (2015). Open-source hardware - microcontrollers and physics education - integrating diy sensors and data acquisition with arduino. *122nd ASEE Annual Conference & Exposition*, June 14-17, 2015, Seattle, WA.

IR Sensor for Obstacle Avoidance KY-032 (AD-032), (n.d.). IR sensor. Retrieved March 18, 2022 from http://irsensor.wizecode.com/

Kırıkkaya, E. B., & Başaran, B. (2017). Fizik laboratuvarında gerçekleştirilenelektrik deneylerinin arduino programı ile yeniden düzenlenmesi. *IV. International Eurasian Educational REsearch Congress*, May 11-14, 2017, Bayburt, Turkey.

Morresi, L., & Piermarteri, A. K. (2020). *Embedded system architecture notes.* Retrieved March 22, 2022 from https://luciano. defalcoalfano.it/esa-appunti/E07_sheet06.html

Moya, A.A. (2018). An arduino experiment to study free fall at schools. *Physics Education*, *53*(5), 055020, DOI: 10.1088/1361-6552/aad4c6.

Music, P. (2017). Development of computer-based experiment set on simple harmonic motion of mass. *The Turkish Online Journal of Educational Technology*, *16*(4), 1-11.

Nichols, D. (2017). Arduino-based data acquisition into. *The Physics Teacher*, *55*, 226-227.

Opera, M. (2018). The Integration of the arduino platform in the physics Lessons. *International Scientific Conference eLearning and Software for Education*. *2*, 99-106, Bucharest: "Carol I" National Defence University. DOI:10.12753/2066-026X-18-084. Retrieved from https://www.proquest.com/ docview/2038220795/fulltextPDF/325AE7847344C9CPQ/1?accountid = 25088

Organtini, G. (2018). Arduino as a tool for physics experiments. *J. Phys.: Conf. Ser.*, *1076*, 012026
Organtini, G. (2016). *Scientific Arduino Programming*. Retrieved from https://www.roma1.infn.it/people/organtini/publications/scientificArduino.pdf.
Sari, H. (2008). *Part 1: Physics and Measurement.* Retrieved from https://acikders.ankara.edu.tr/pluginfile.php/279/mod_resource/content/2/fizik%20ve%20%C3%B6l%C3%A7me.pdf
Sari, U., Pektas, H. M., Celik, H., & Kirindi, T. (2019). The effects of virtual and computer based real laboratory applications on the attitude, motivation and graphic skills of university students. *International Journal of Innovation in Science and Mathematics Education*. *27*(1), 1-17.
Sarik, J., & Kymissis, I. (2010). Lab kits using the Arduino prototyping platform. *40th ASEE/IEEE Frontiers in Education Conference*, October 27 - 30, Washington, DC.
Sanghvi, Y. (2021). Calculate time of operation in Arduino. Retrieved from https://www.tutorialspoint.com/calculate-time-of-operation-in-arduino#:~:text=Often%2C%20you%20need%20to%20measure,started%20running%20the%20current%20program.
Tasdemir, C. (2015). *Arduino*. Istanbul: Dikeyeksen Publishing Distribution.
Tong-on, A., Saphet, P. ,& Thepnurat, M. (2017). Simple harmonics motion experiment based on LabVIEW interface for Arduino. *Journal of Physics: Conf. Series*, (s. 901 (012114)).
Wernier. (n.d.). Interfaces. Retrieved June 6, 2022 from https://www.vernier.com/product-category/?category=interfaces
Wikipedia. (n.d.). Educational technology. Retrieved June 6, 2022 from https://en.wikipedia.org/wiki/Educational_ technology#Computers,_tablets_and_mobile_devices