

# Physical Computing

Designing Physical Interactions for a Digital World

ARTS 370

Fall 2019

Wednesday 1:40PM – 5:30PM

Klapper 107

Professor Danne Woo

[pcomp.dannewoo.com](http://pcomp.dannewoo.com)

[dwoo@qc.cuny.edu](mailto:dwoo@qc.cuny.edu)

# Week 1-9

Week 1: What is Physical Computing?

Week 2: Introduction to Electronics

Week 3: Arduino, Hello World

Week 4: Analog Input and Output

Week 5: Digital and Analog Review

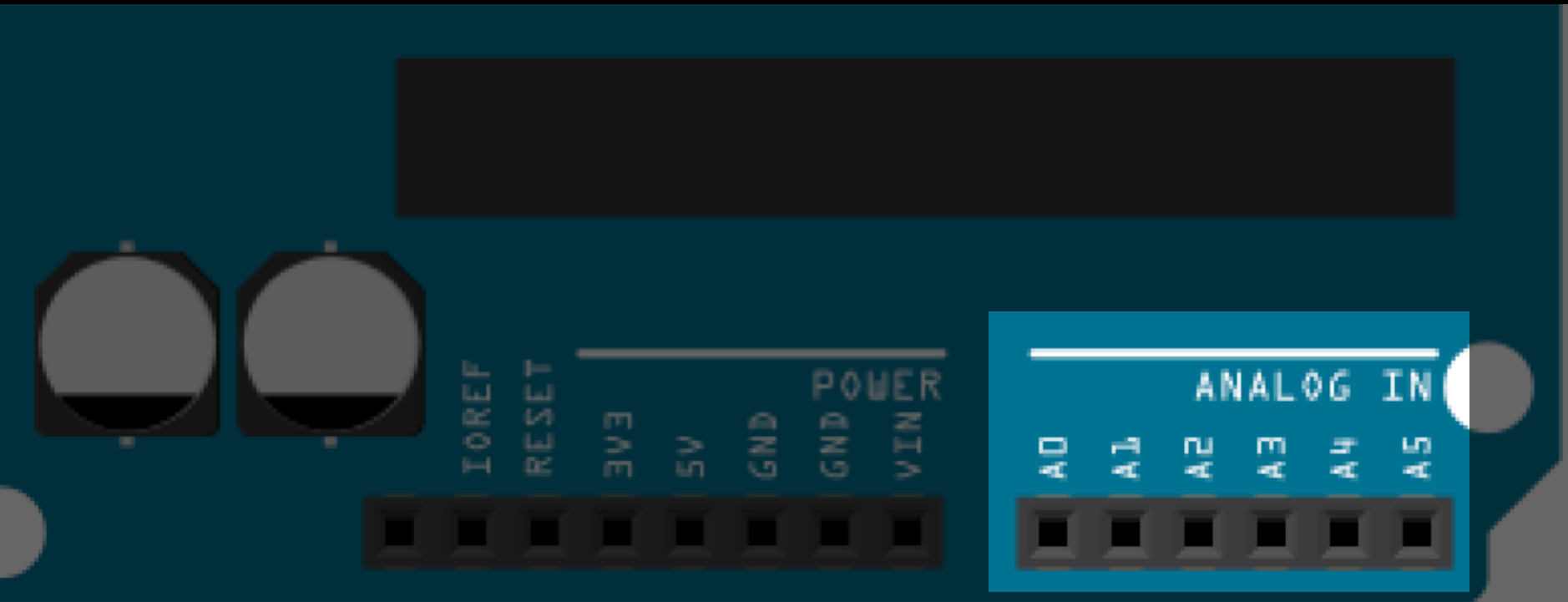
Week 6: Enclosures

Week 7: Serial Communication, Processing and p5.js

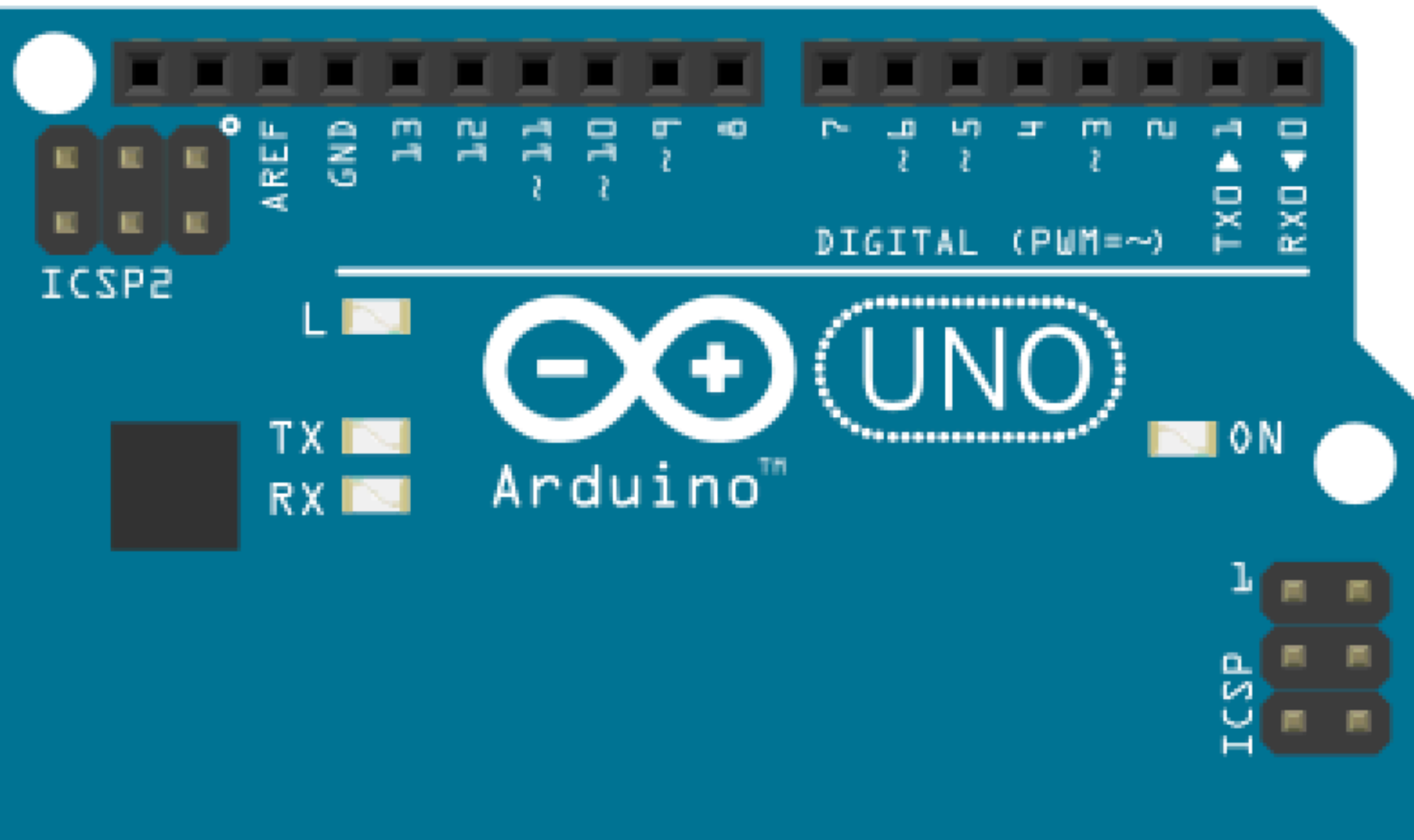
Week 8: Soldering Workshop

Week 9: Midterm Presentation

# Analog In



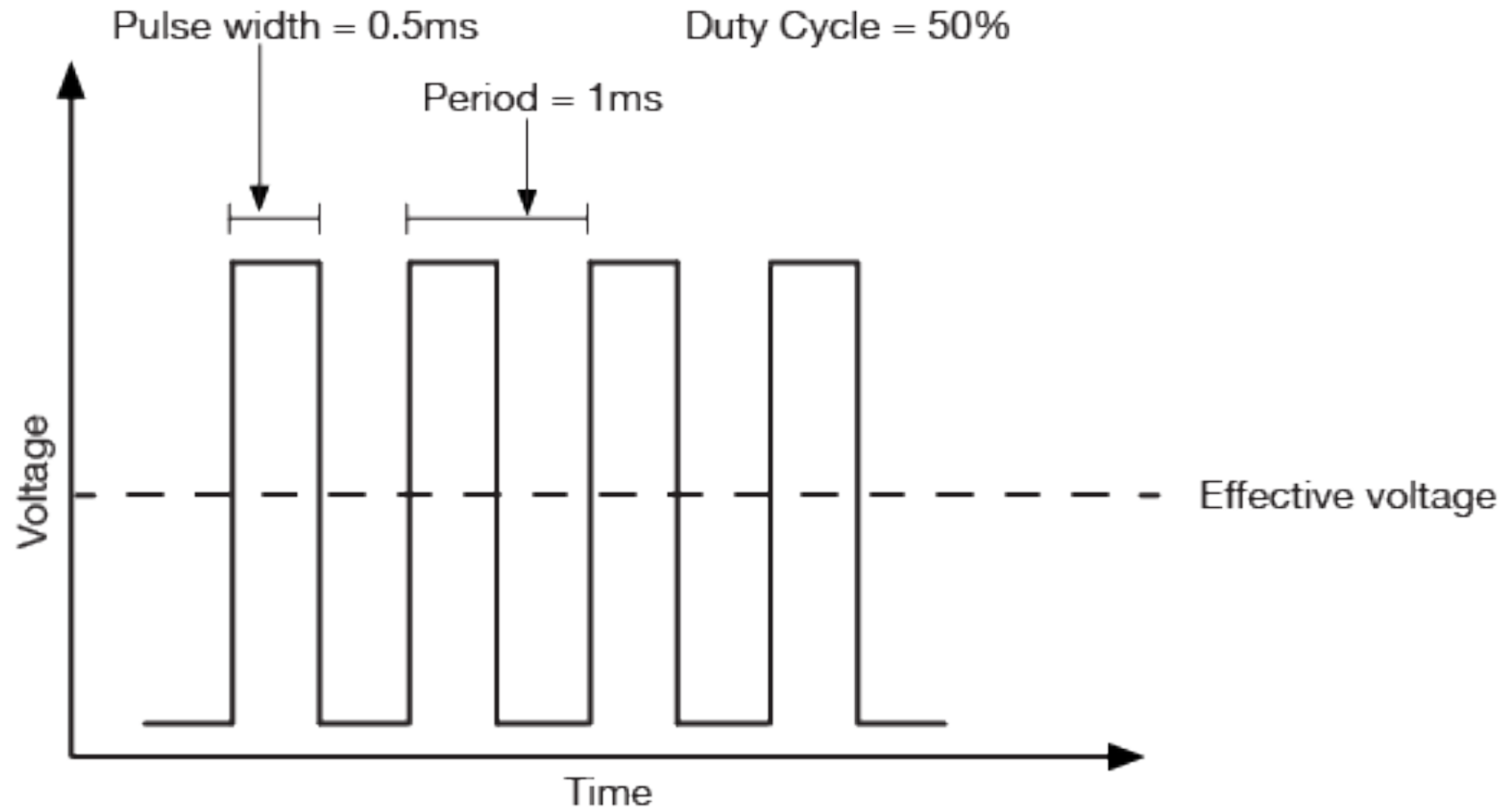
# Analog Out



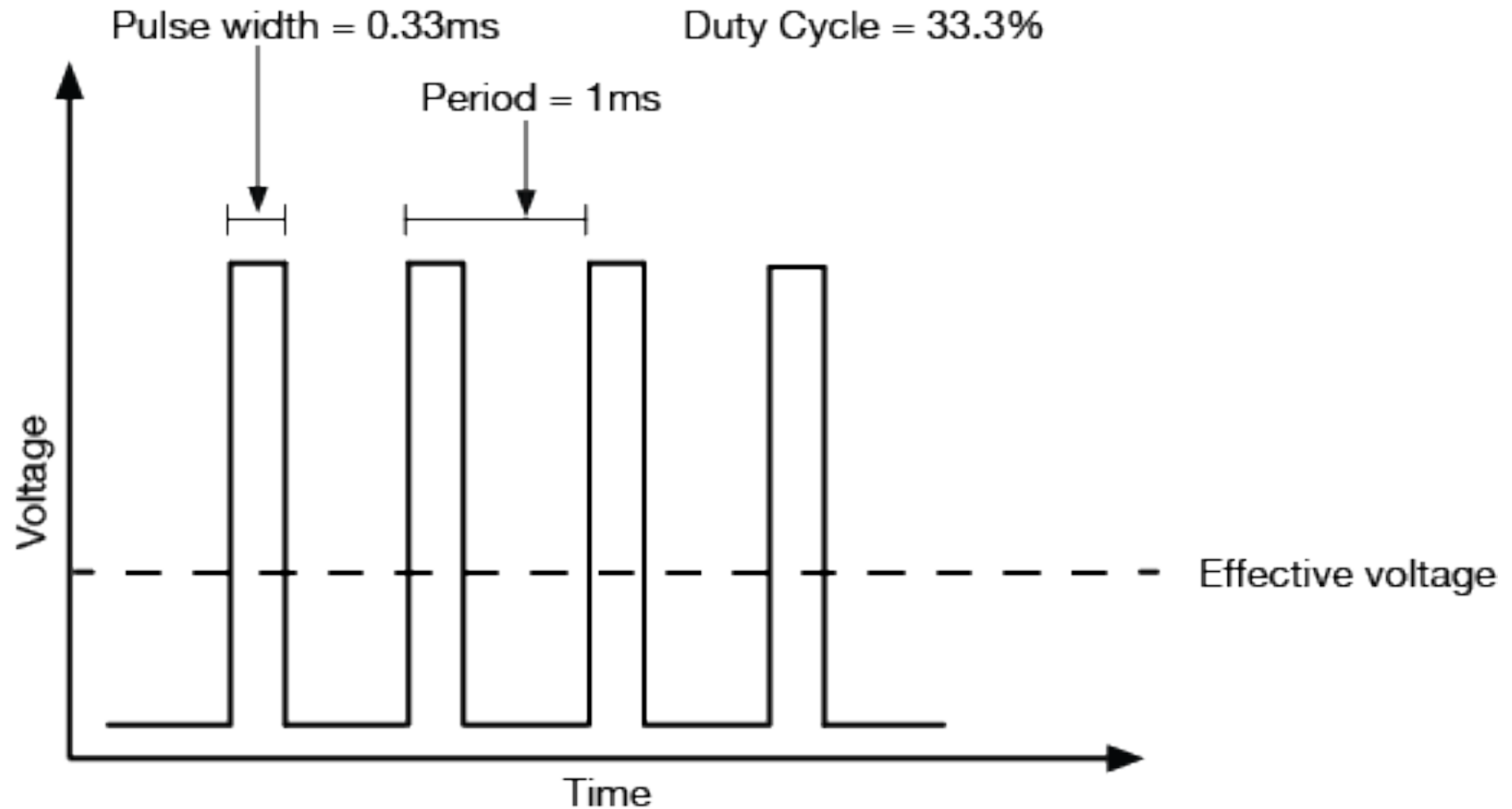
# Pulse Width Modulation (Fake Analog)



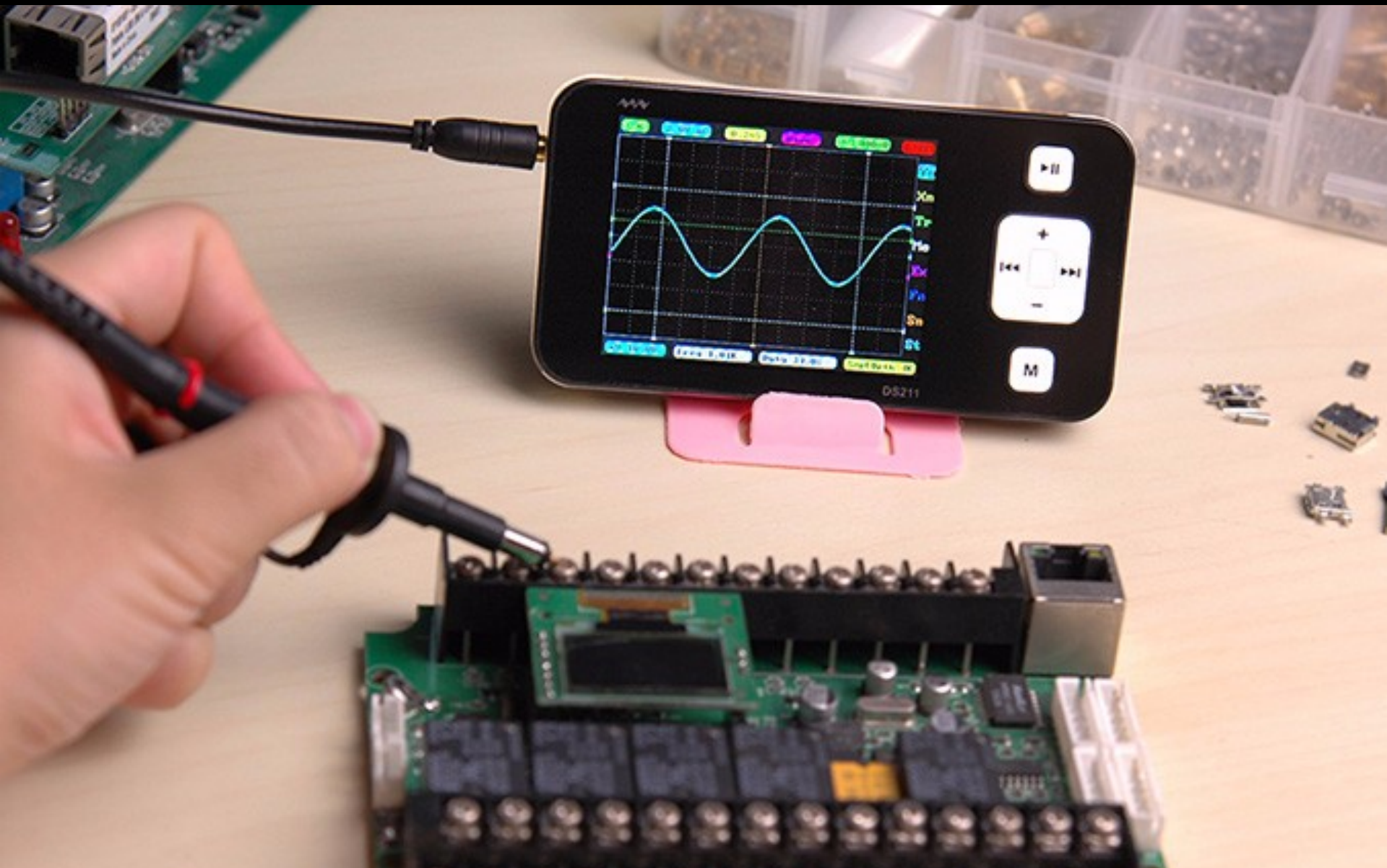
# Pulse Width Modulation (Fake Analog)



# Pulse Width Modulation (Fake Analog)

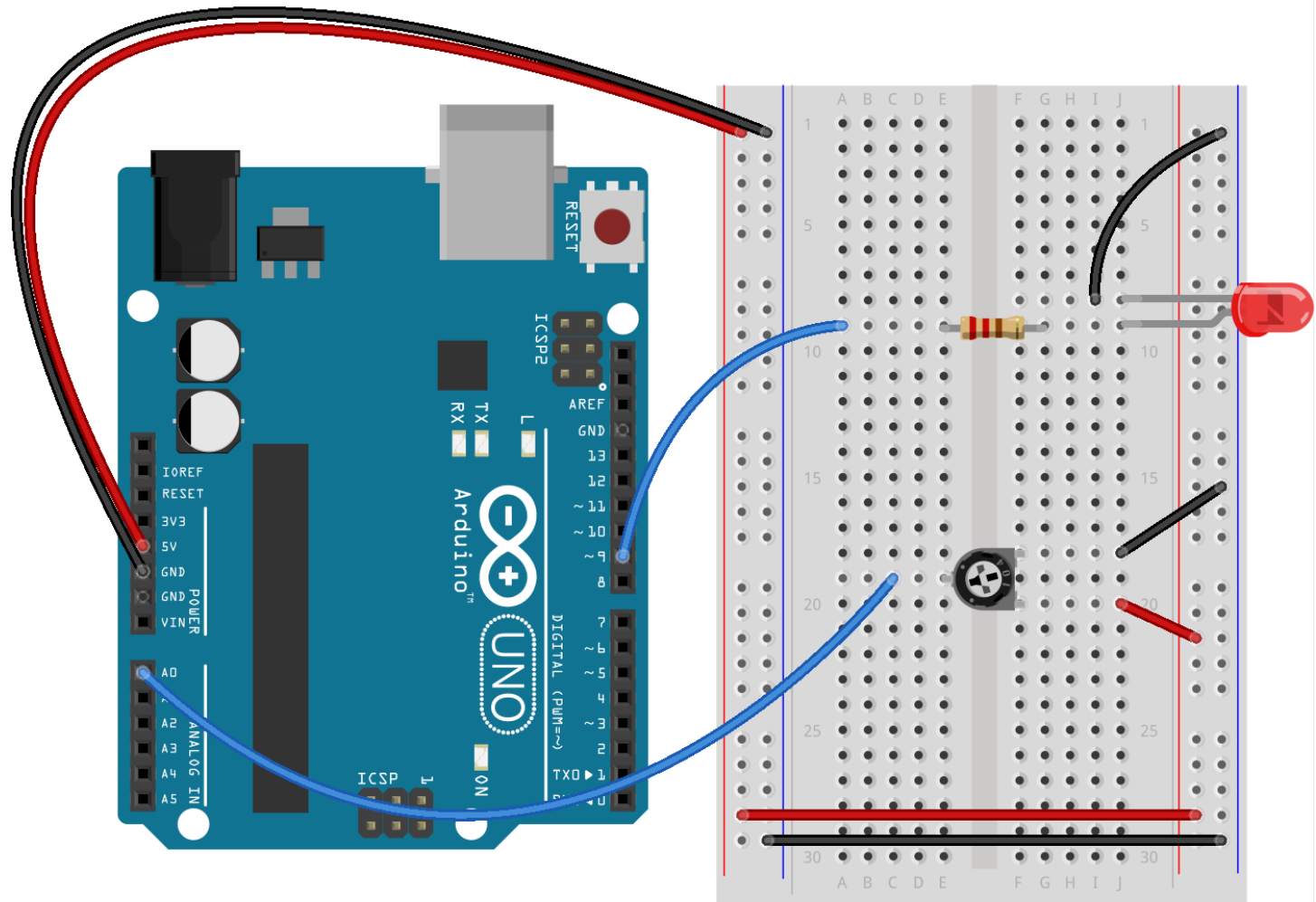


# Oscilloscope

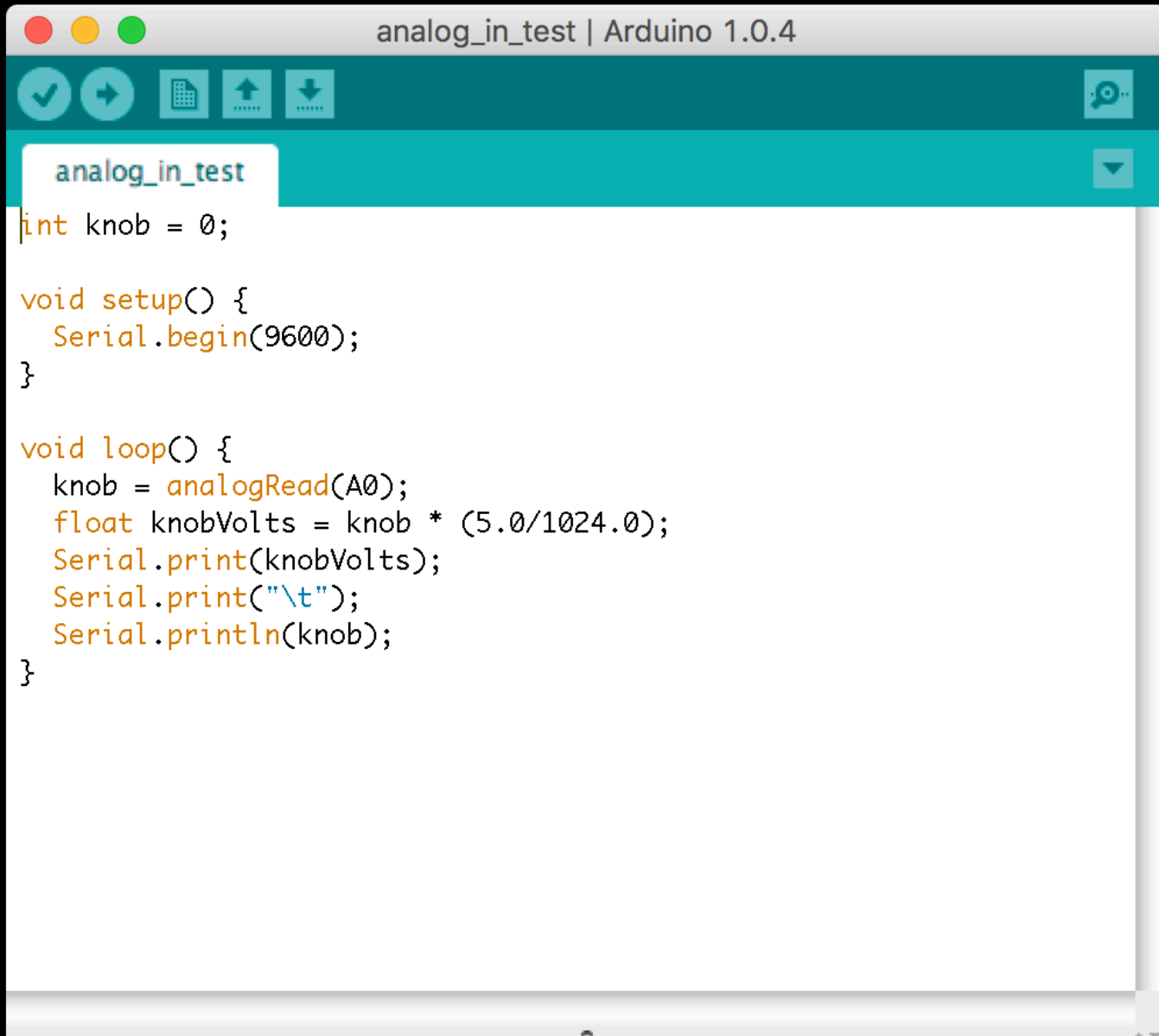




# Analog In and Out (Potentiometer)



# Analog In – Test Sensor Values

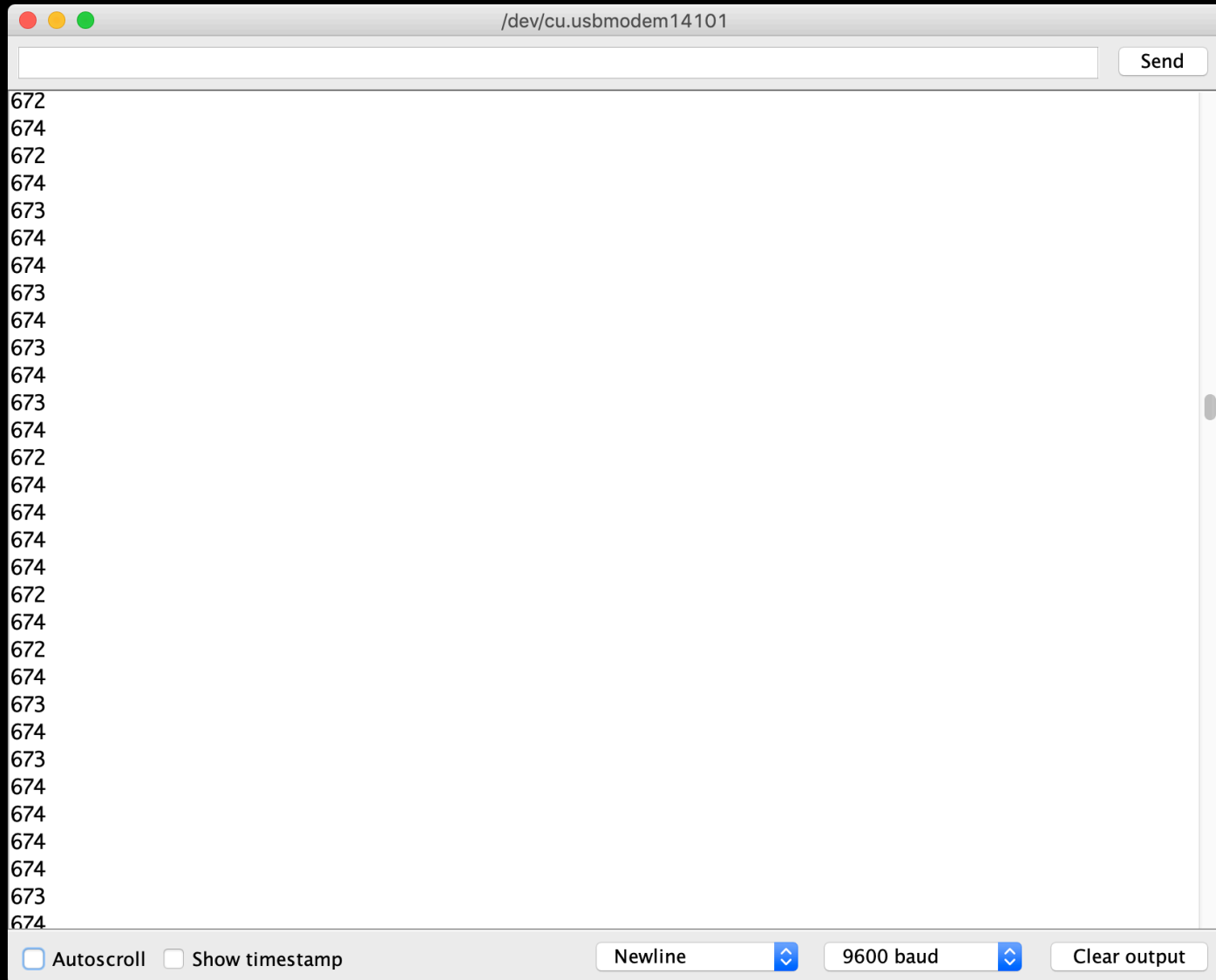
A screenshot of the Arduino IDE interface. The title bar at the top reads 'analog\_in\_test | Arduino 1.0.4'. Below the title bar is a toolbar with icons for checking, running, opening, uploading, and downloading. The main text area contains the following C++ code:

```
int knob = 0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  knob = analogRead(A0);
  float knobVolts = knob * (5.0/1024.0);
  Serial.print(knobVolts);
  Serial.print("\t");
  Serial.println(knob);
}
```

# Analog In – Serial Monitor



# Analog In and Out (Potentiometer)

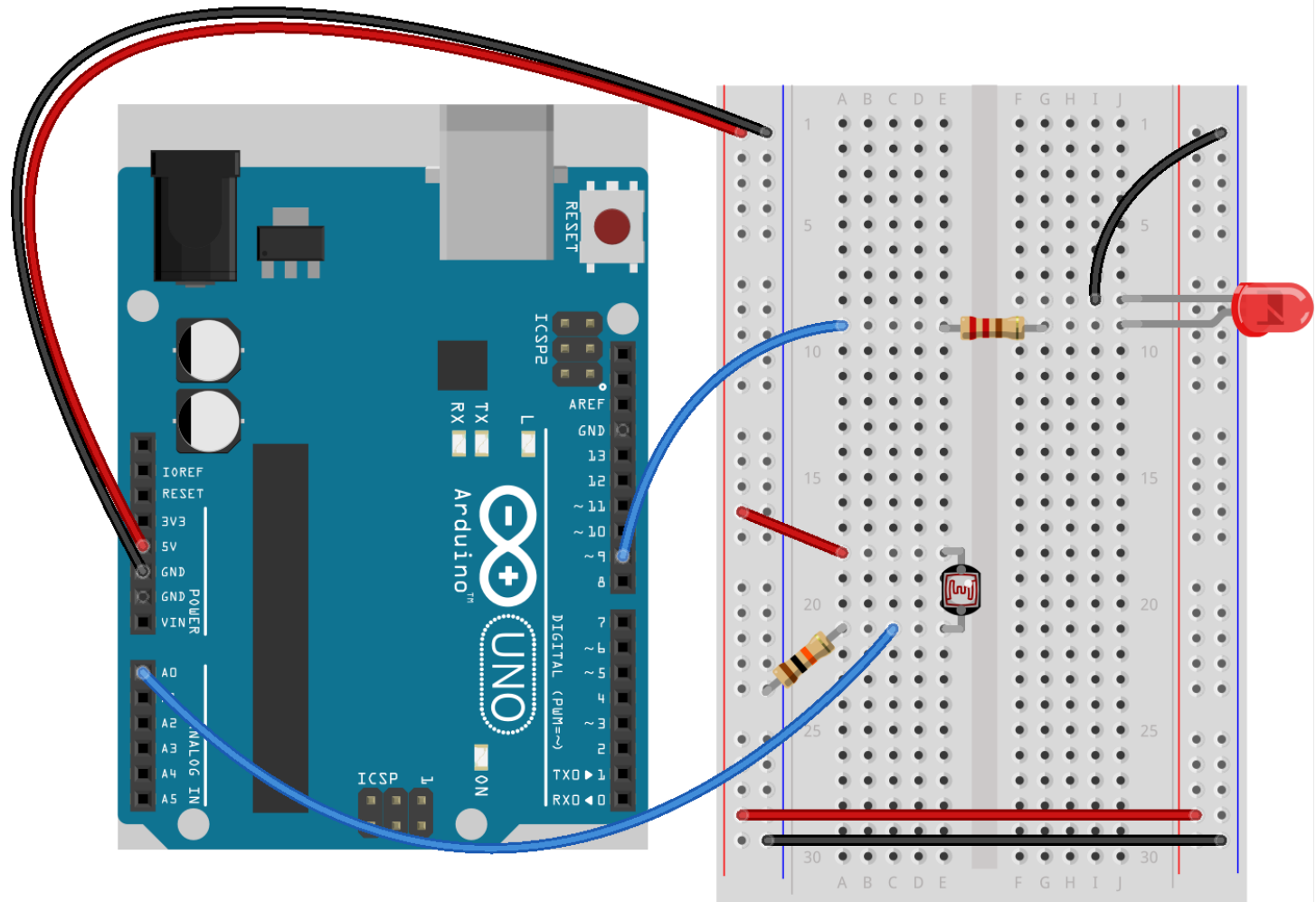
A screenshot of the Arduino IDE interface. The window title is 'analog\_in\_pot | Arduino 1.0.4'. The toolbar shows icons for checking, compiling, uploading, and downloading. The sketch name 'analog\_in\_pot' is displayed in the top bar. The code is as follows:

```
int ledPin = 9;
int analogValue = 0;
int brightness = 0;

void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // Value between 0 and 1023
  analogValue = analogRead(A0);
  brightness = analogValue / 4;
  analogWrite(ledPin, brightness);
  Serial.println(analogValue);
}
```

# Analog In and Out (Photo Resistor)



# Analog In and Out (Photo Resistor)

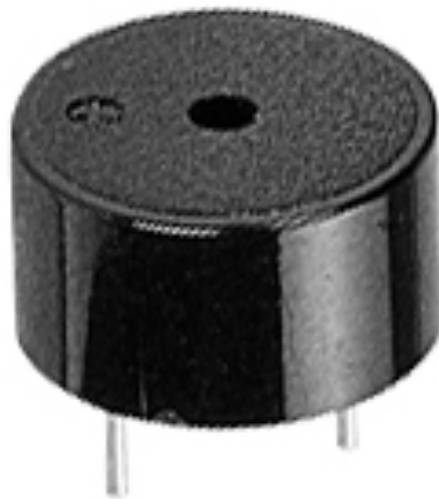
A screenshot of the Arduino IDE interface. The window title is 'analog\_in\_photo | Arduino 1.0.4'. The toolbar shows icons for checking, running, opening, uploading, and downloading. The file explorer on the left shows a folder named 'analog\_in\_photo'. The main editor area contains the following C++ code:

```
//Needs to be in a PWM output
int ledPin = 9;
int analogValue = 0;
int brightness = 0;

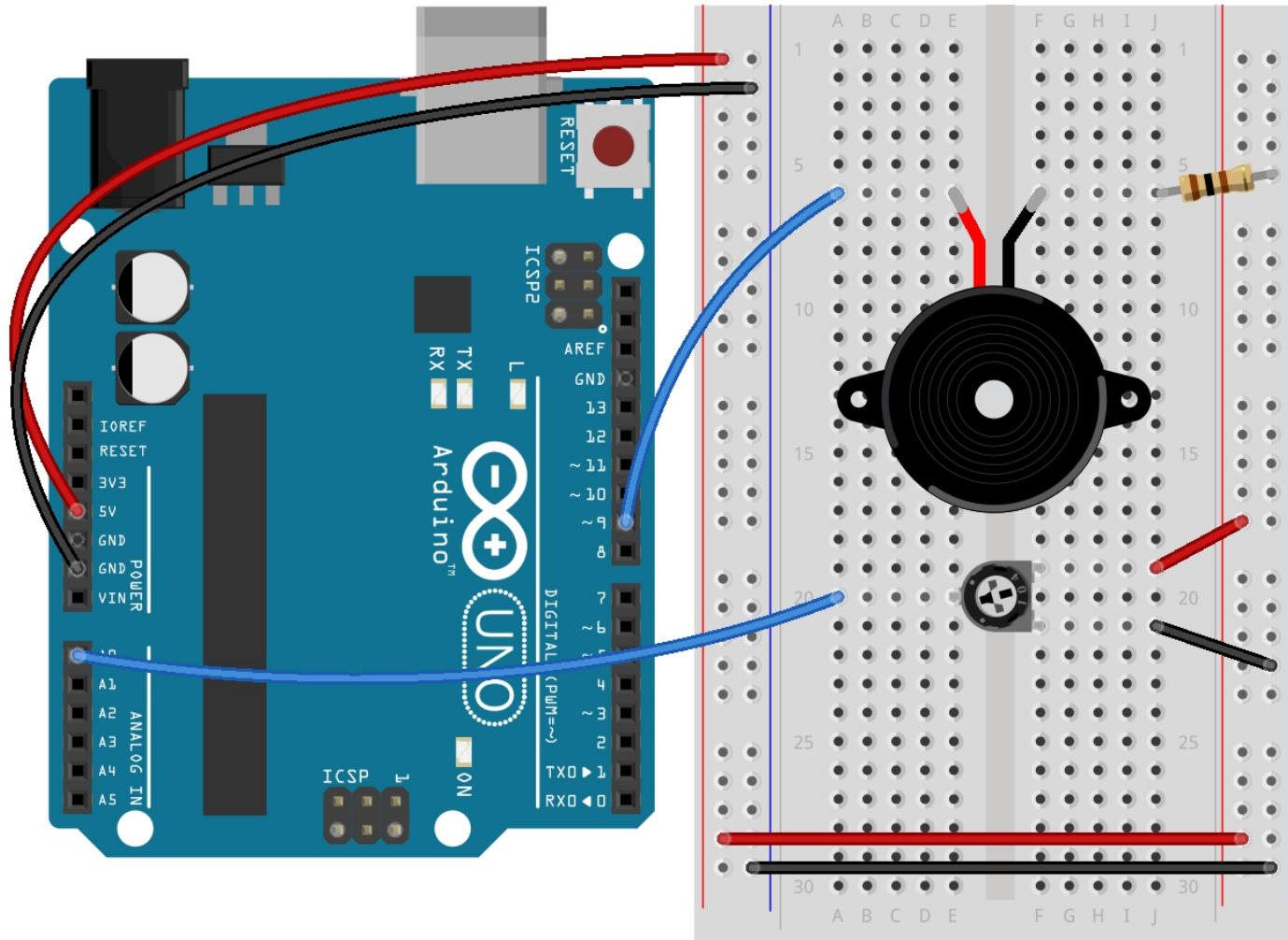
void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
}

void loop() {
  analogValue = analogRead(A0);
  brightness = map(analogValue, 300, 750, 0, 255);
  if (brightness < 0) brightness = 0;
  if (brightness > 255) brightness = 255;
  analogWrite(ledPin, 255 - brightness);
  Serial.println(analogValue);
}
```

# Tone

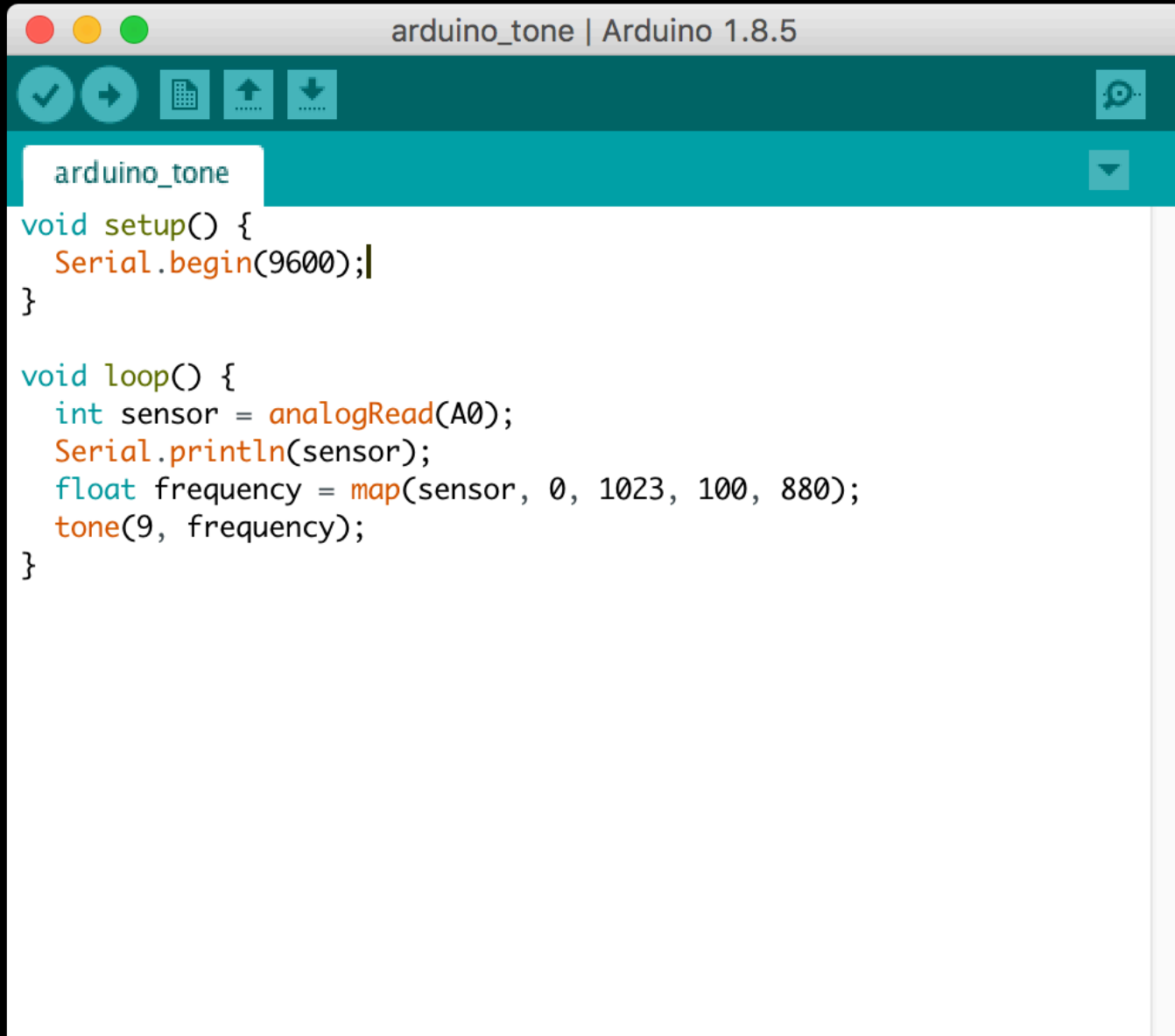


# Tone





# Tone



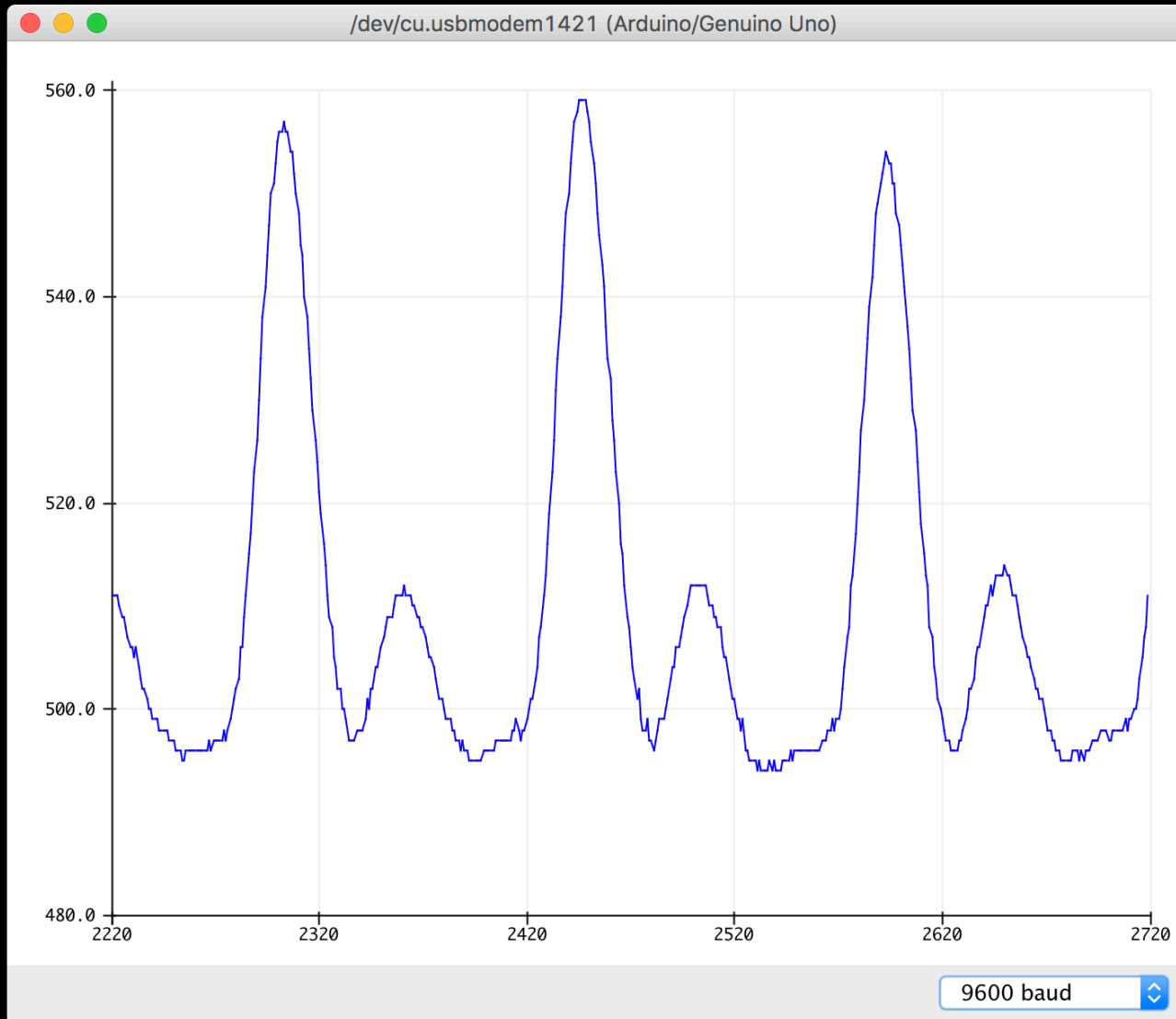
The screenshot shows the Arduino IDE interface. The title bar reads "arduino\_tone | Arduino 1.8.5". The toolbar includes icons for a checkmark, a right arrow, a document, an upload arrow, a download arrow, and a search icon. The file explorer on the left shows a single file named "arduino\_tone". The main editor area contains the following C++ code:

```
void setup() {  
  Serial.begin(9600);  
}  
  
void loop() {  
  int sensor = analogRead(A0);  
  Serial.println(sensor);  
  float frequency = map(sensor, 0, 1023, 100, 880);  
  tone(9, frequency);  
}
```

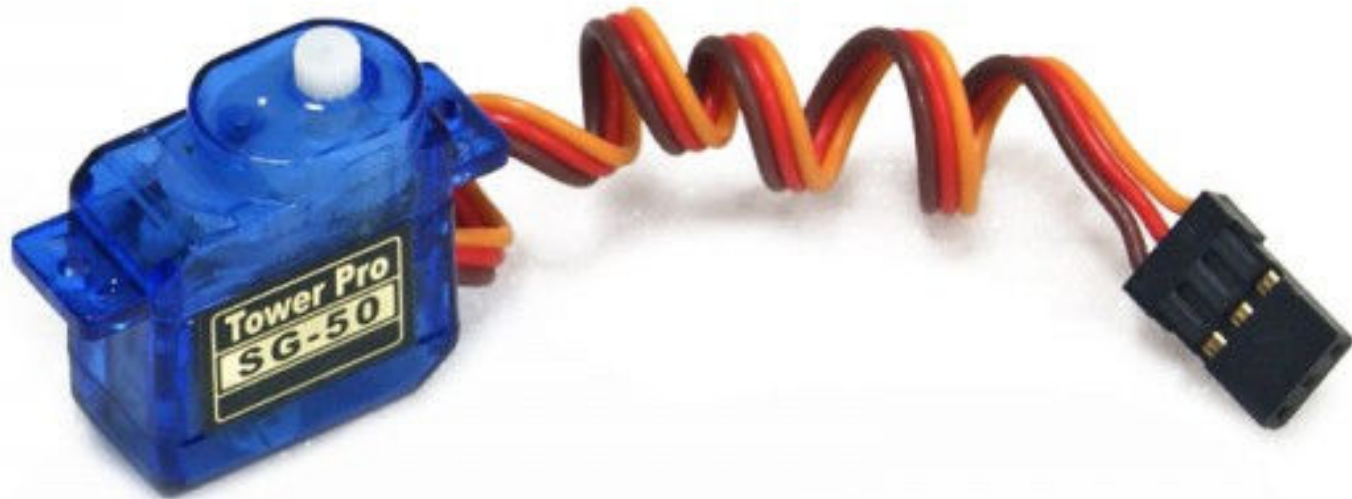
# Pulse Sensor



# Serial Plotter



# Servo Motor

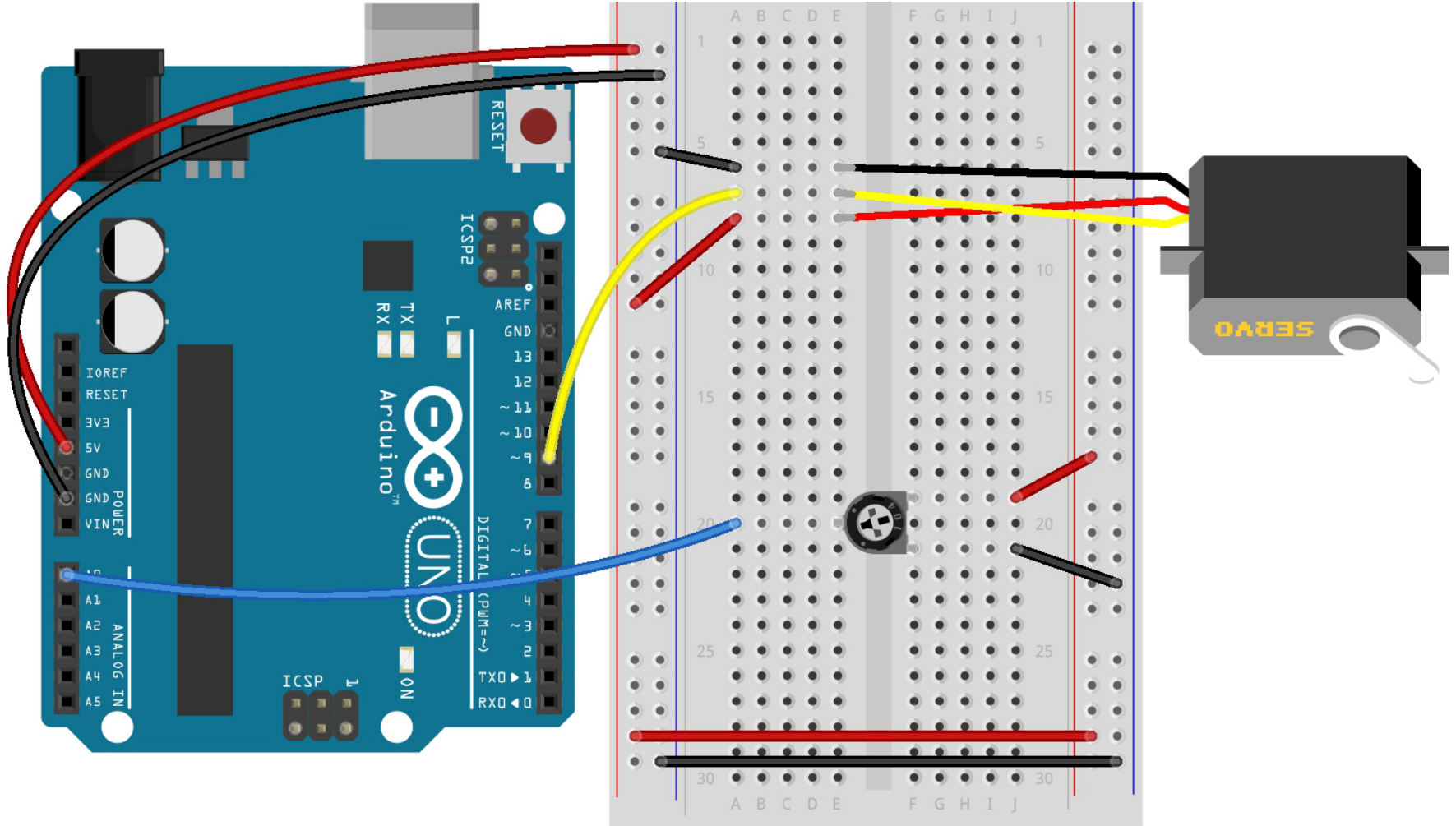


# Servo Motor

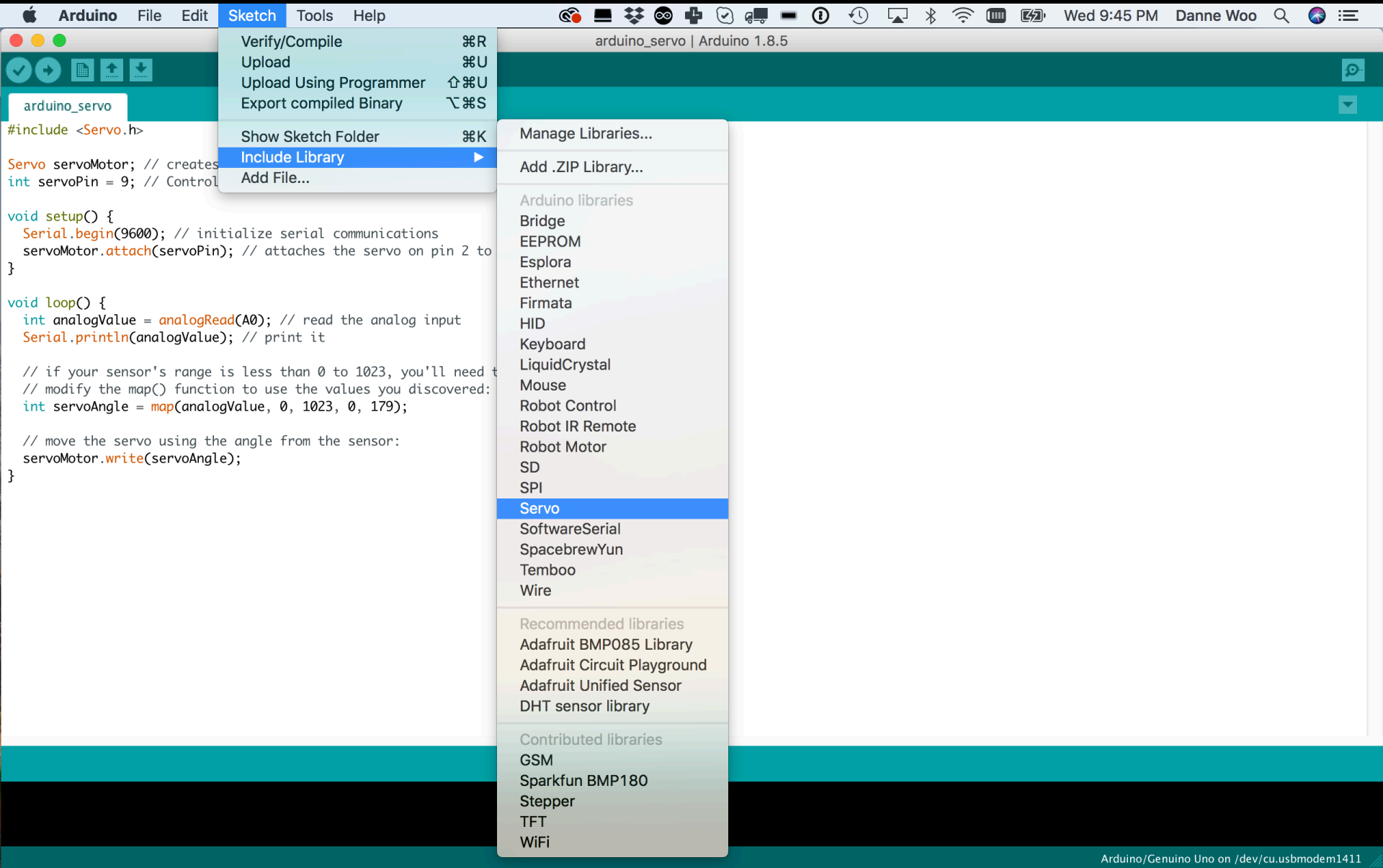




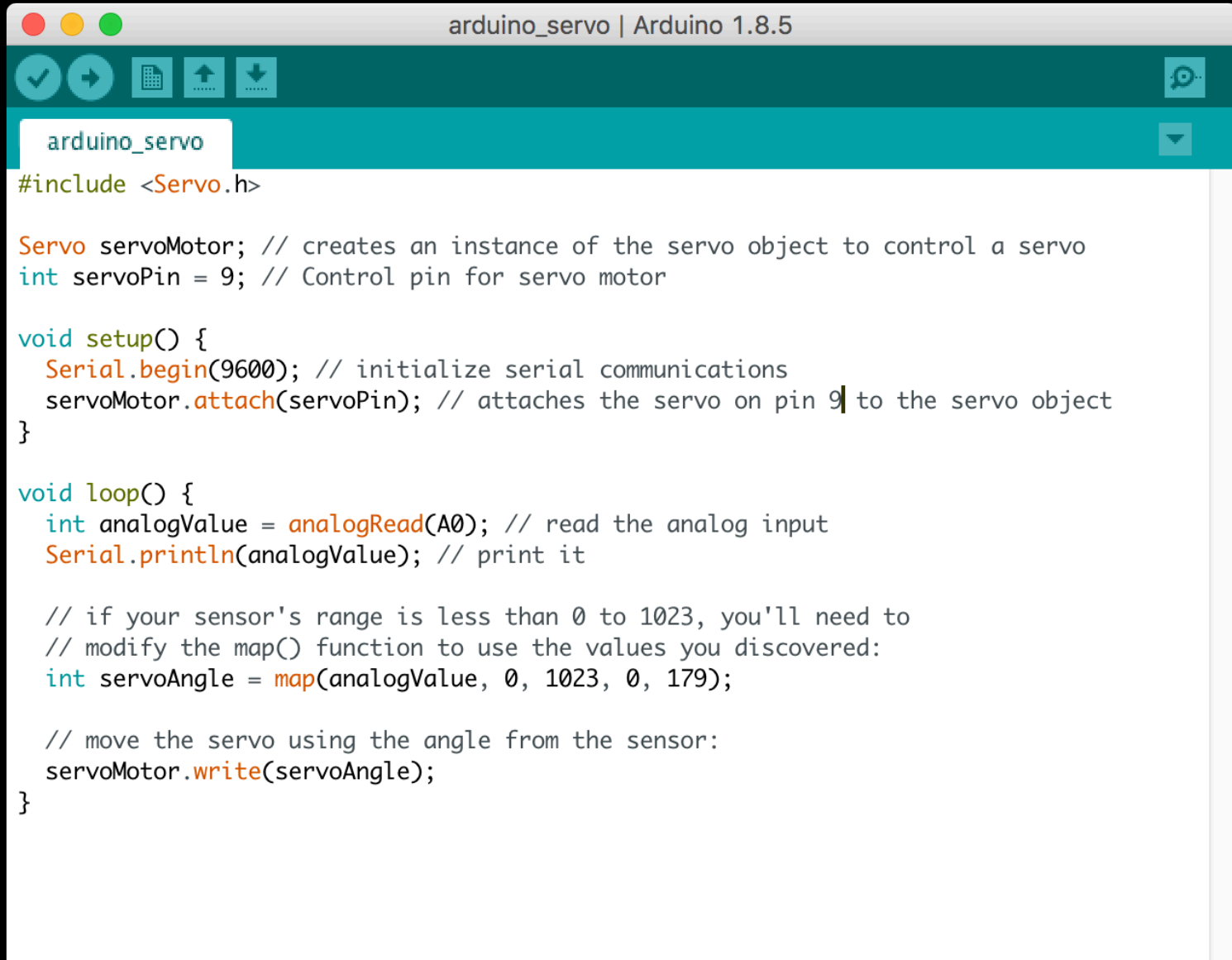
# Servo Motor



# Servo Motor (Libraries)



# Servo Motor



The image shows a screenshot of the Arduino IDE interface. The title bar at the top reads "arduino\_servo | Arduino 1.8.5". Below the title bar is a toolbar with icons for checking, running, serial monitor, and file operations. The main text area contains the following C++ code:

```
arduino_servo
#include <Servo.h>

Servo servoMotor; // creates an instance of the servo object to control a servo
int servoPin = 9; // Control pin for servo motor

void setup() {
  Serial.begin(9600); // initialize serial communications
  servoMotor.attach(servoPin); // attaches the servo on pin 9 to the servo object
}

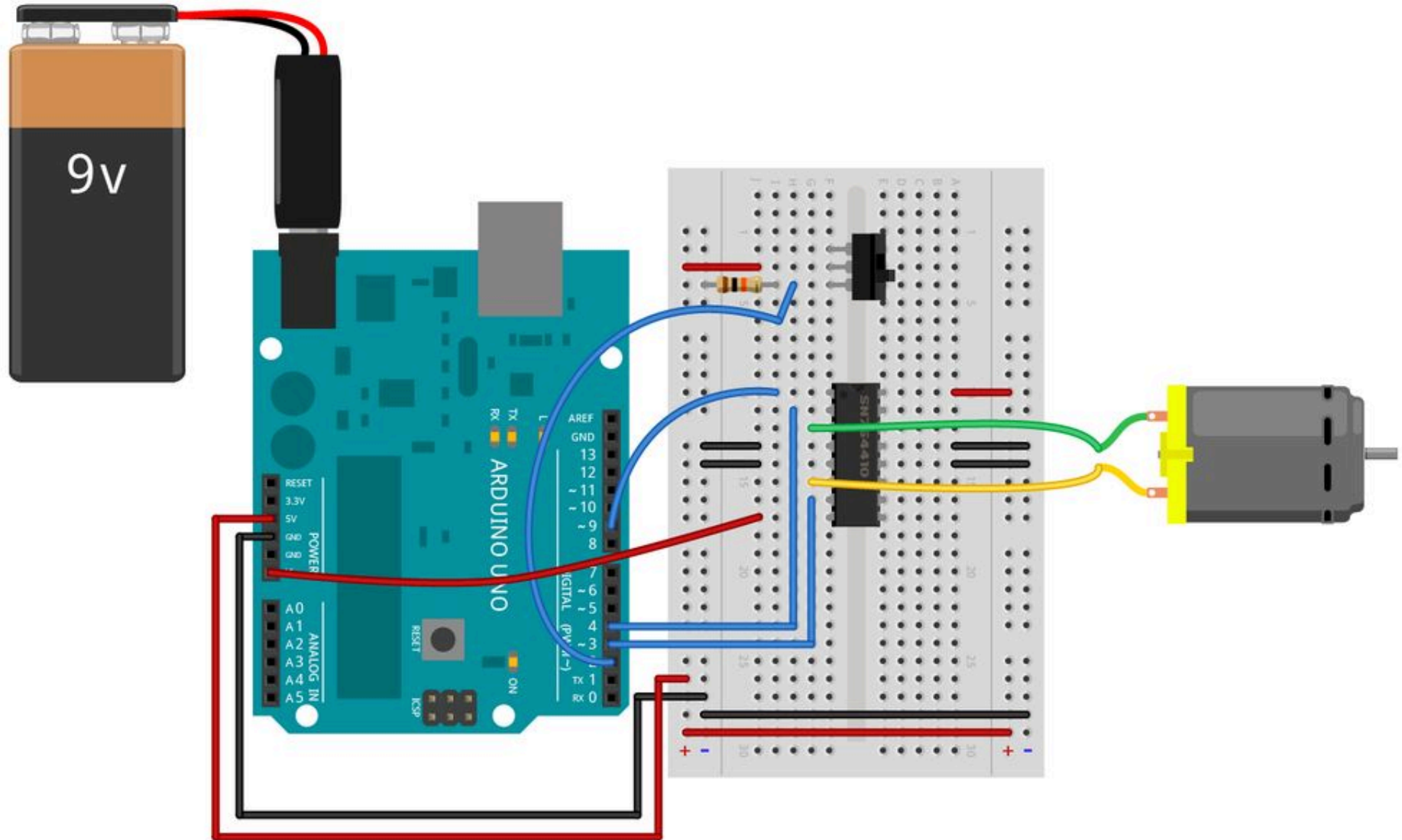
void loop() {
  int analogValue = analogRead(A0); // read the analog input
  Serial.println(analogValue); // print it

  // if your sensor's range is less than 0 to 1023, you'll need to
  // modify the map() function to use the values you discovered:
  int servoAngle = map(analogValue, 0, 1023, 0, 179);

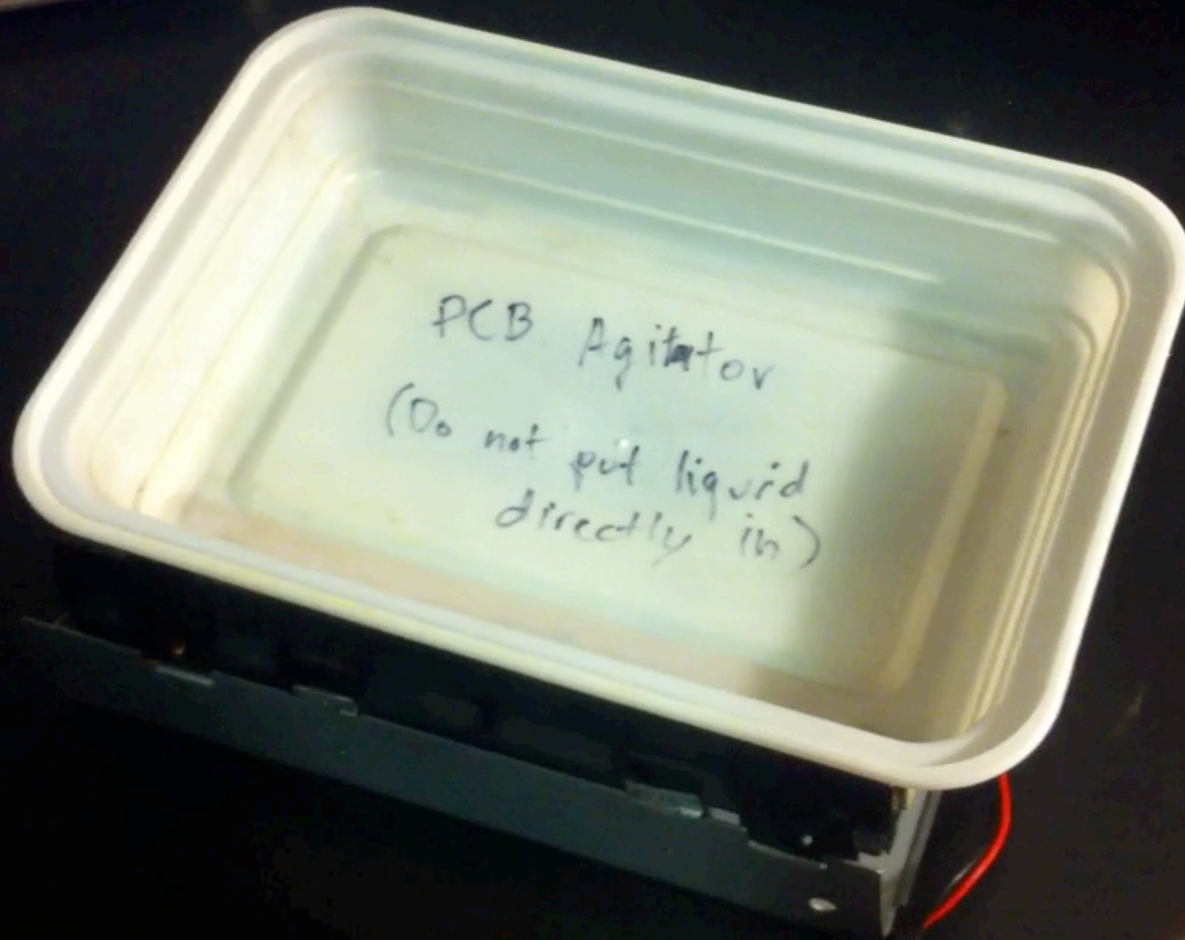
  // move the servo using the angle from the sensor:
  servoMotor.write(servoAngle);
}
```



# DC Motor H-Bridge

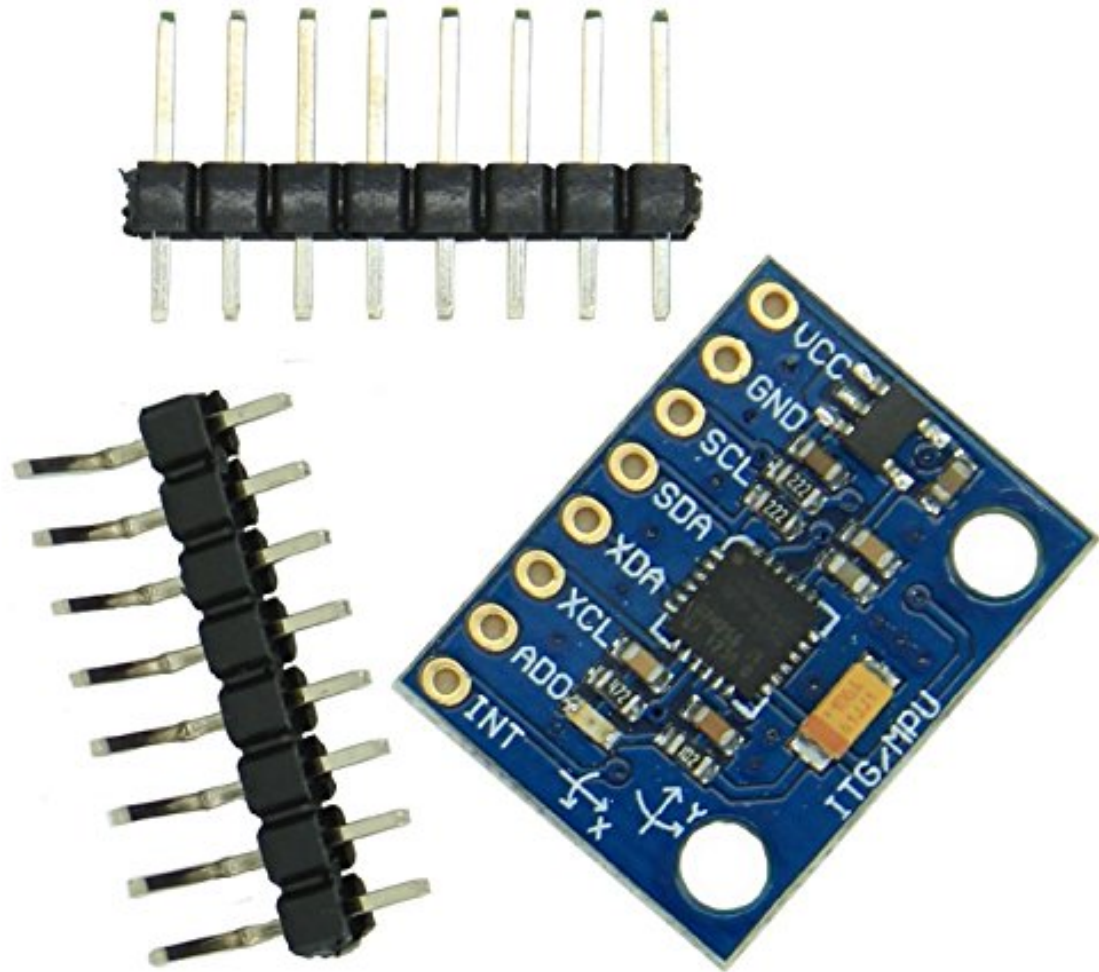


# DC Motor H-Bridge



# I2C Communications

VCC > 5v  
GND > Ground  
SCL > A5 (UNO)  
SDA > A4 (UNO)




# Wire Library

Arduino - Wire

Secure | https://www.arduino.cc/en/Reference/Wire

Apps Bookmarks Personal ITP Scripts Materials Design DV CUNY CRISPR COC Other Bookmarks

 ARDUINO

HOME BUY SOFTWARE PRODUCTS EDUCATION RESOURCES COMMUNITY HELP

Wire Library

This library allows you to communicate with I2C / TWI devices. On the Arduino boards with the R3 layout (1.0 pinout), the SDA (data line) and SCL (clock line) are on the pin headers close to the AREF pin. The Arduino Due has two I2C / TWI interfaces SDA1 and SCL1 are near to the AREF pin and the additional one is on pins 20 and 21.

As a reference the table below shows where TWI pins are located on various Arduino boards.

Board	I2C / TWI pins
Uno, Ethernet	A4 (SDA), A5 (SCL)
Mega2560	20 (SDA), 21 (SCL)
Leonardo	2 (SDA), 3 (SCL)
Due	20 (SDA), 21 (SCL), SDA1, SCL1

As of Arduino 1.0, the library inherits from the Stream functions, making it consistent with other read/write libraries. Because of this, send() and receive() have been replaced with read() and write().

Functions

- [begin\(\)](#)
- [requestFrom\(\)](#)
- [beginTransaction\(\)](#)
- [endTransmission\(\)](#)
- [write\(\)](#)
- [available\(\)](#)
- [read\(\)](#)
- [SetClock\(\)](#)
- [onReceive\(\)](#)
- [onRequest\(\)](#)

# In Class/Homework

Review the labs that were covered in class this week, and come up with a simple yet creative application using analog output. Post your experiments to your blog.

# **Physical Computing**

Professor Danne Woo

[dwoo@qc.cuny.edu](mailto:dwoo@qc.cuny.edu)

[pcomp.dannewoo.com](http://pcomp.dannewoo.com)