# 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
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

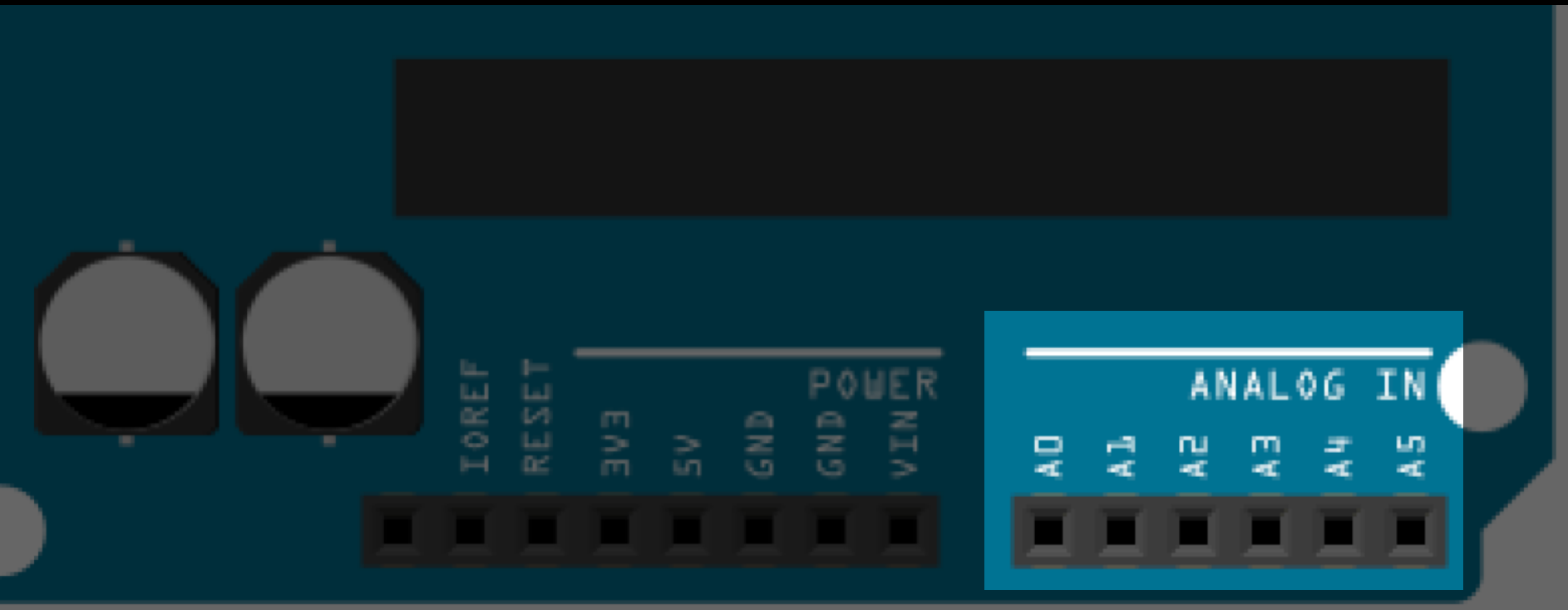
Week 6: Enclosures

Week 7: Serial Communication, Processing and p5.js

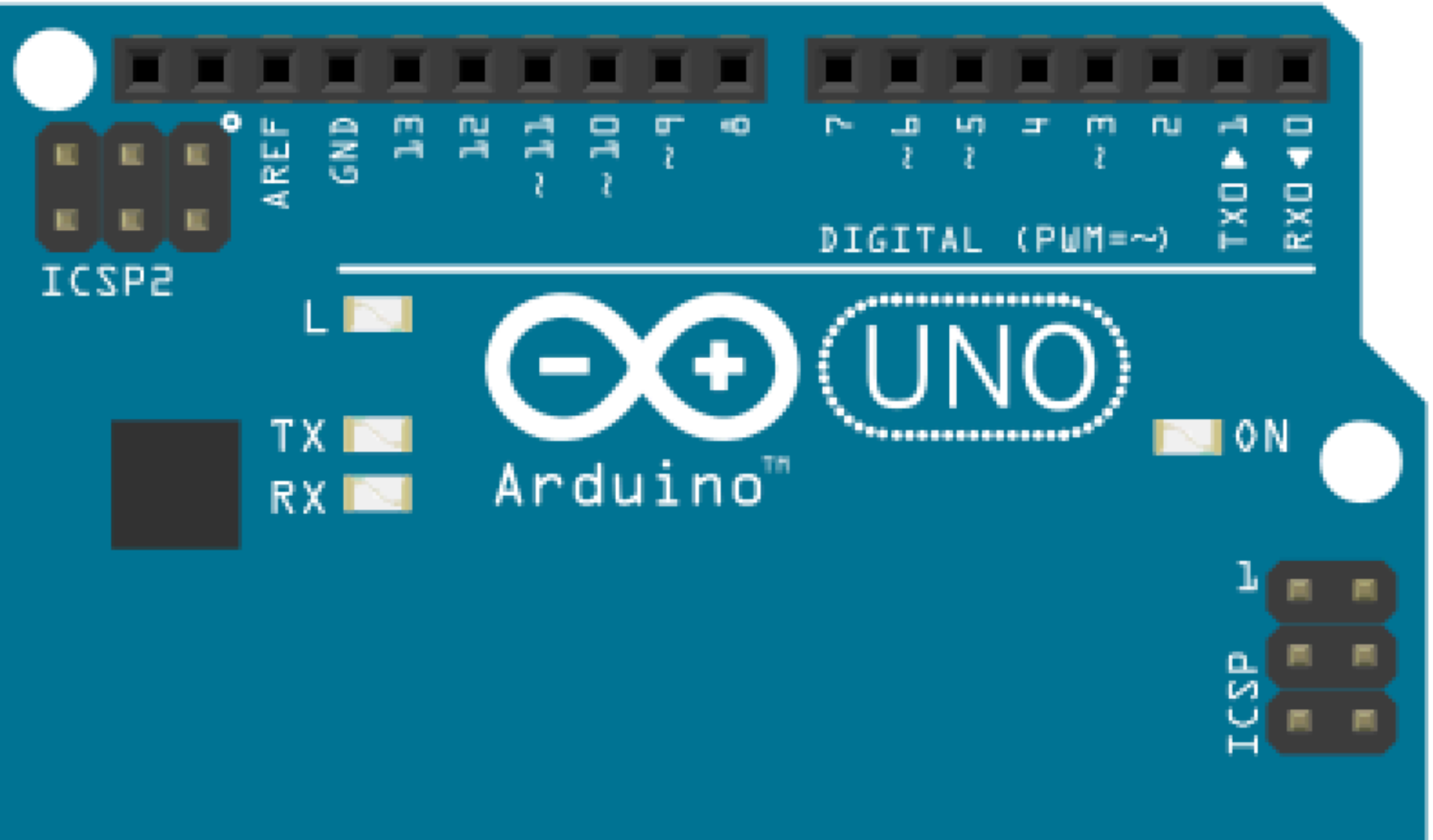
Week 8: Soldiering Workshop

Week 9: Midterm Presentation

# Analog In

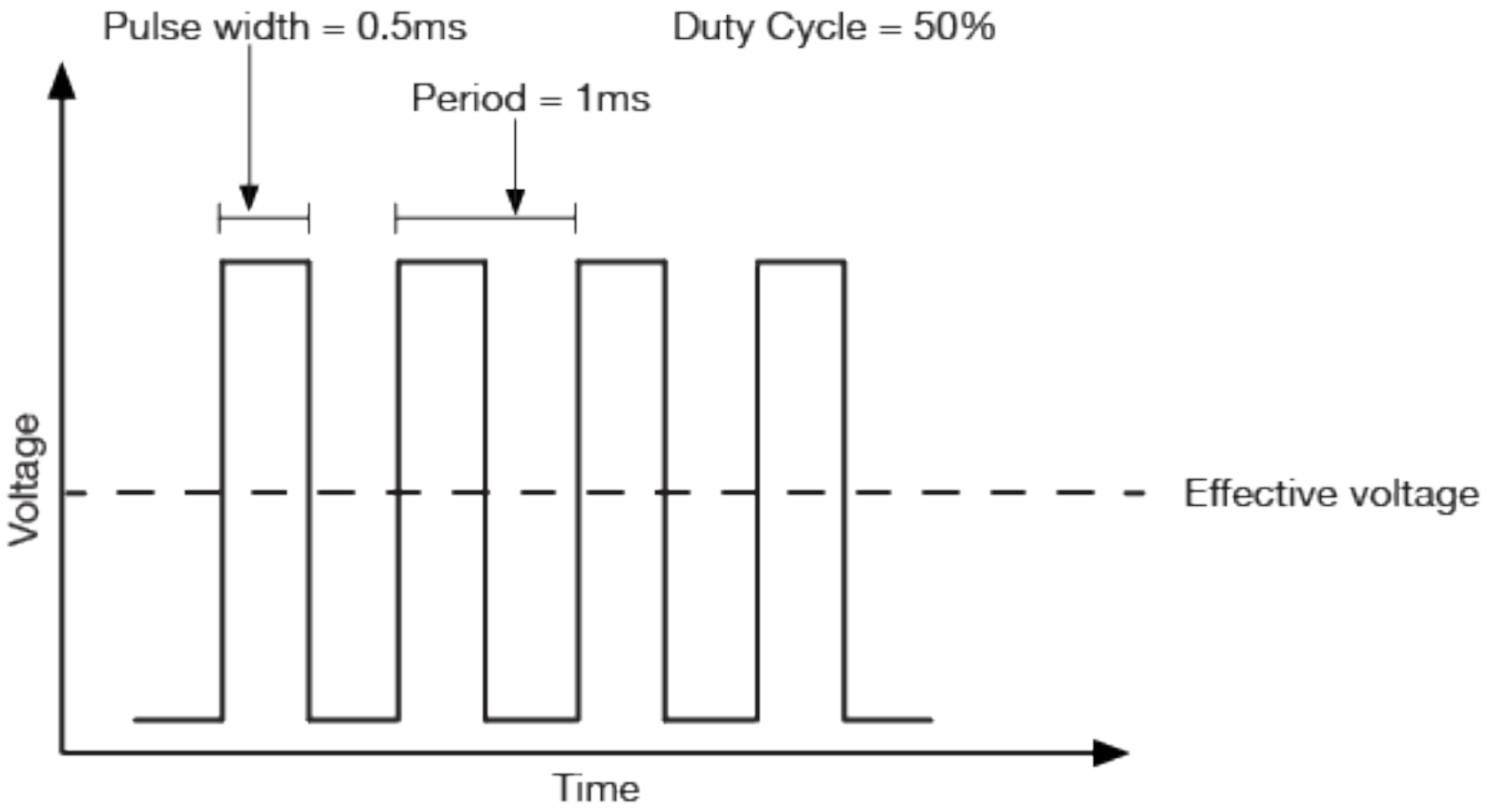


# Analog Out

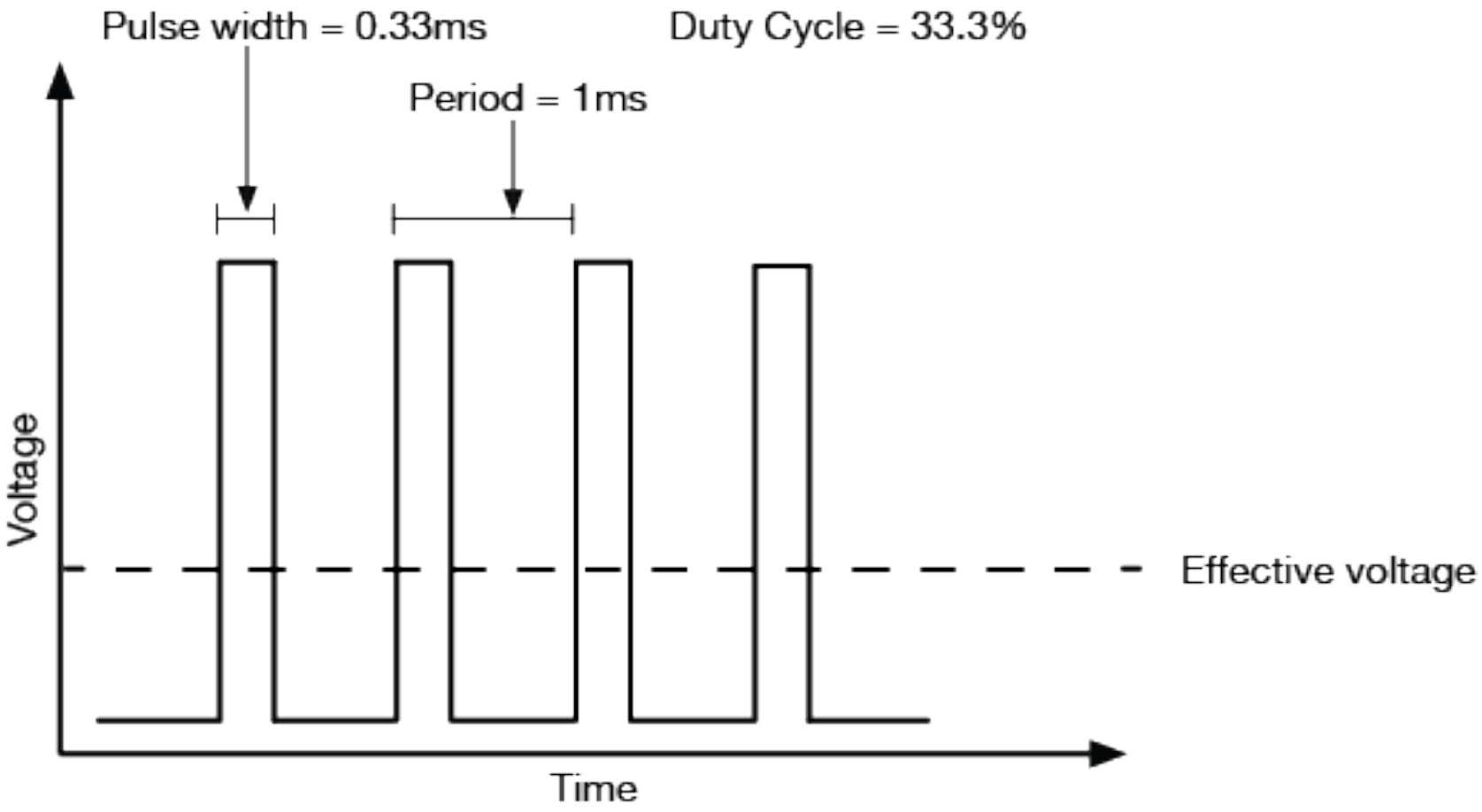

AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM=~)
ICSP2
L
TX
RX
Arduino™
UNO
ON
ICSP
1

# Pulse Width Modulation (Fake Analog)

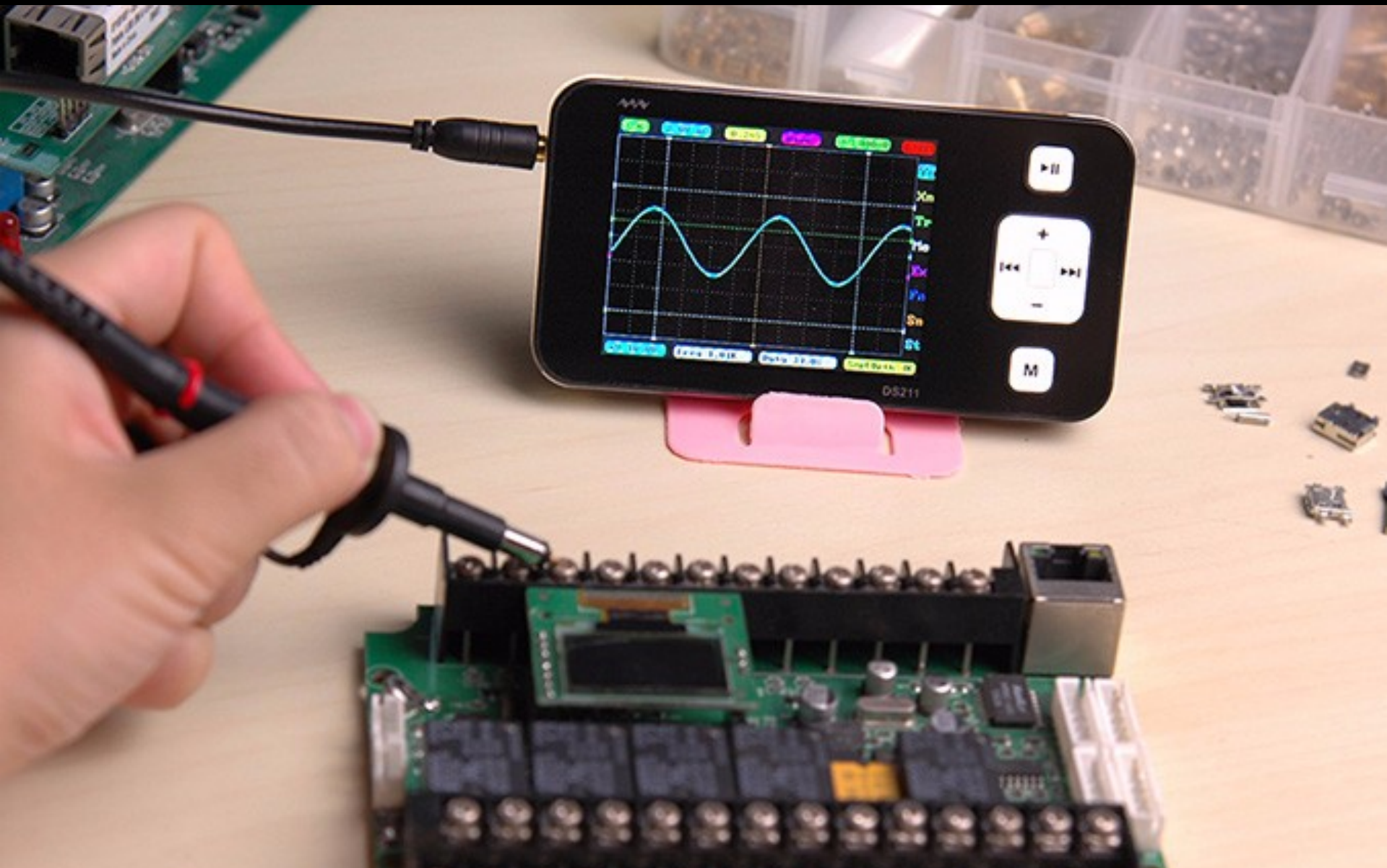
# Pulse Width Modulation (Fake Analog)



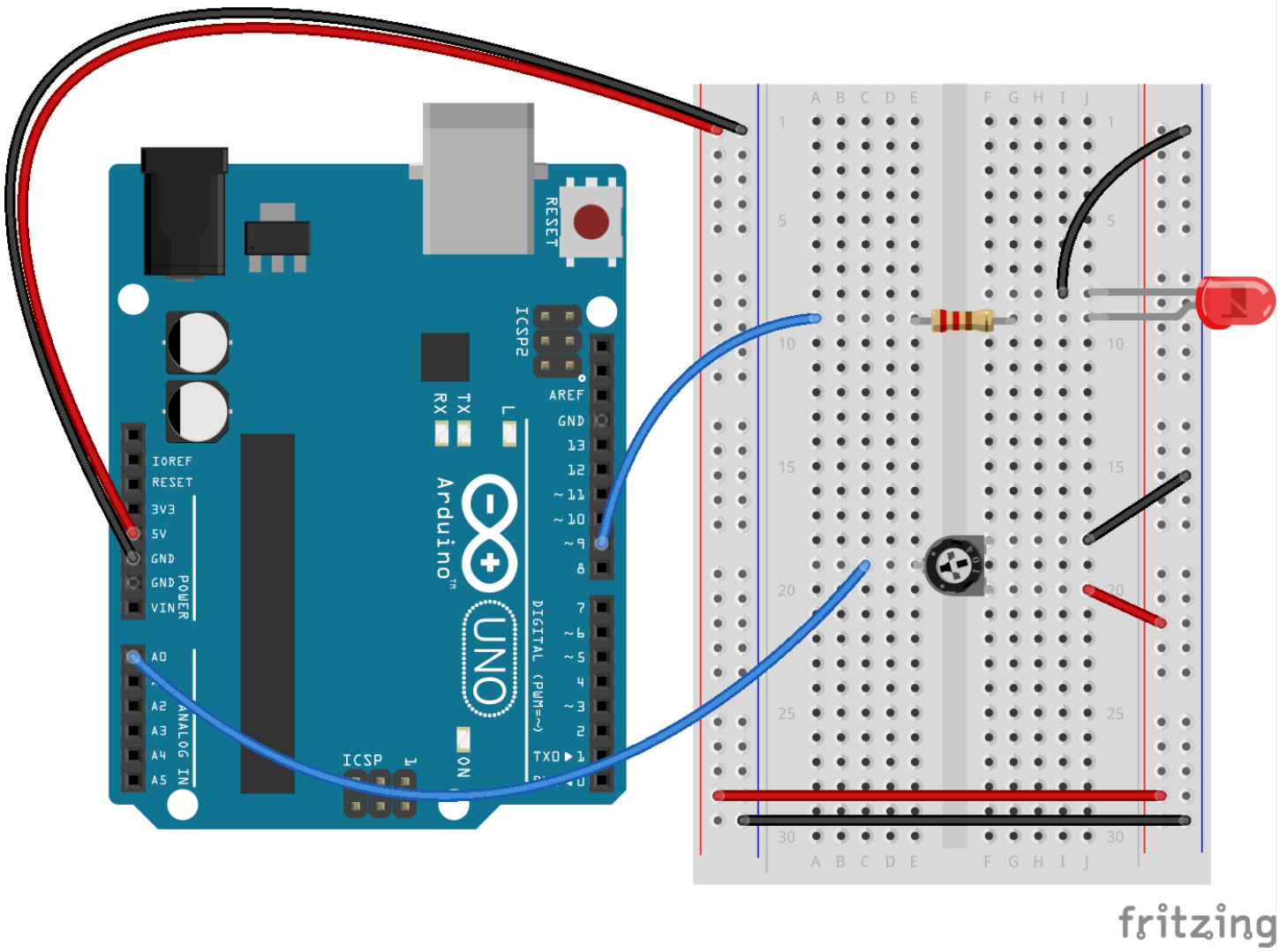
# Pulse Width Modulation (Fake Analog)



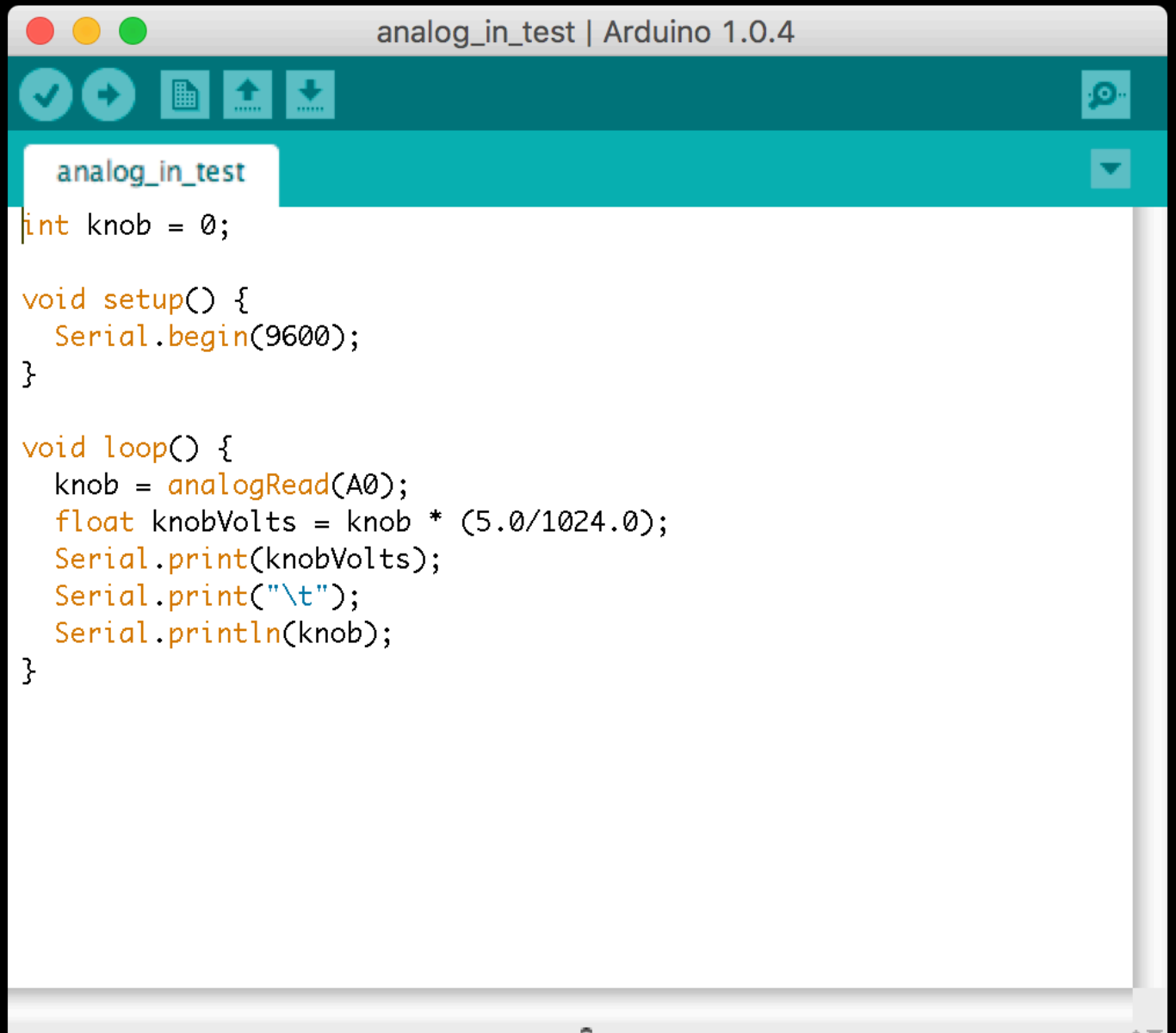
# Oscilloscope



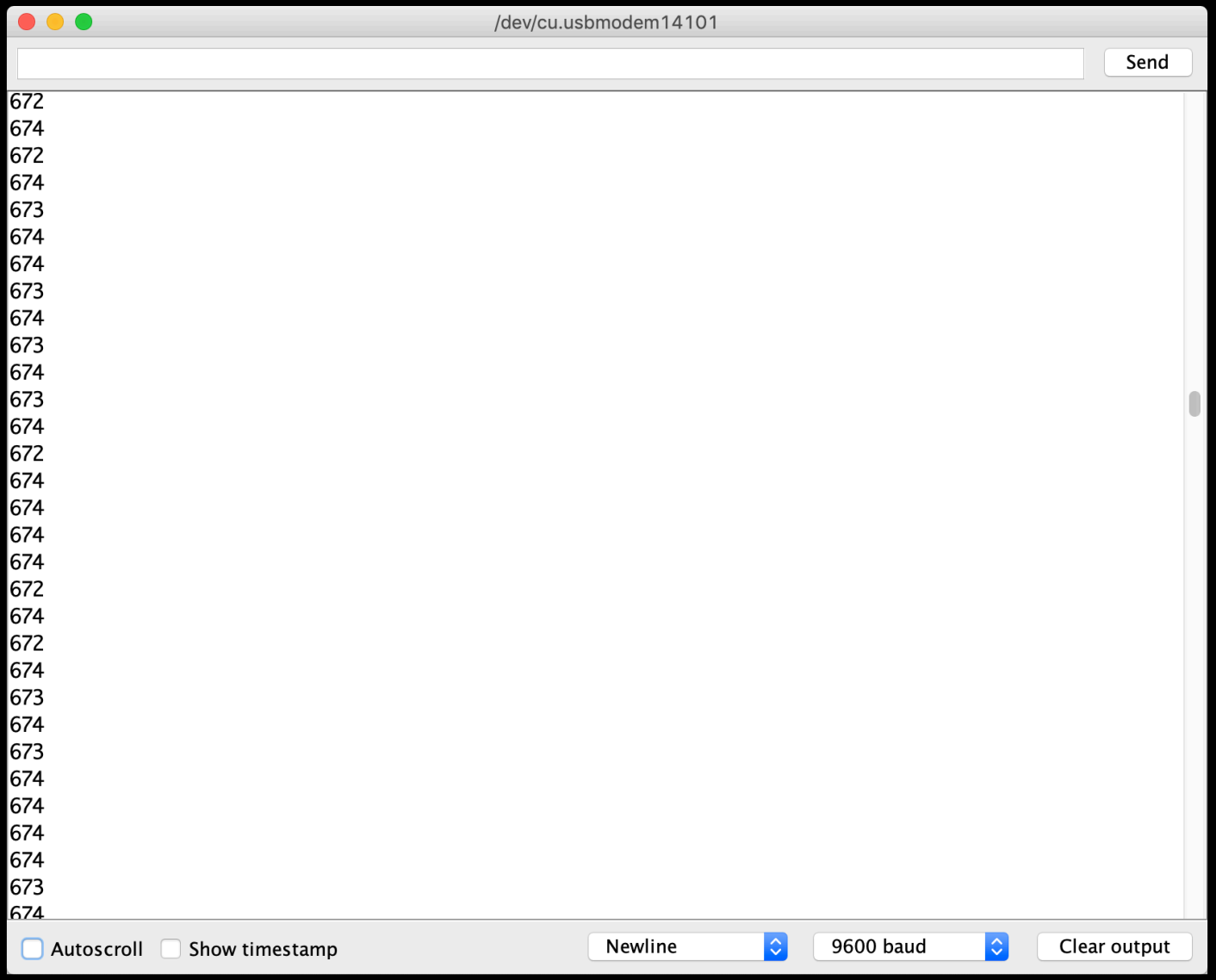
# Analog In and Out (Potentiometer)



# Analog In – Test Sensor Values

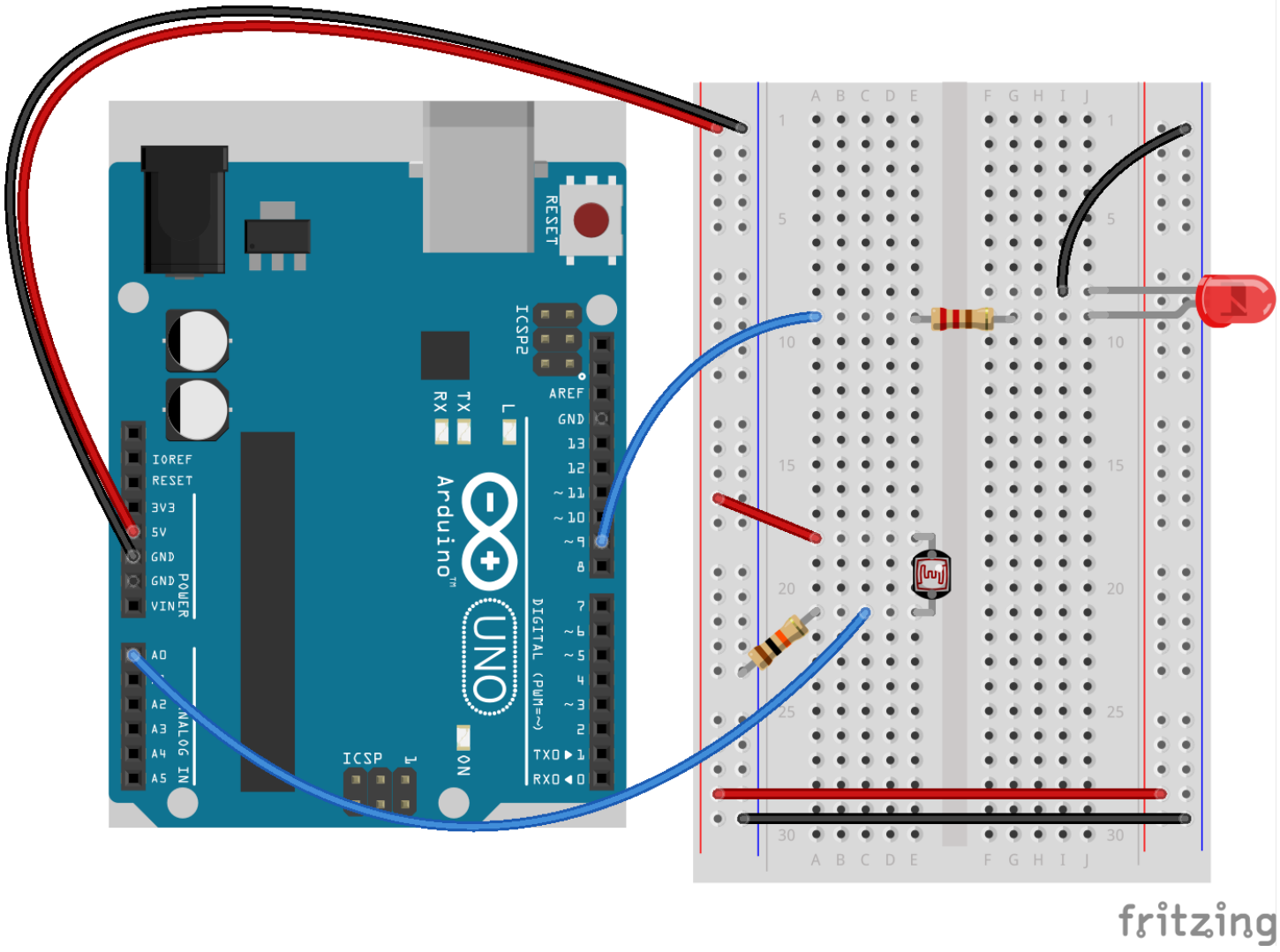


# Analog In – Serial Monitor



# Analog In and Out (Potentiometer)

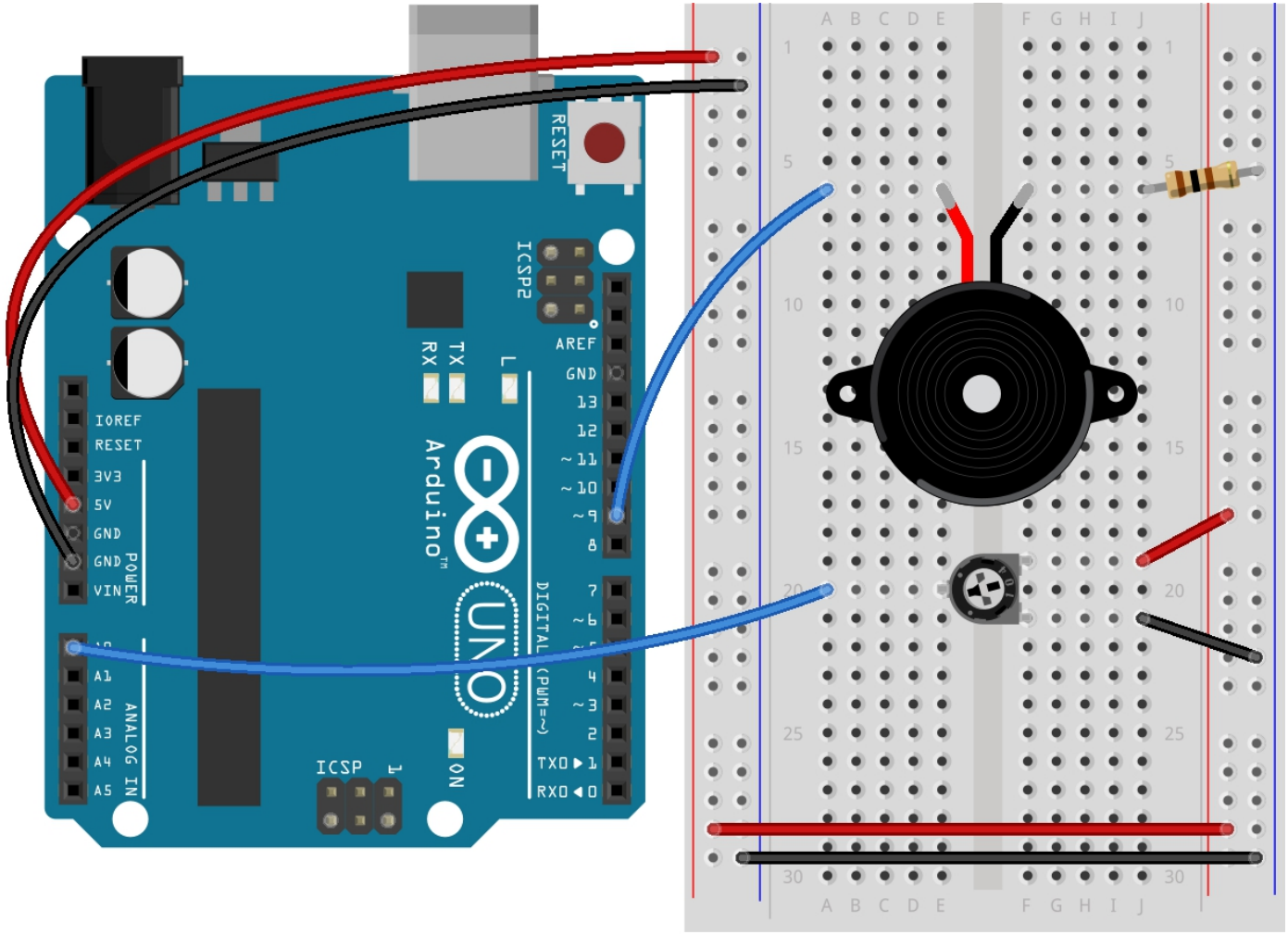
# Analog In and Out (Photo Resistor)



# Analog In and Out (Photo Resistor)

# Tone

# Tone

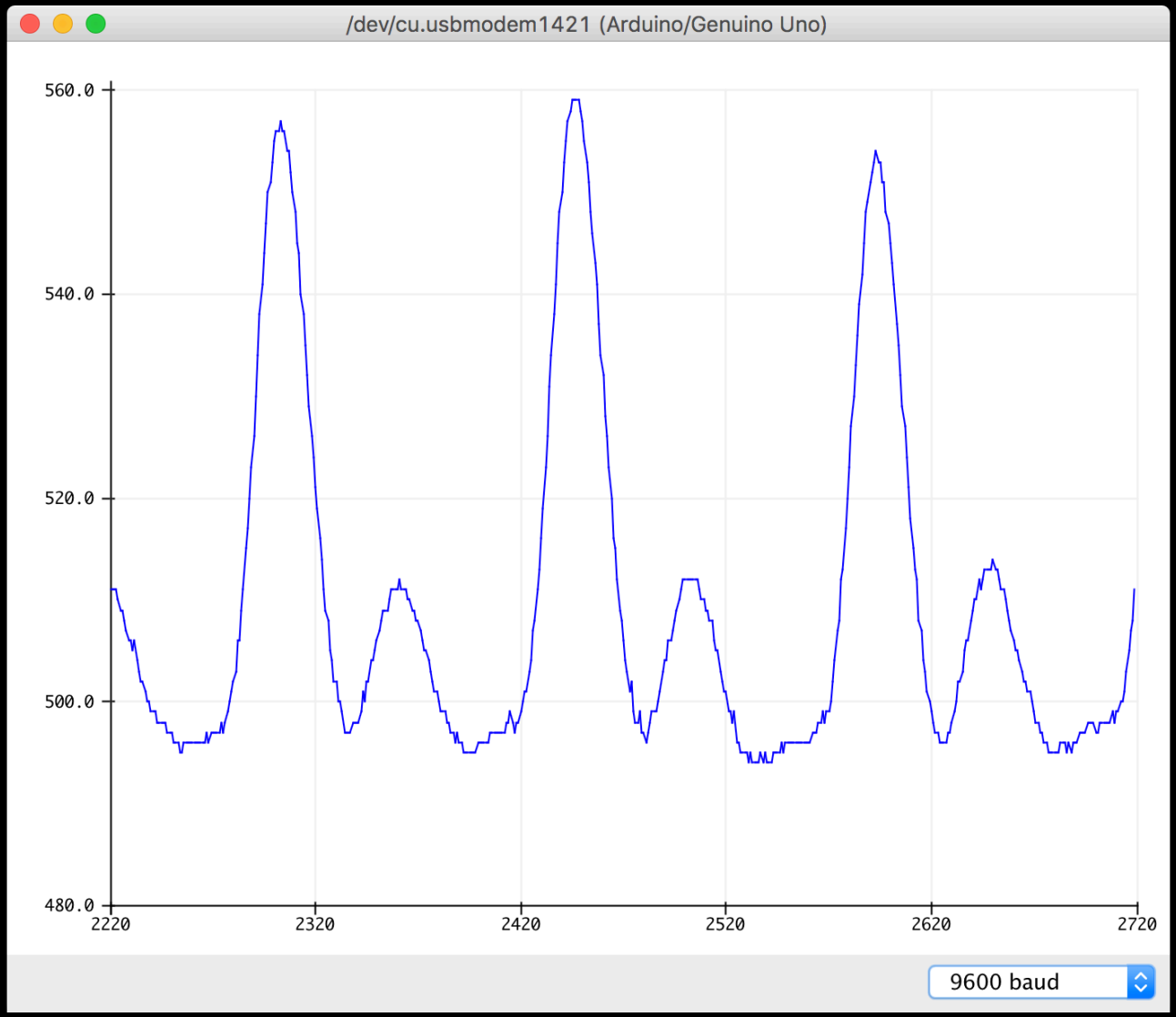

RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TXD▶1
RXD◀0
DIGITAL (PWM=~)
TX
RX
L
Arduino™
UNO
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
A1
A2
A3
A4
A5
ANALOG IN
A B C D E
F G H I J
1
5
10
15
20
25
30

# Tone

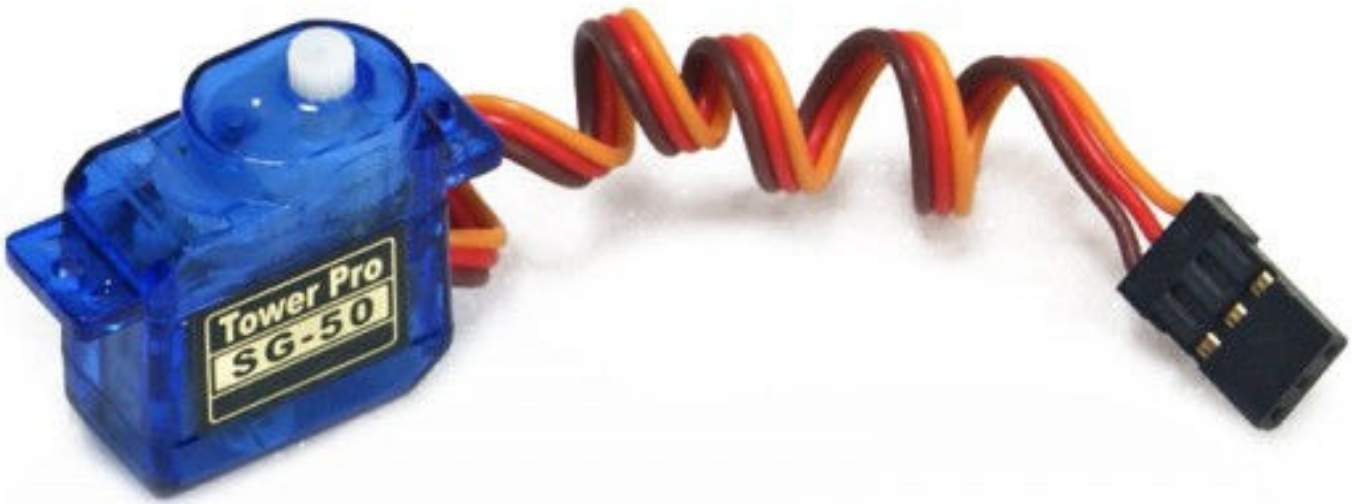

arduino_tone | Arduino 1.8.5
arduino_tone
void setup() {
Serial.begin(9600);
}
void loop() {
int sensor = analogRead(A0);
Serial.println(sensor);
float frequency = map(sensor, 0, 1023, 100, 880);
tone(9, frequency);
}

# Pulse Sensor

# Serial Plotter

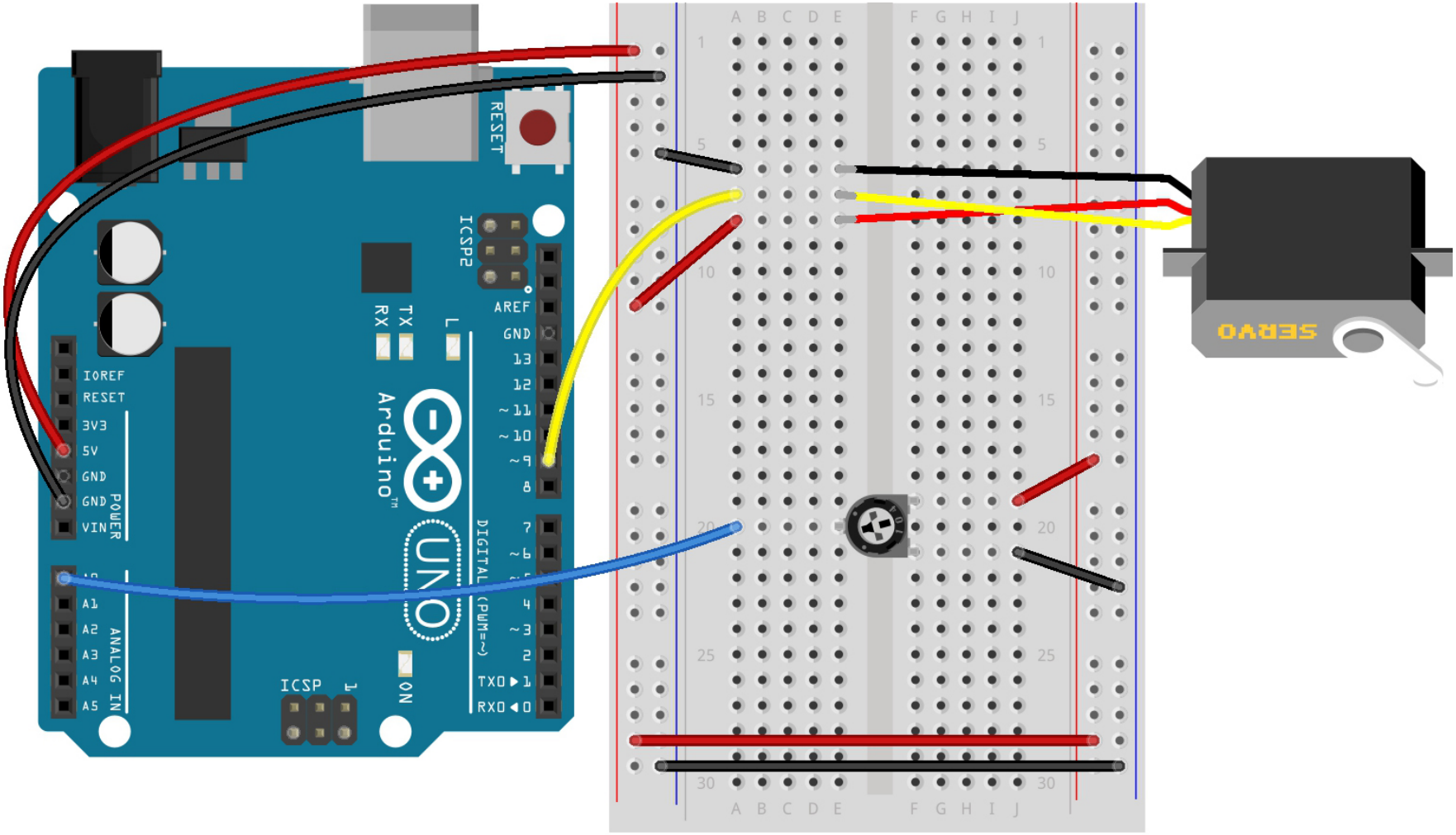


# Servo Motor

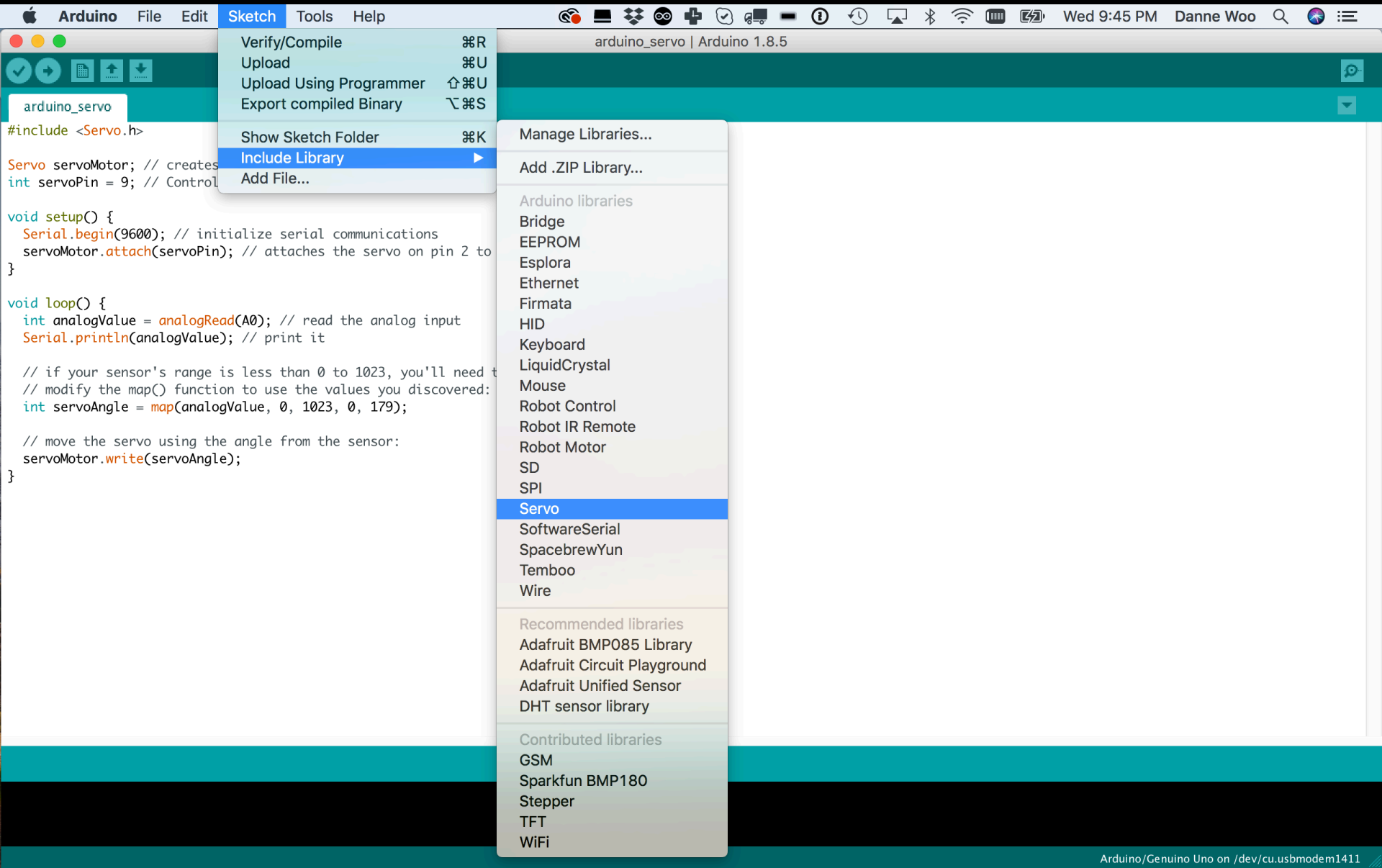


# Servo Motor

# Servo Motor


RESET
ICSP2
AREF
GND
13
12
~ 11
~ 10
~ 9
8
7
~ 6
4
~ 3
2
TXO ▶ 1
RXO ◀ 0
DIGITAL (PWM=~)
TX
RX
L
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
A1
A2
A3
A4
A5
ANALOG IN
Arduino™
UNO
A B C D E
F G H I J
1
5
10
15
20
25
30
SERVO

# Servo Motor (Libraries)



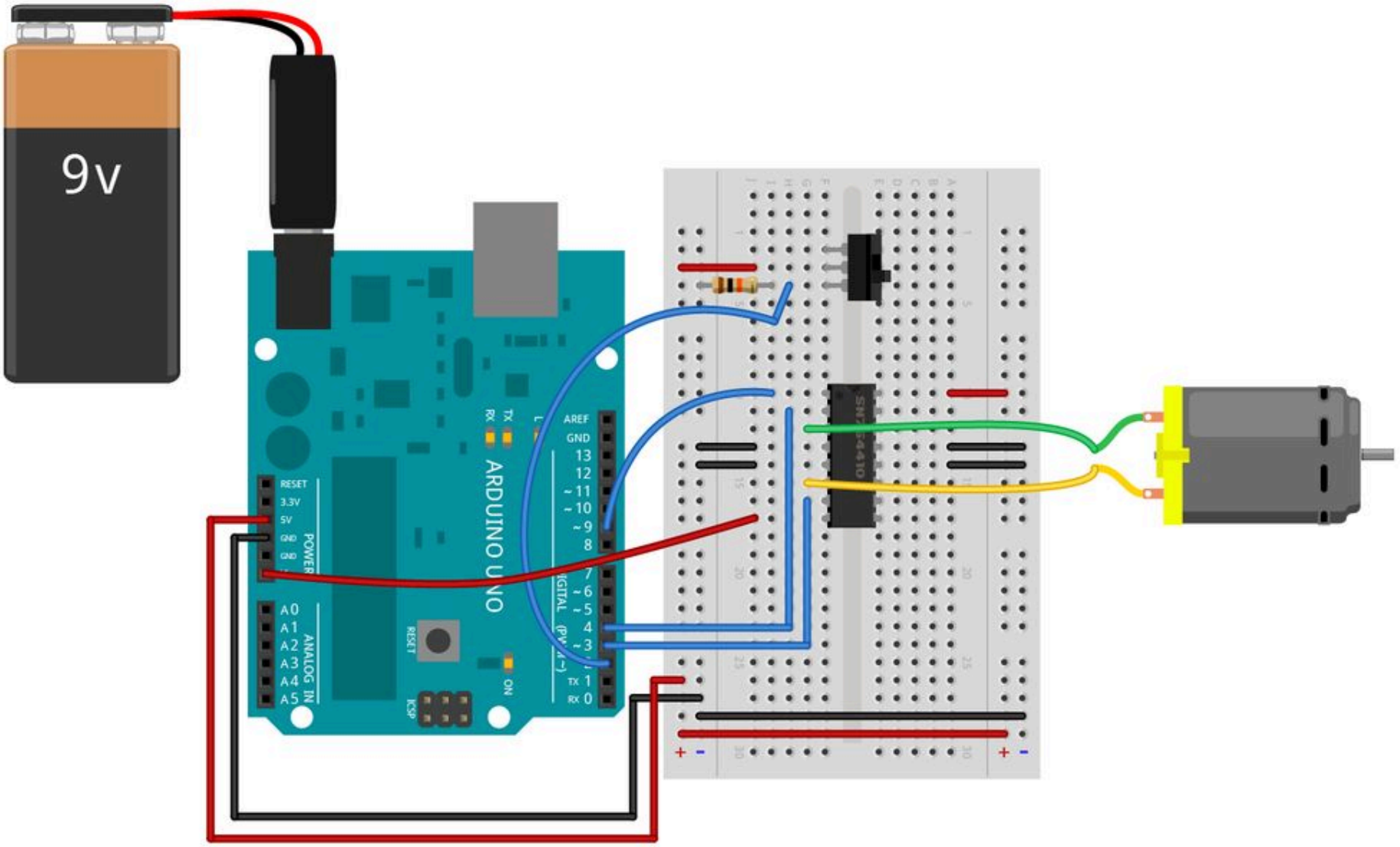
# Servo Motor

arduino_servo | Arduino 1.8.5

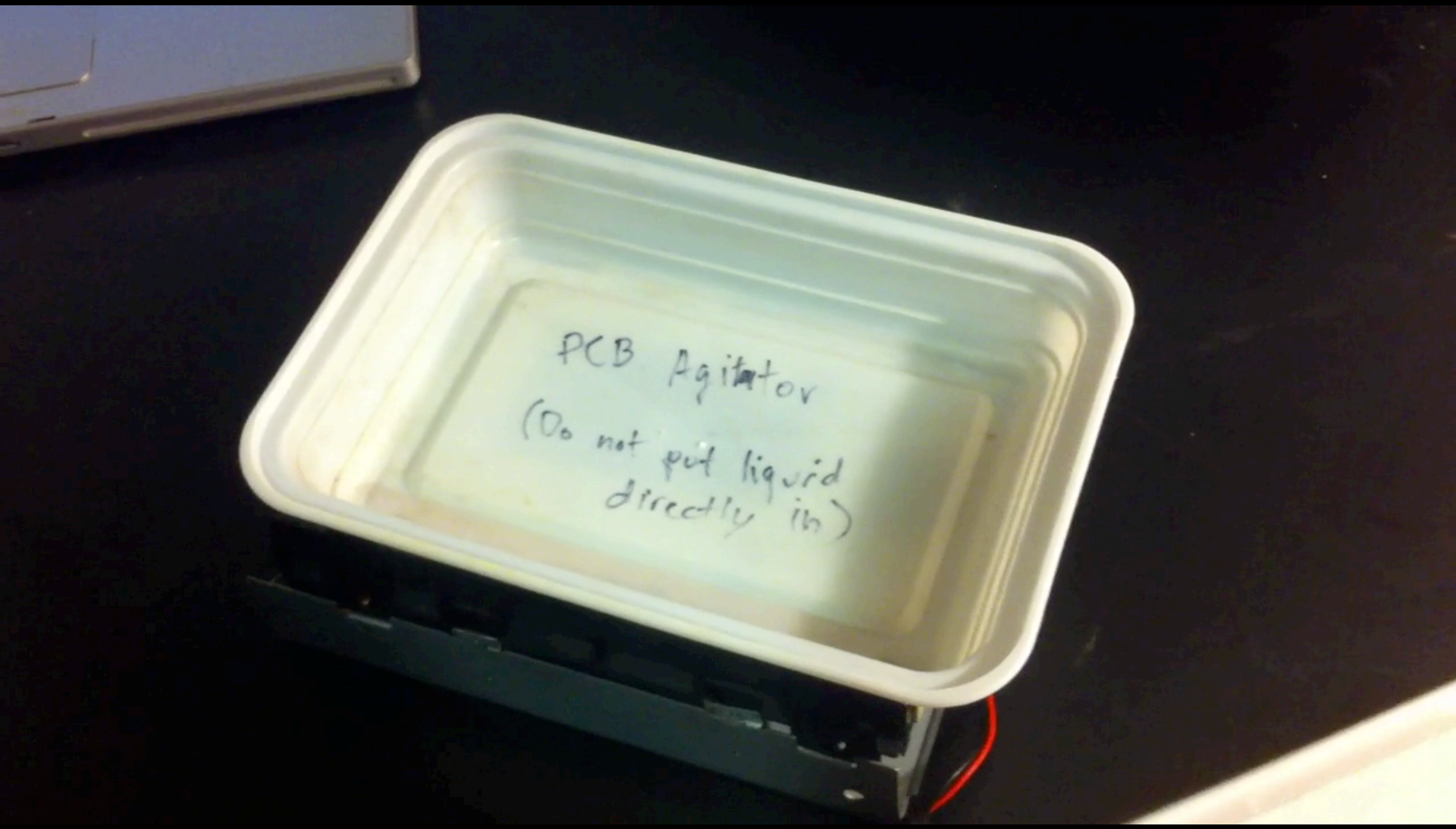
arduino_servo

```
#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

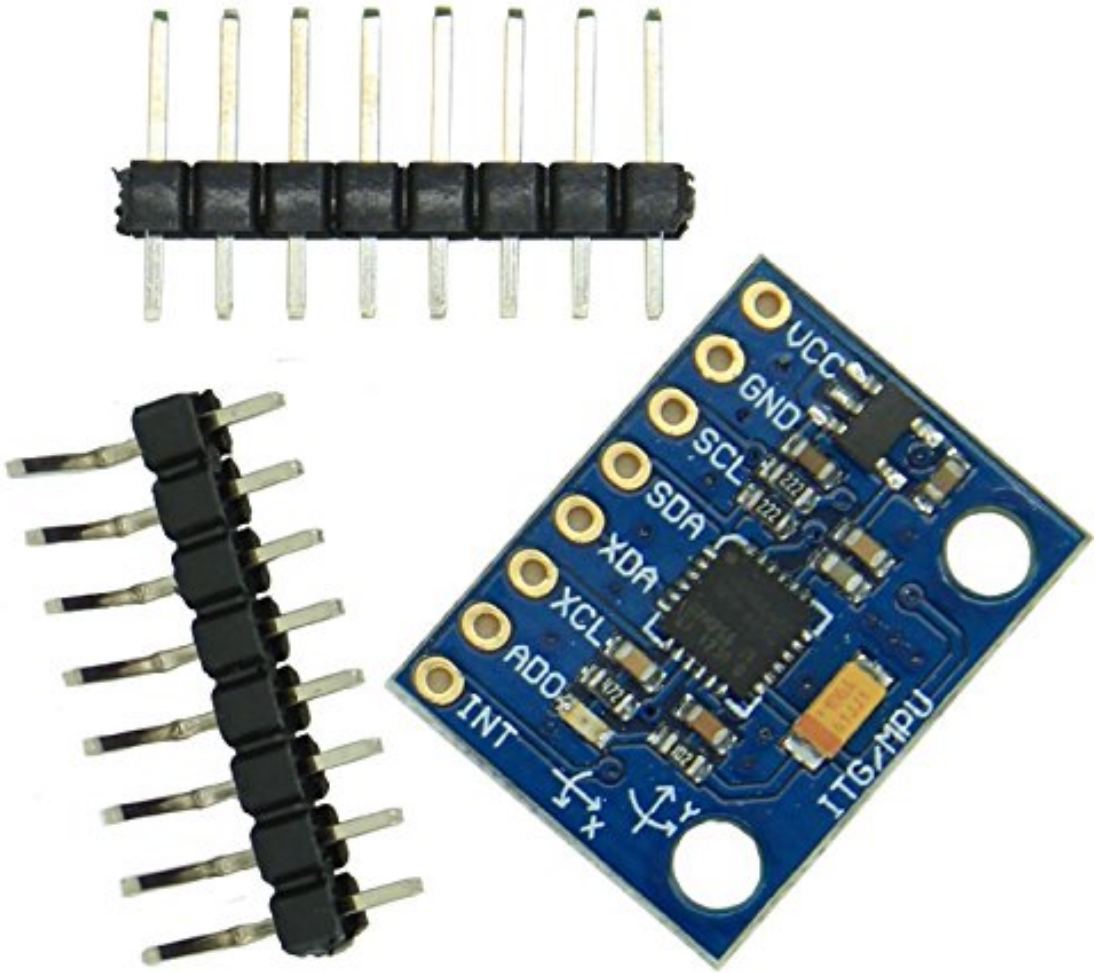


# DC Motor H-Bridge

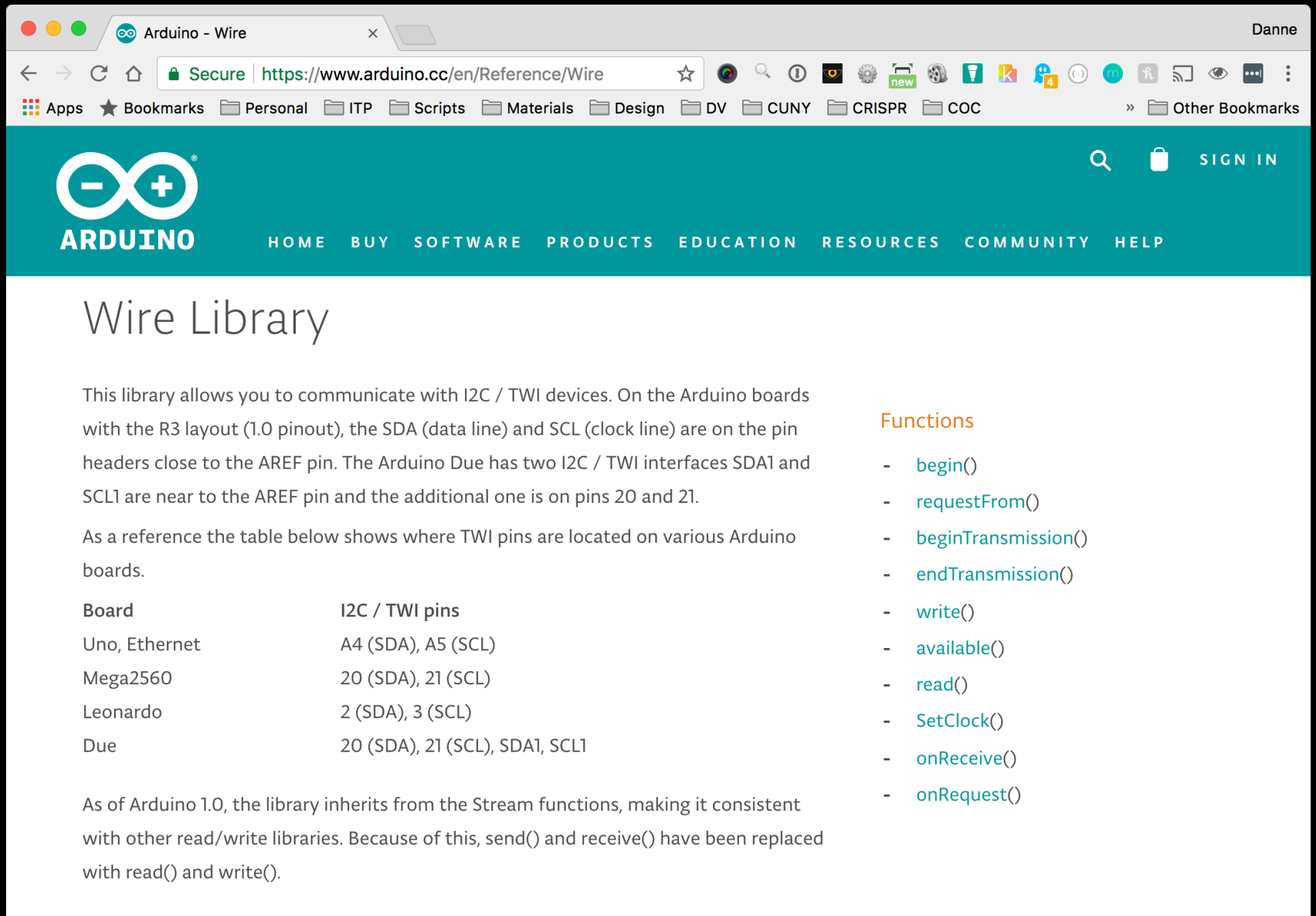


# I2C Communications

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



# Wire Library




## 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()
- beginTransmission()
- 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

pcomp.dannewoo.com