

Introduction to Arduino

Week 4

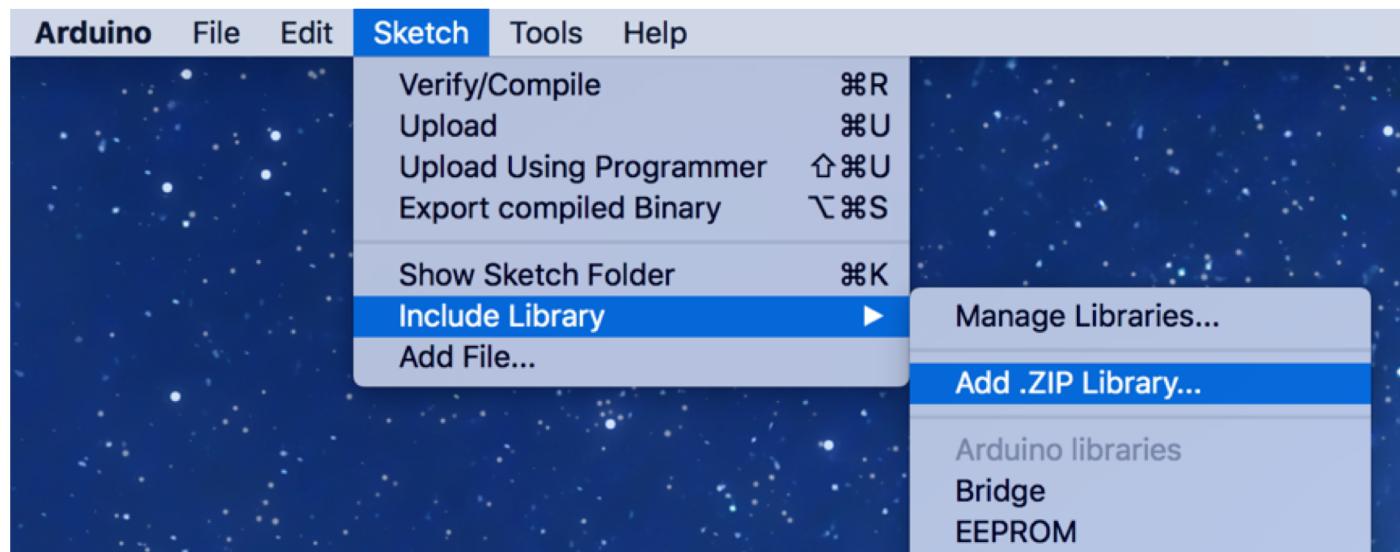
Questions about last 3 weeks

Experiments

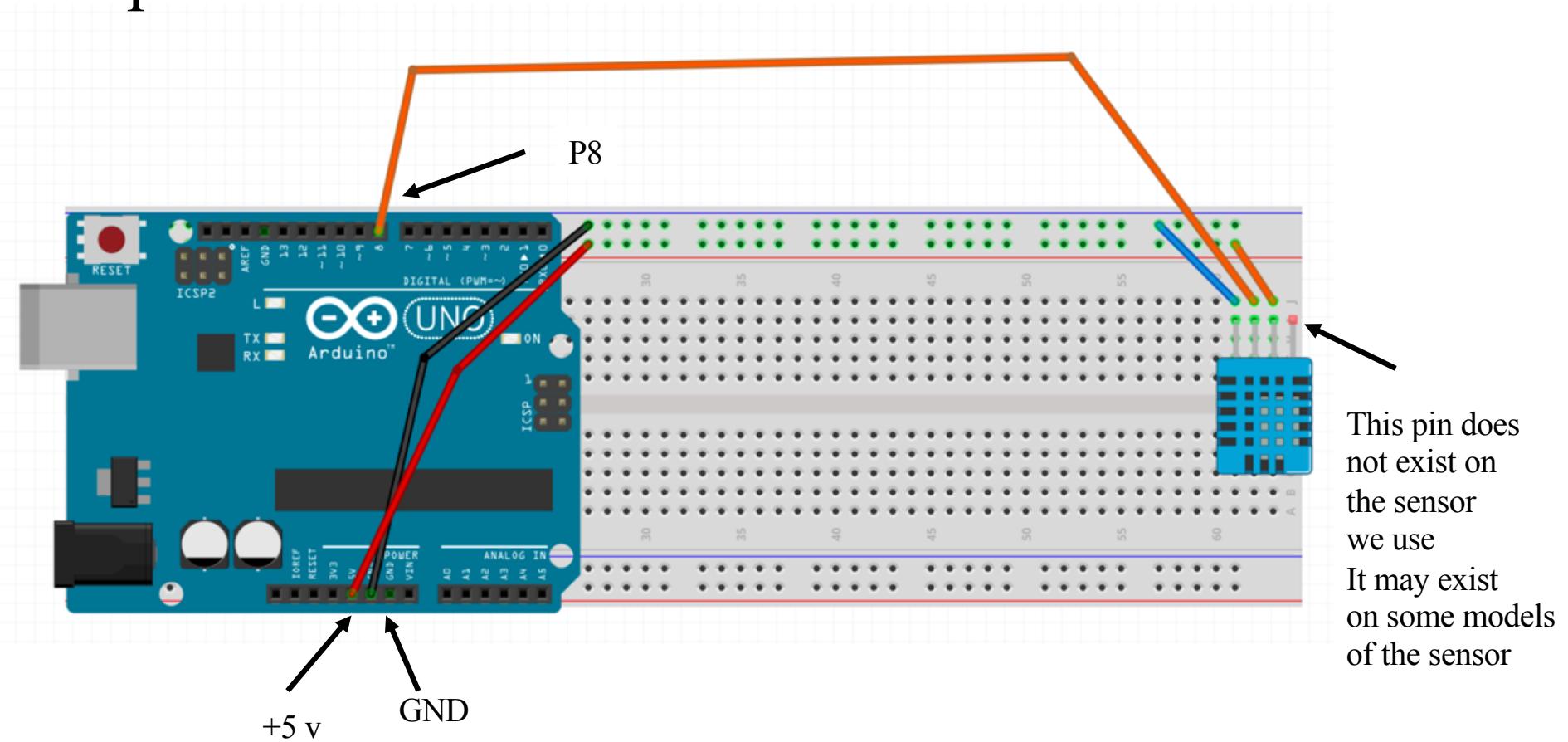
- Experimental test setup
- Temperature and Humidity Sensor using serial output
- Temperature and Humidity Sensor with LCD display
- Servo
- Ultra-sonic ranging using serial output
- Ultra-sonic ranging with LCD display
- Other (if time permits)
 - Buzzer
 - Relay

Temperature & Humidity Sensor experiment

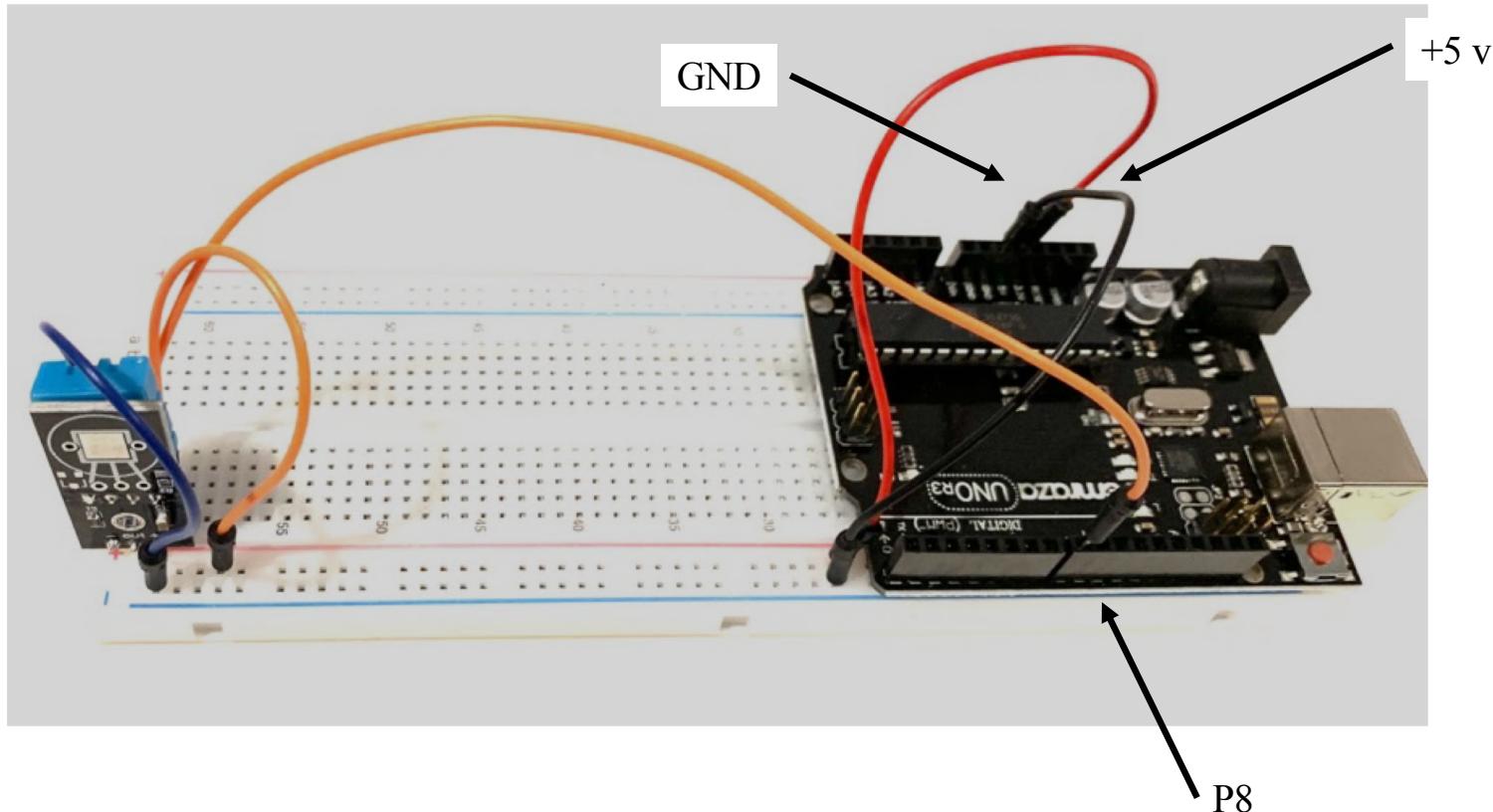
- The supplied kit comes with a basic temperature and humidity sensor
- We have to install the driver for the sensor
- We then use the driver by:



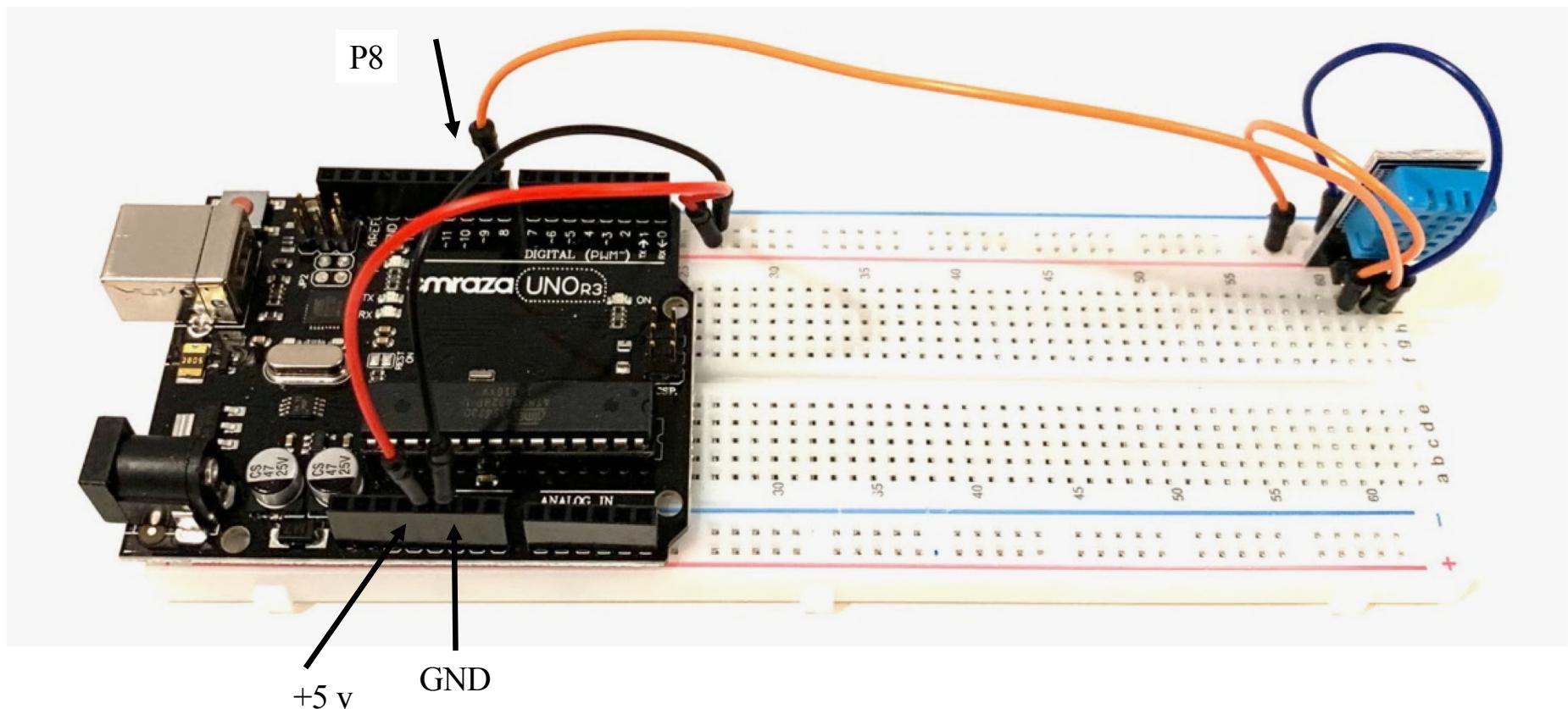
Temperature & Humidity Sensor experimental setup



Temperature & Humidity Sensor experimental setup – View 1

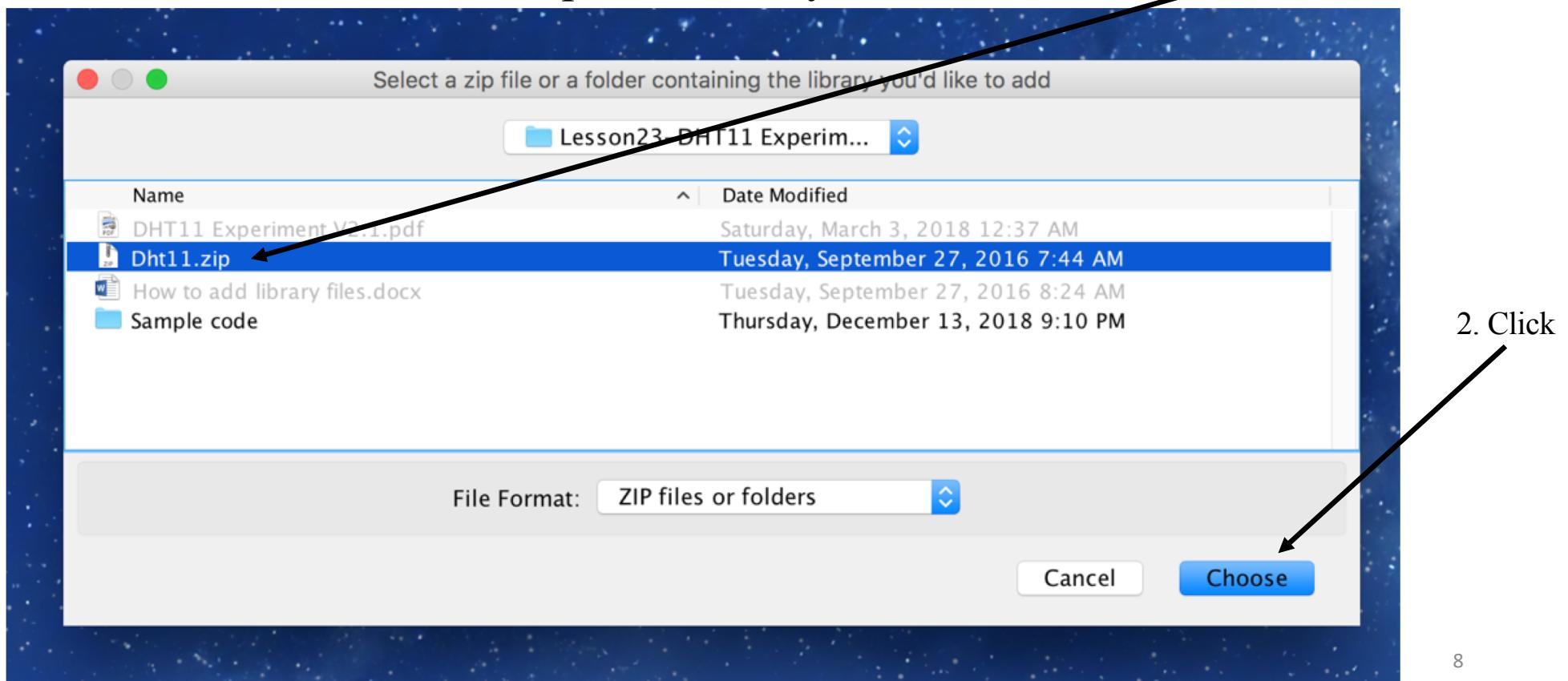


Temperature & Humidity Sensor experimental setup – View 2



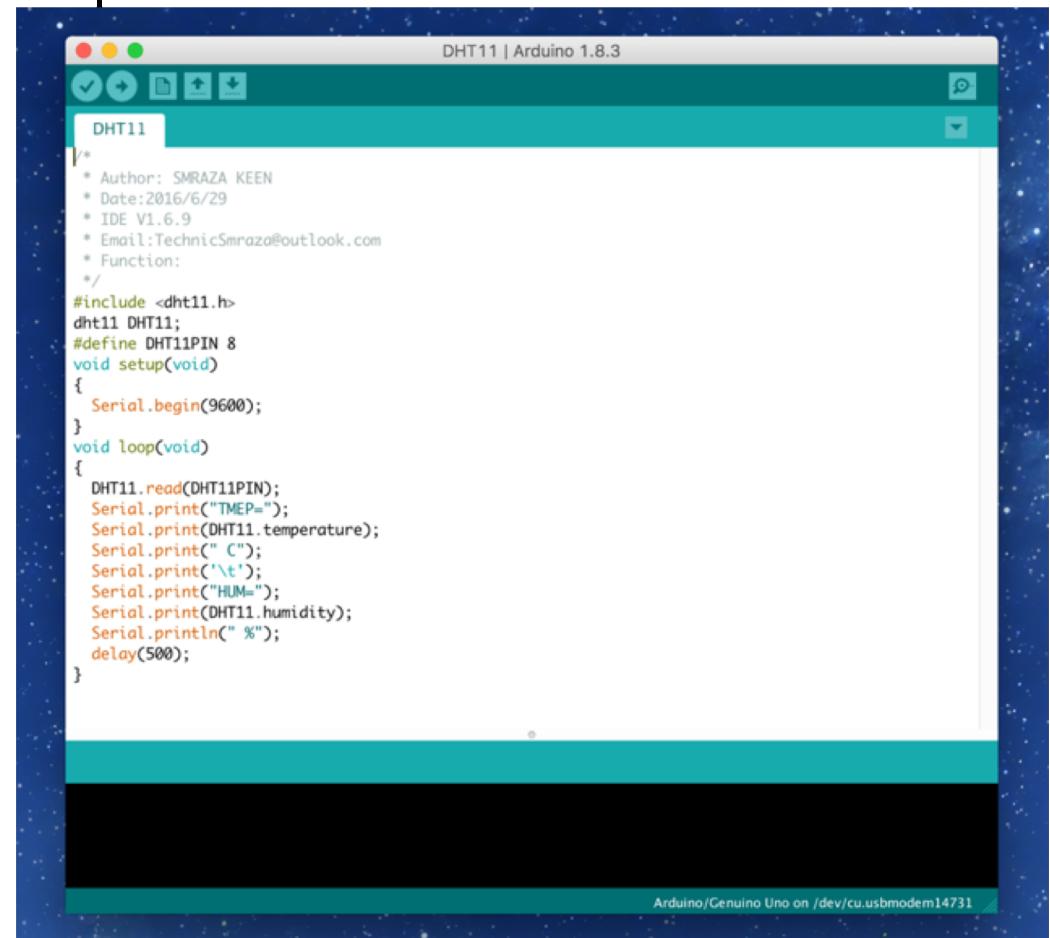
Adding DHT library

- Find the less23-DHT Experiment on your hard drive



Temperature & Humidity Sensor experiment

- Open the sample program from the downloaded zip file
- Lesson 23

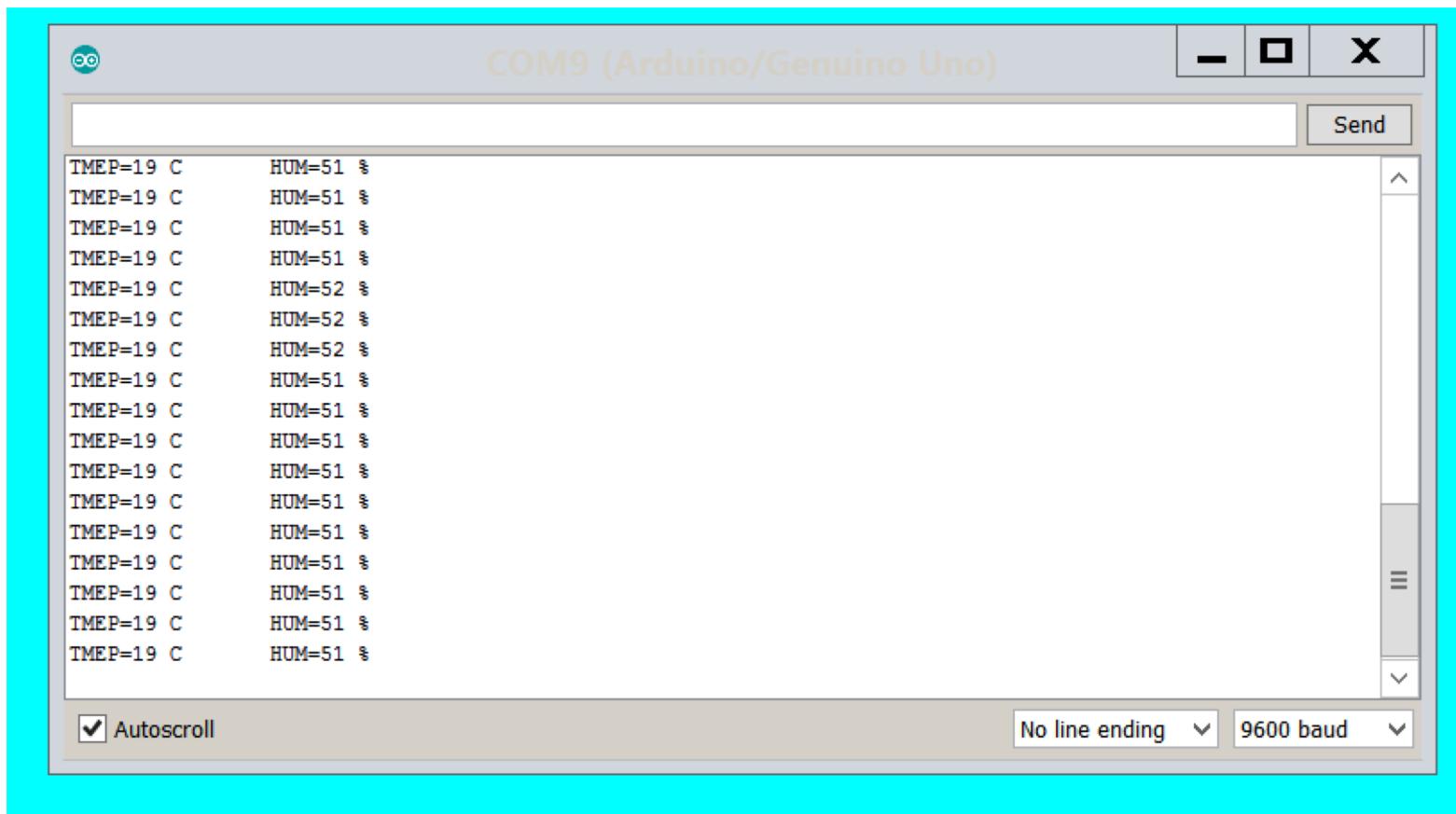


The screenshot shows the Arduino IDE interface with a sketch titled "DHT11". The code is as follows:

```
DHT11
/*
 * Author: SMRAZA KEEN
 * Date:2016/6/29
 * IDE V1.6.9
 * Email:TechnicSmraza@outlook.com
 * Function:
 */
#include <dht11.h>
dht11 DHT11;
#define DHT11PIN 8
void setup(void)
{
    Serial.begin(9600);
}
void loop(void)
{
    DHT11.read(DHT11PIN);
    Serial.print("TMEP=");
    Serial.print(DHT11.temperature);
    Serial.print(" C");
    Serial.print('\t');
    Serial.print("HUM=");
    Serial.print(DHT11.humidity);
    Serial.println(" %");
    delay(500);
}
```

The status bar at the bottom indicates "Arduino/Genuino Uno on /dev/cu.usbmodem14731".

Sample Output of Temperature and Humidity Measure



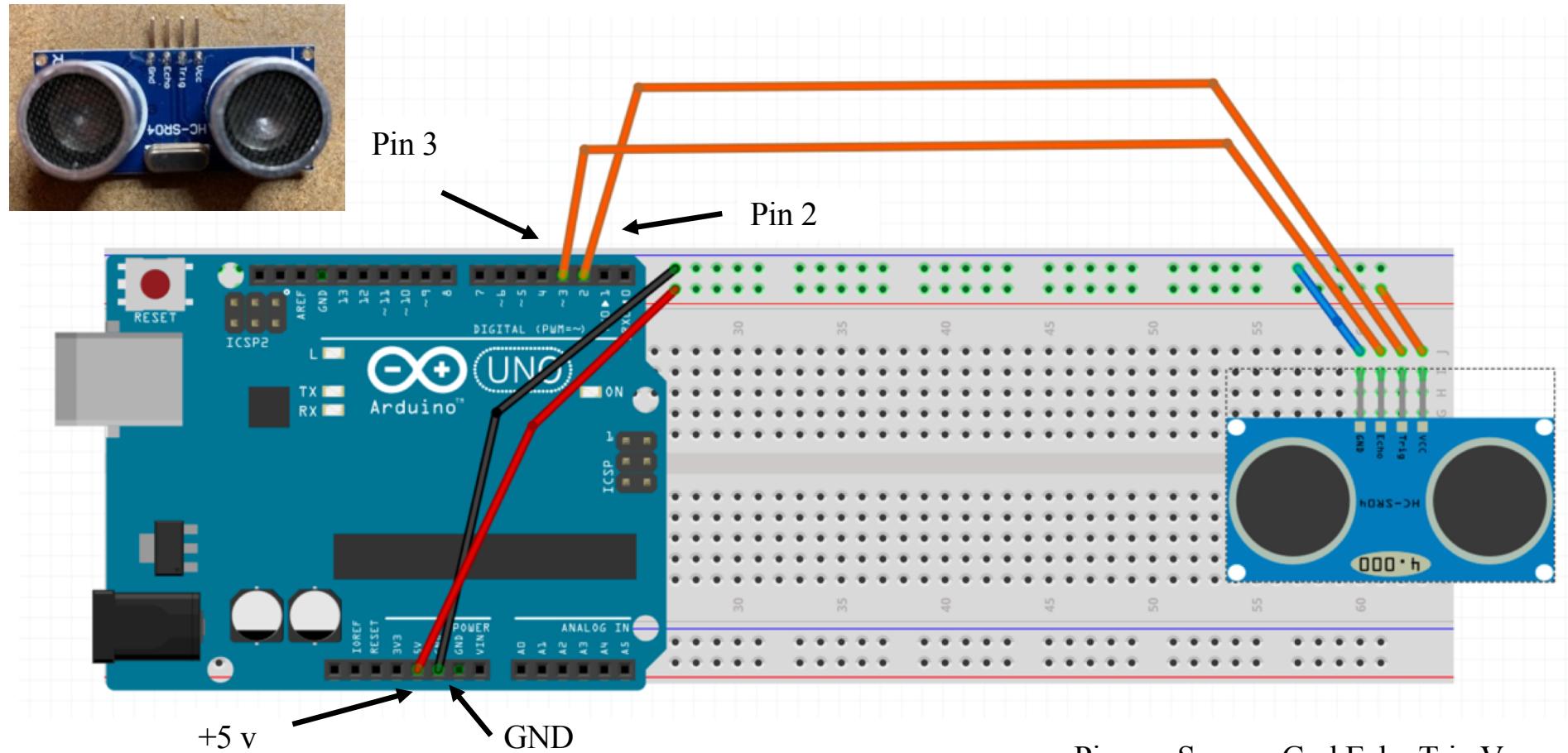
Ultra-sonic ranging Experiment

- The ultrasonic sensor has a
 - Transmitter (Trigger pin)
 - Receiver (Echo pin)
- Over a range of 0 – 1 meter (or so),
 - The sound waves will be transmitted by the sensor
 - Reflected off an object
 - Received at the sensor
- It functions like a radar using sound rather than radio waves
- It is useful to measure distance
- It can also be used on a robot to avoid collisions with objects

Warning

- The HC-SR04 ultrasonic sensor is a 5 volt device
- The Arduino Uno is a 5 volt device
- Other Arduino types use 3.3 volts
 - On them we must use a level converter on the Echo line
- Not required for the Uno

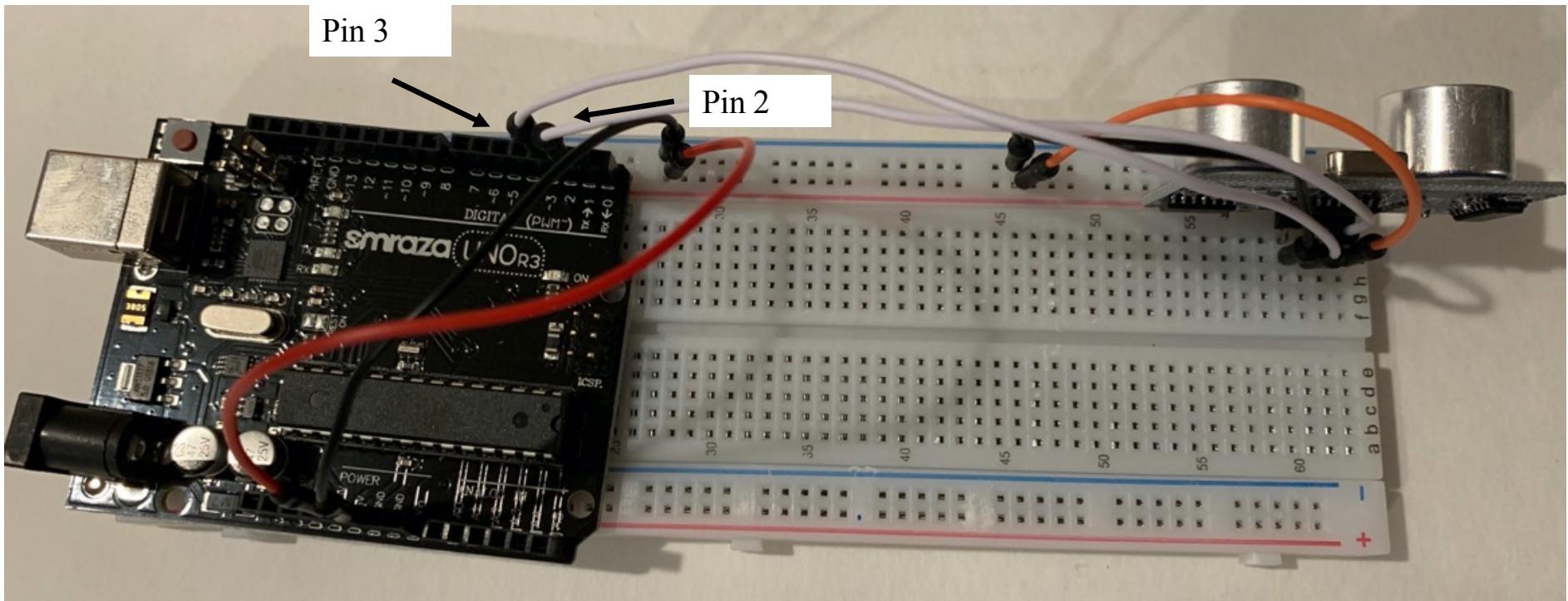
Distance Measure Setup



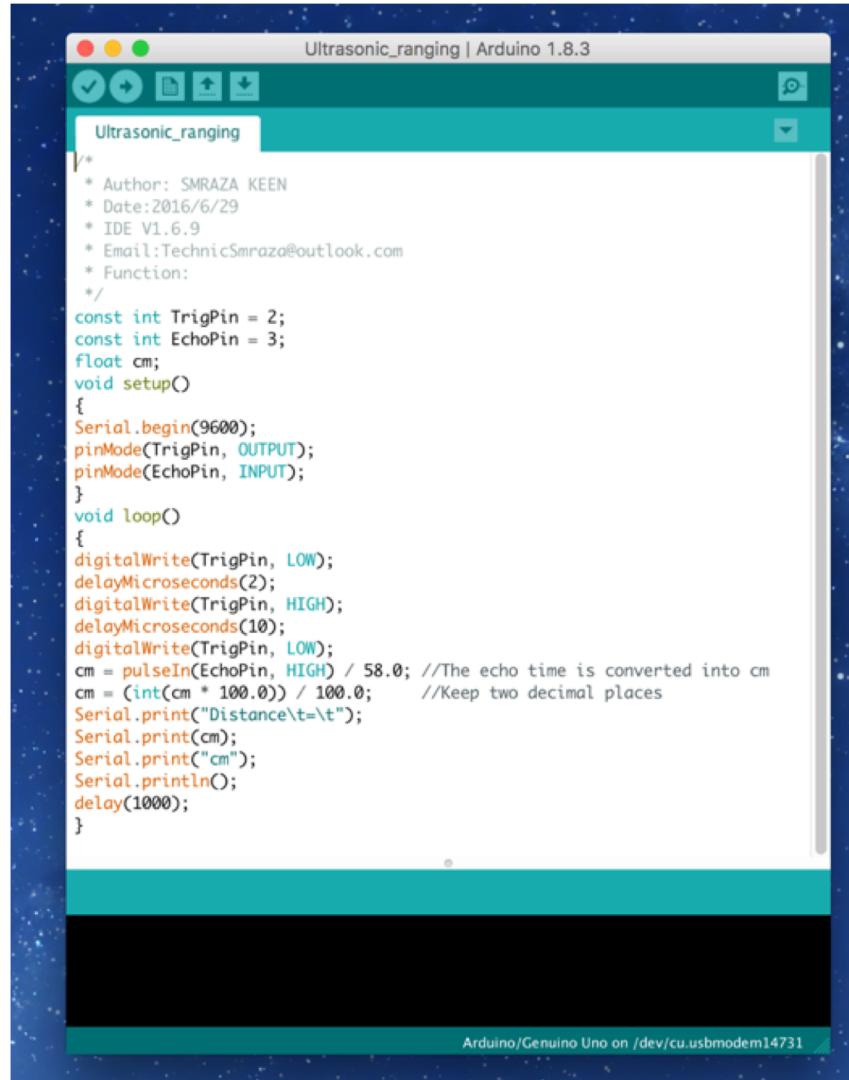
Copyright 2023 Joseph E. Wilkes

Pins on Sensor: Gnd Echo Trig Vcc

Distance Measure Setup



Ultrasonic Ranging Code



The screenshot shows the Arduino IDE interface with the title bar "Ultrasonic_ranging | Arduino 1.8.3". The main window displays the C++ code for an ultrasonic ranging application. The code includes comments about the author, date, IDE version, email, and function. It defines pins for the trig and echo pins, initializes the serial port at 9600 bps, sets up the pins, and performs a loop where it sends a pulse, measures the duration, converts it to distance, and prints the result to the Serial monitor. The code uses standard Arduino libraries like `Serial` and `digitalWrite`.

```
/*
 * Author: SMRAZA KEEN
 * Date:2016/6/29
 * IDE V1.6.9
 * Email:TechnicSmraza@outlook.com
 * Function:
 */
const int TrigPin = 2;
const int EchoPin = 3;
float cm;
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);
  cm = pulseIn(EchoPin, HIGH) / 58.0; //The echo time is converted into cm
  cm = (int(cm * 100.0)) / 100.0; //Keep two decimal places
  Serial.print("Distance\t=\t");
  Serial.print(cm);
  Serial.print("cm");
  Serial.println();
  delay(1000);
}
```

Arduino/Genuino Uno on /dev/cu.usbmodem14731

Example Output

An arrow points to the line 'Distance = -325.91cm' in the terminal window.

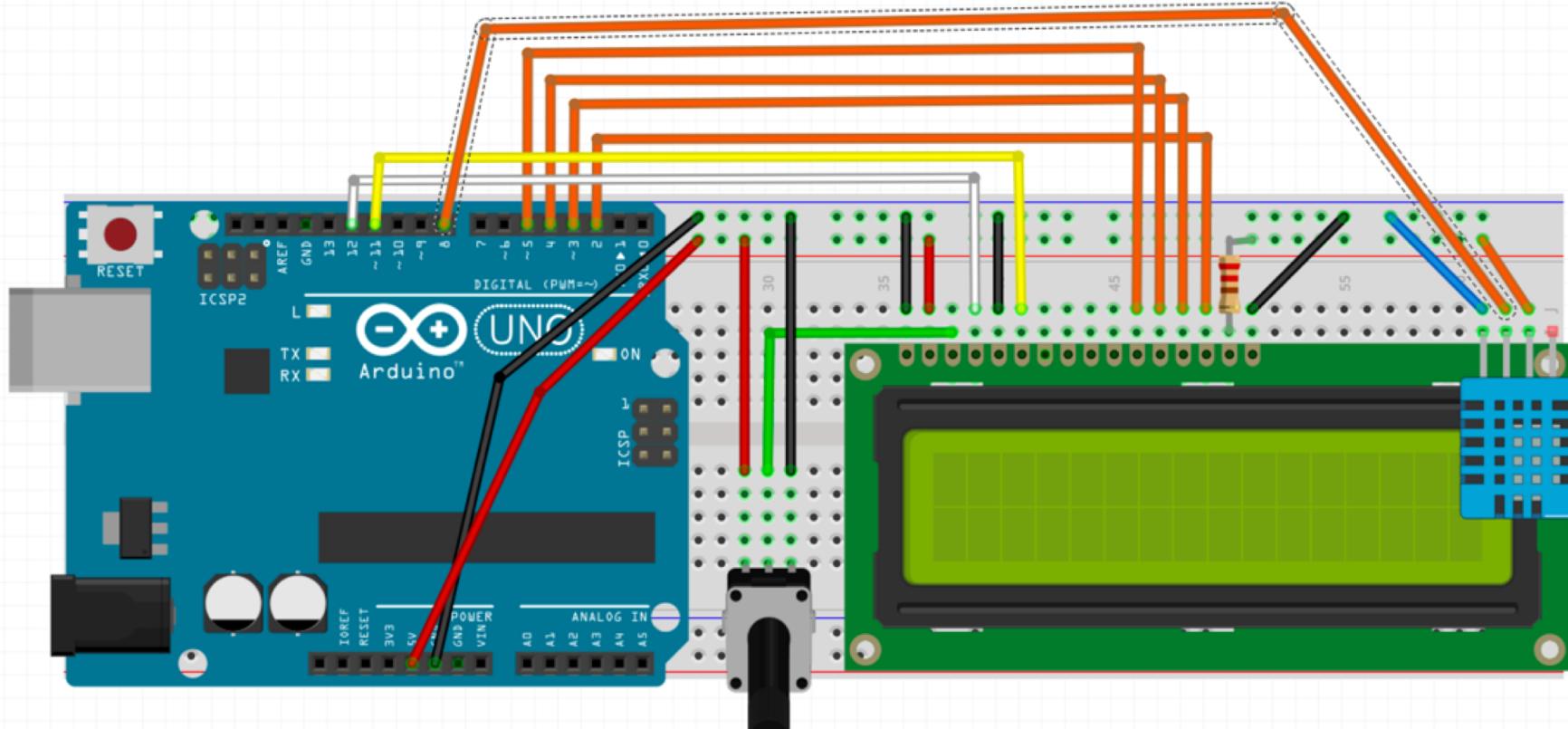
Distance	=	Value
Distance	=	10.36cm
Distance	=	10.27cm
Distance	=	14.96cm
Distance	=	14.65cm
Distance	=	11.32cm
Distance	=	12.17cm
Distance	=	11.70cm
Distance	=	10.53cm
Distance	=	10.51cm
Distance	=	10.87cm
Distance	=	10.87cm
Distance	=	10.87cm
Distance	=	10.84cm
Distance	=	10.79cm
Distance	=	11.03cm
Distance	=	11.24cm
Distance	=	-325.91cm

Anomalous Reading

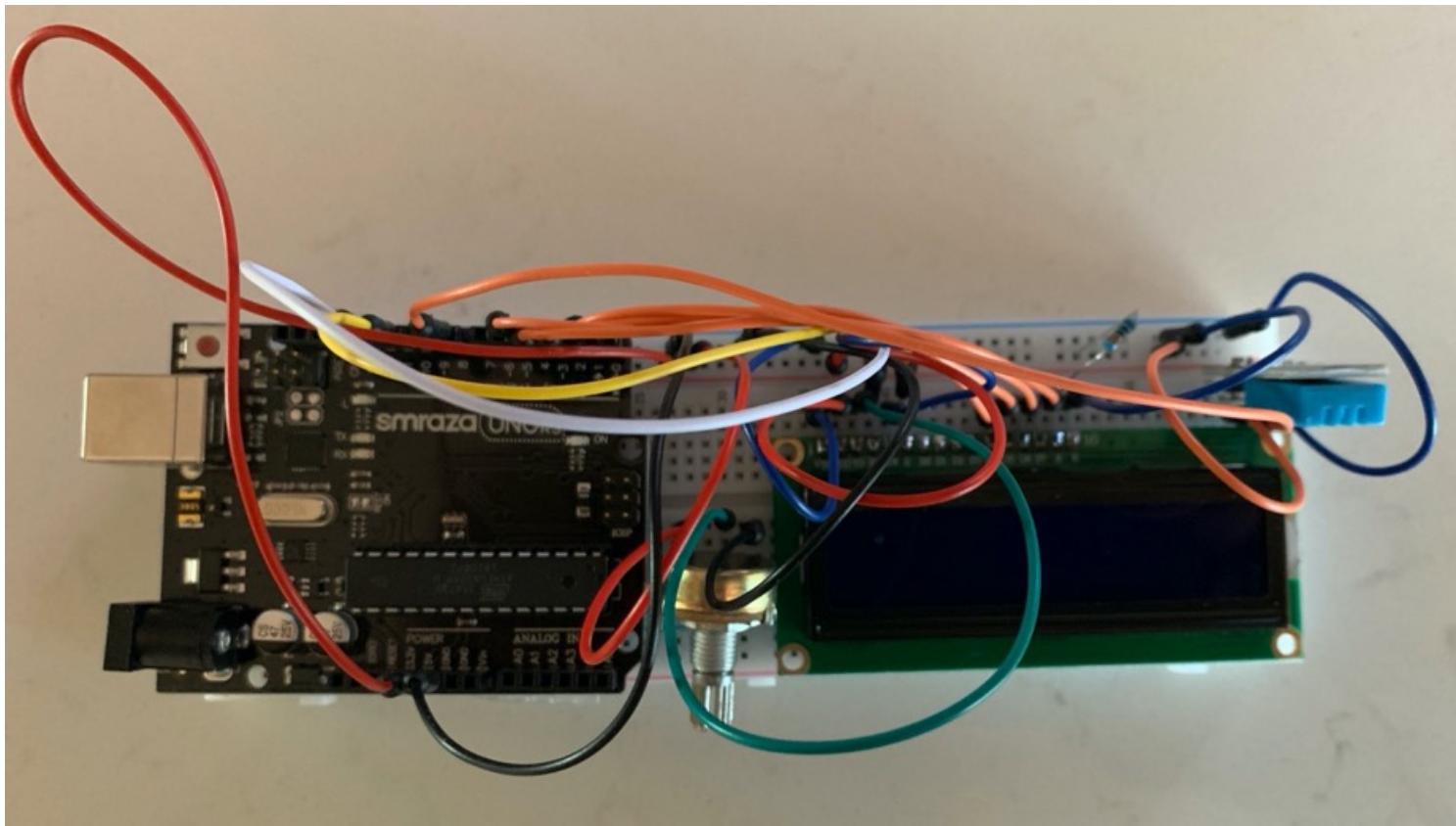
Combining Temperature and Humidity Sensor with LCD

- Instead of displaying the temperature and humidity on the serial terminal will we now display it on the LCD display

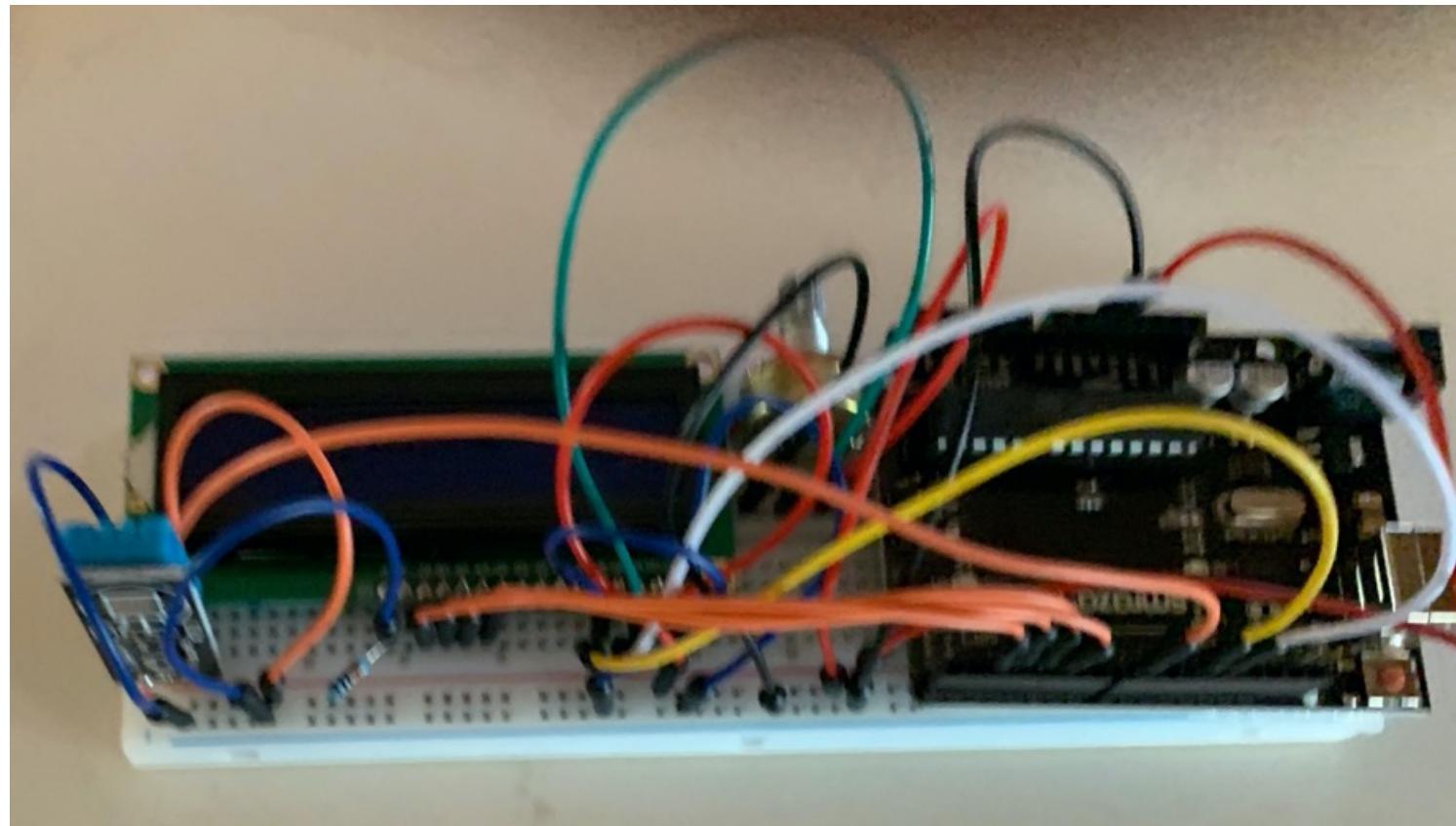
Temperature and Humidity Sensor with LCD of Wiring



Temperature and Humidity Sensor with LCD experimental setup



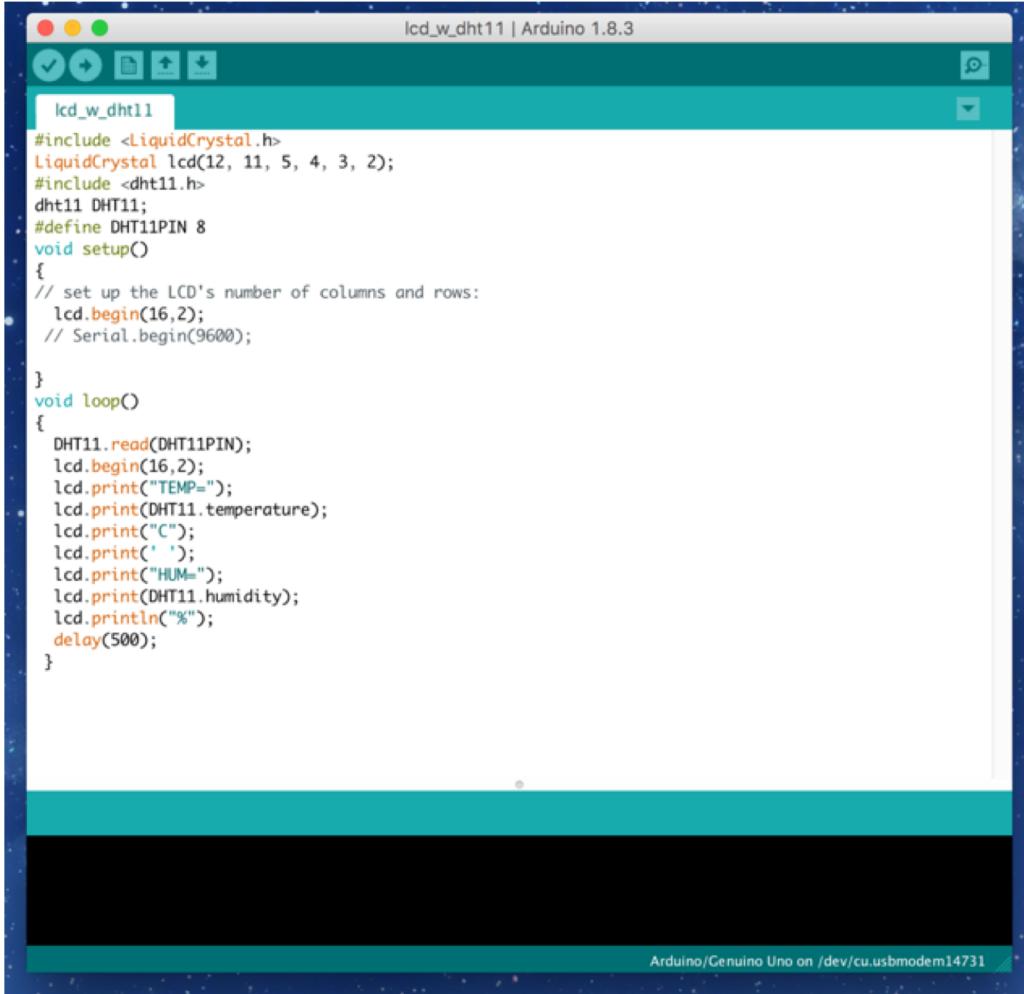
Temperature and Humidity Sensor with LCD experimental setup (another view)



Copyright 2023 Joseph E. Wilkes

20

Code



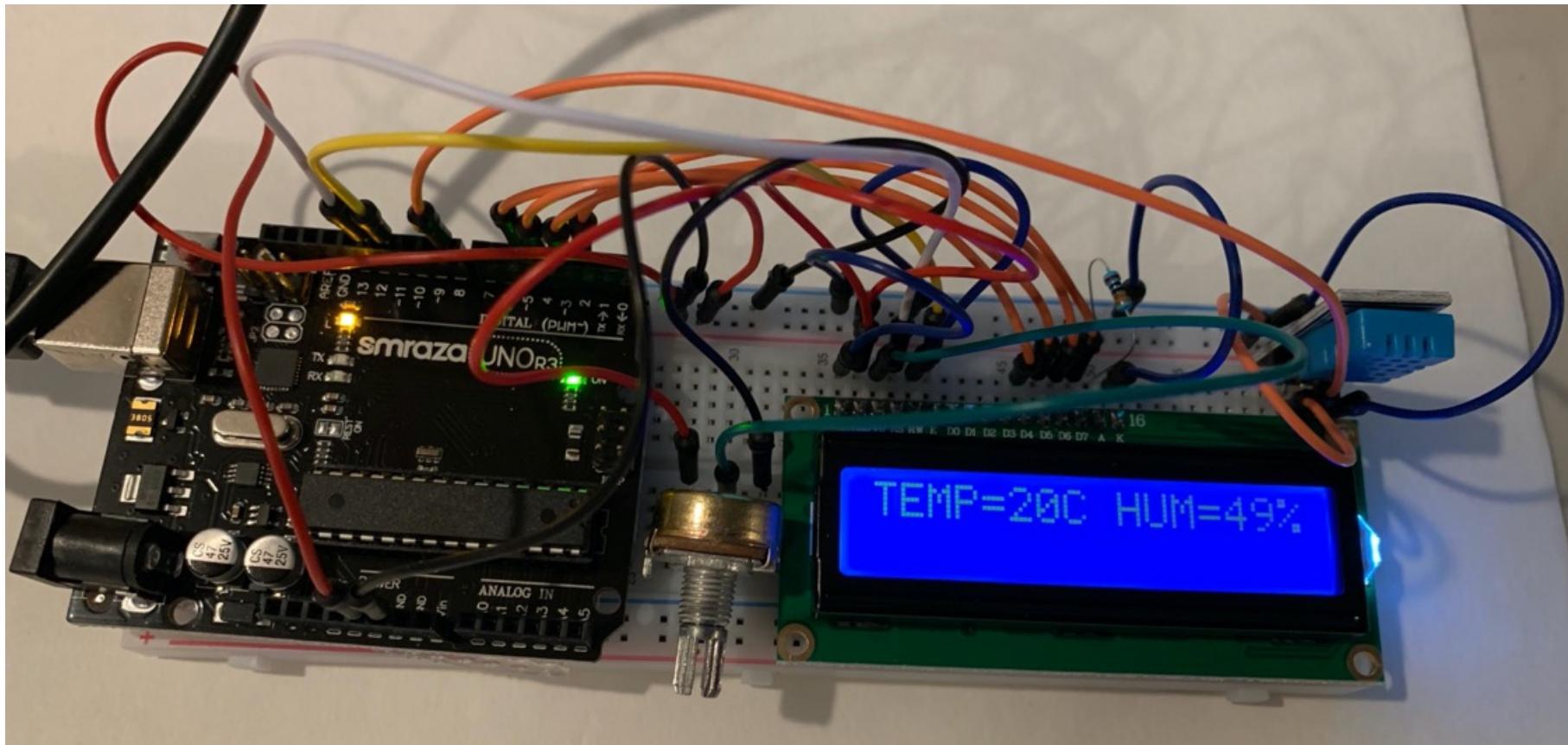
The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** lcd_w_dht11 | Arduino 1.8.3
- Code Area:** The code is written in C++ and defines a class for an LCD and a DHT11 sensor. It includes headers for LiquidCrystal and dht11, initializes the LCD and serial communication, reads data from the DHT11 sensor, and prints the temperature and humidity to the LCD screen.
- Status Bar:** Shows the connection status: Arduino/Genuino Uno on /dev/cu.usbmodem14731

```
lcd_w_dht11
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
#include <dht11.h>
dht11 DHT11;
#define DHT11PIN 8
void setup()
{
    // set up the LCD's number of columns and rows:
    lcd.begin(16,2);
    // Serial.begin(9600);

}
void loop()
{
    DHT11.read(DHT11PIN);
    lcd.begin(16,2);
    lcd.print("TEMP=");
    lcd.print(DHT11.temperature);
    lcd.print("C");
    lcd.print(' ');
    lcd.print("HUM=");
    lcd.print(DHT11.humidity);
    lcd.println("%");
    delay(500);
}
```

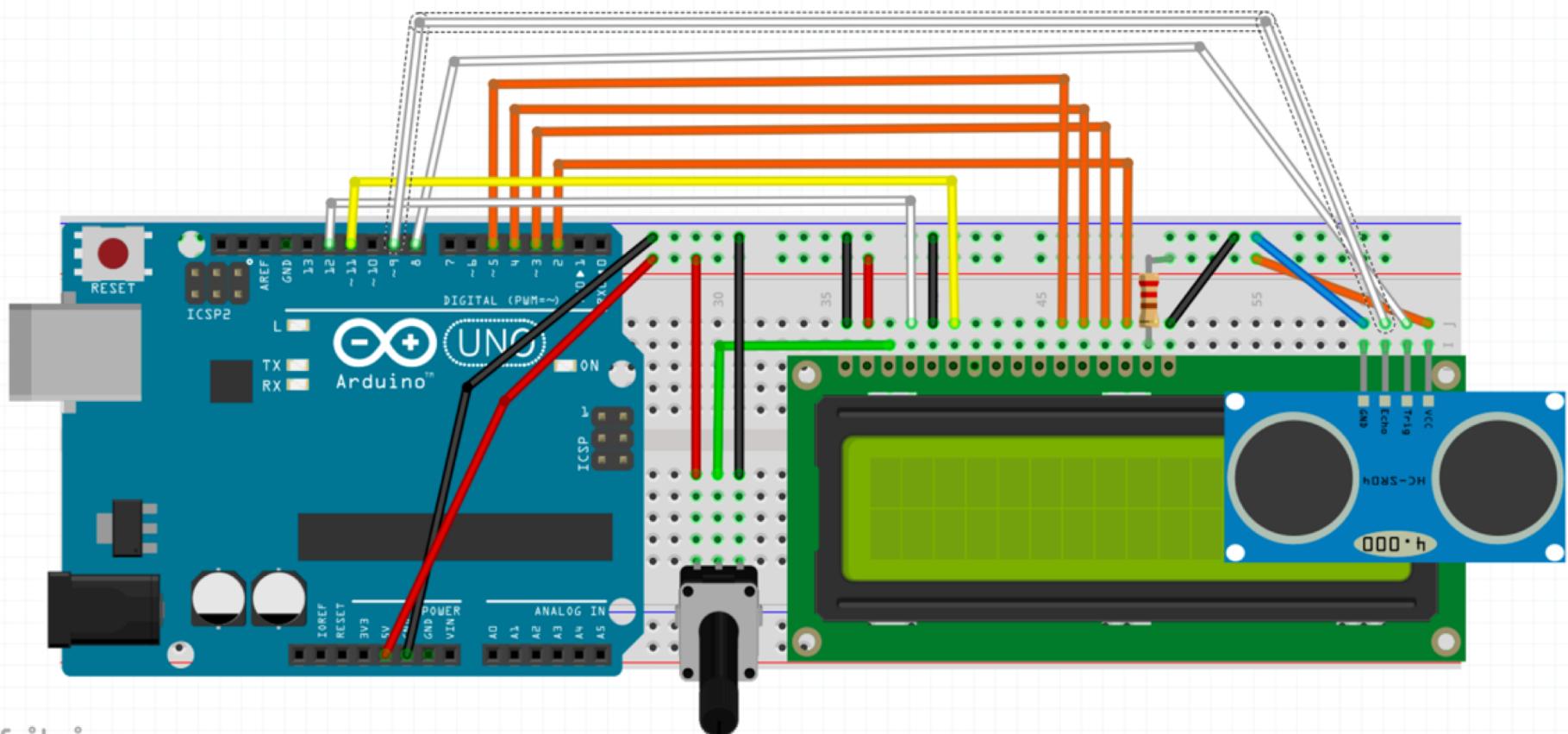
Sample Output



Display Distance Sensor on LCD

- We will combine ultra sonic sensor with LCD display
- The ultra sonic sensor uses pins 2 and 3
- The LCD display uses pins 2 and 3
- So we will change the ultra sonic sensor to use pins 8 and 9

LCD display of range experimental setup

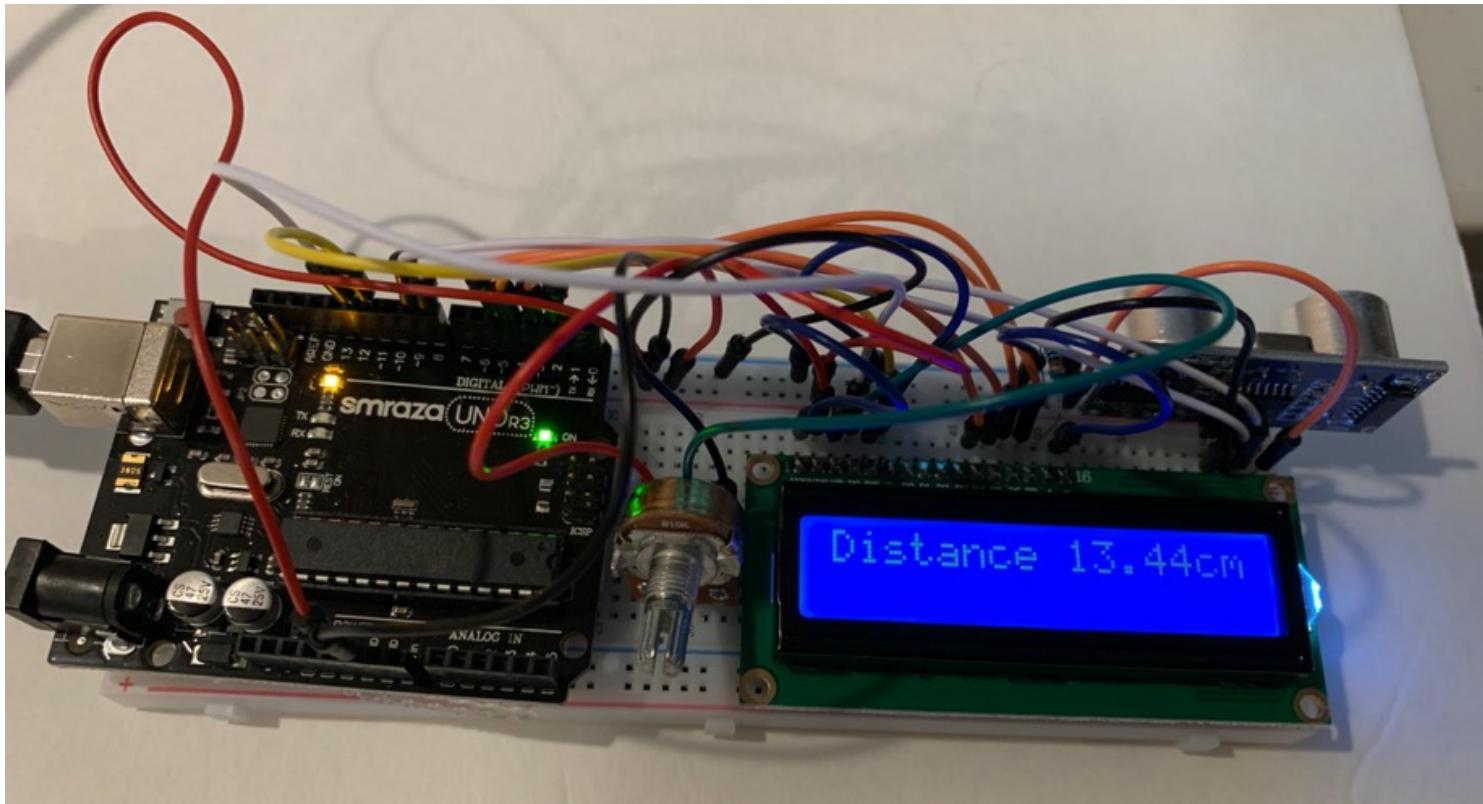


fritzing

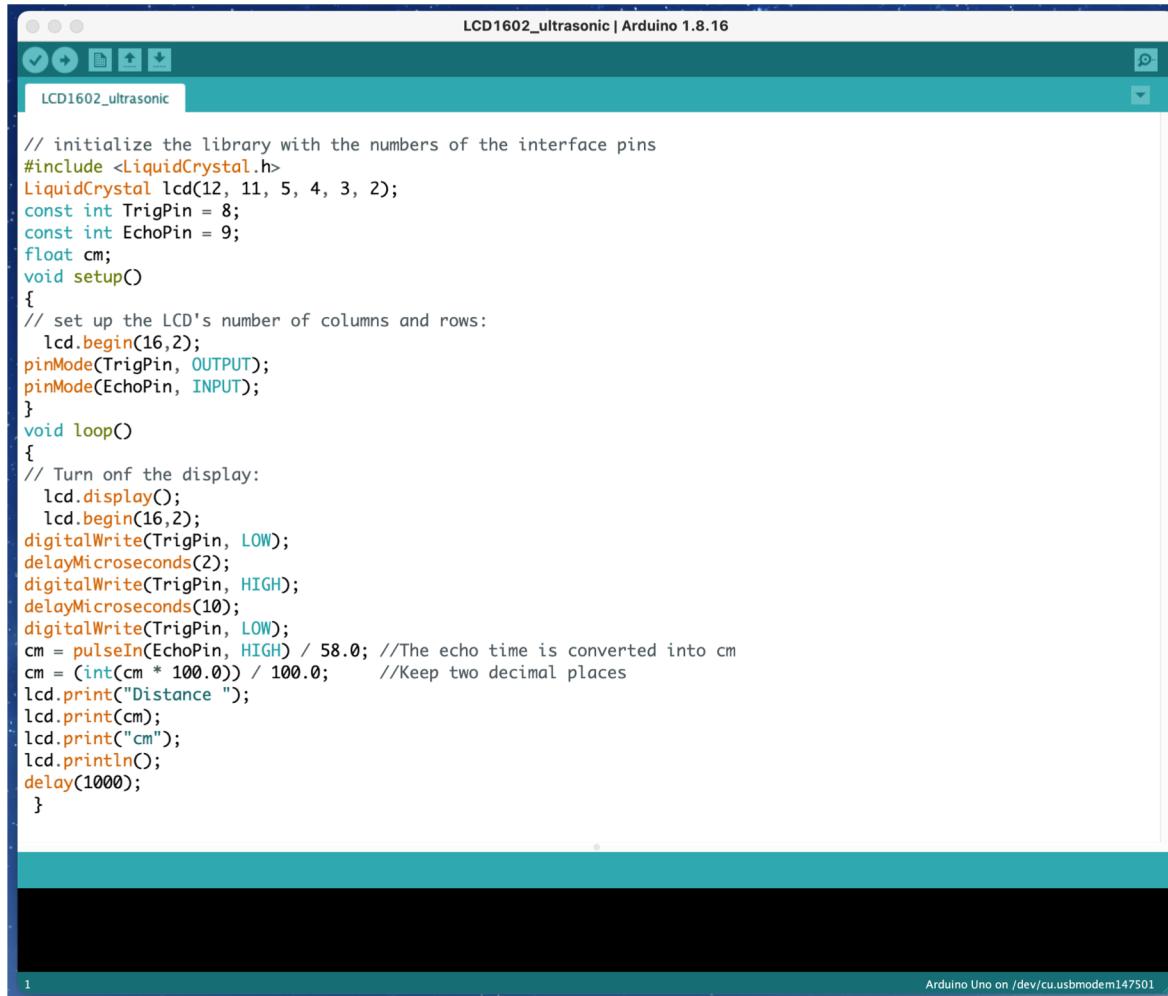
Copyright 2023 Joseph E. Wilkes

24

LCD display of range experimental setup



LCD display of range code

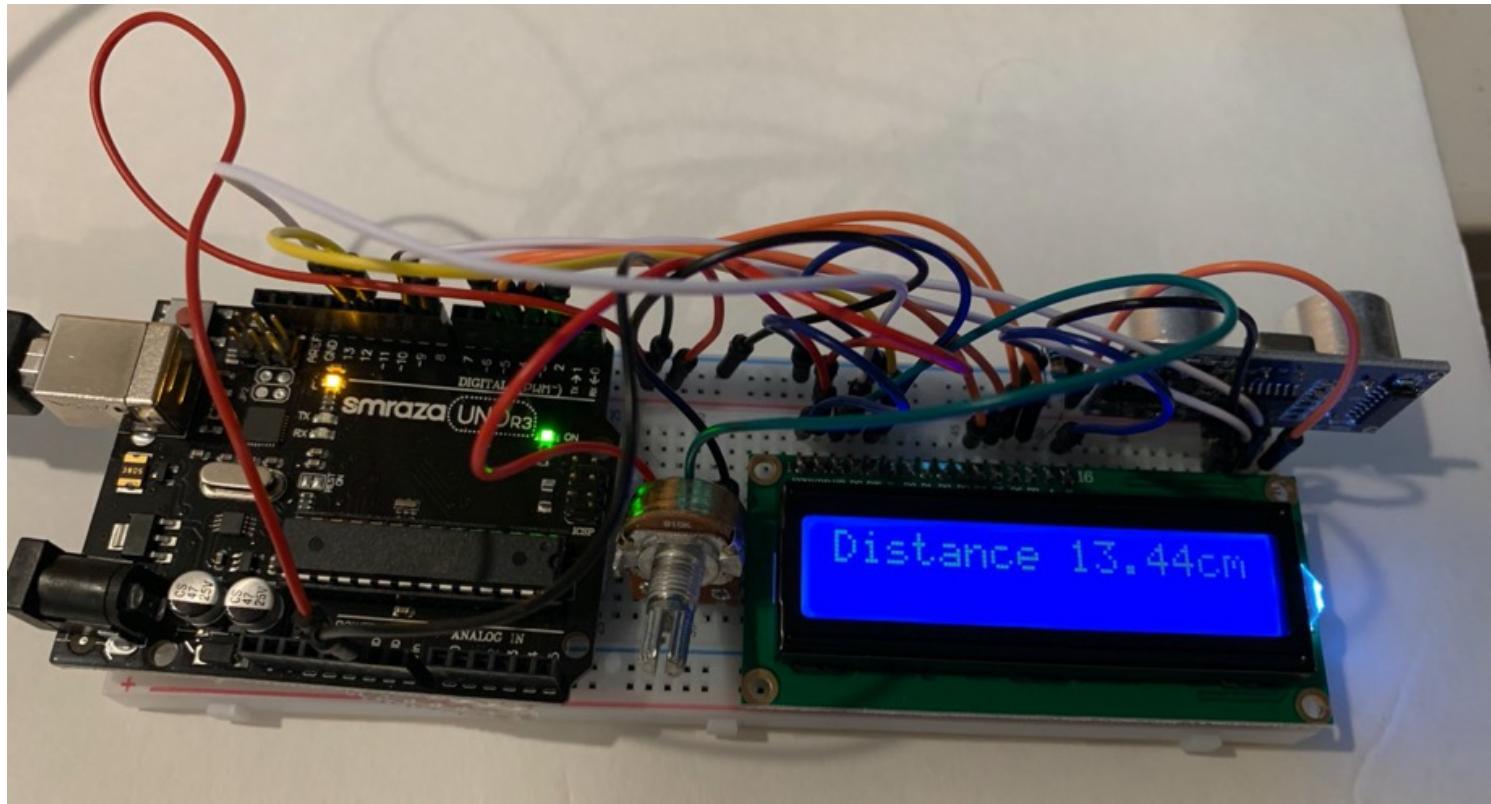


The screenshot shows the Arduino IDE interface with the title bar "LCD1602_ultrasonic | Arduino 1.8.16". The main window displays the following C++ code:

```
// initialize the library with the numbers of the interface pins
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
const int TrigPin = 8;
const int EchoPin = 9;
float cm;
void setup()
{
// set up the LCD's number of columns and rows:
lcd.begin(16,2);
pinMode(TrigPin, OUTPUT);
pinMode(EchoPin, INPUT);
}
void loop()
{
// Turn on the display:
lcd.display();
lcd.begin(16,2);
digitalWrite(TrigPin, LOW);
delayMicroseconds(2);
digitalWrite(TrigPin, HIGH);
delayMicroseconds(10);
digitalWrite(TrigPin, LOW);
cm = pulseIn(EchoPin, HIGH) / 58.0; //The echo time is converted into cm
cm = (int(cm * 100.0)) / 100.0; //Keep two decimal places
lcd.print("Distance ");
lcd.print(cm);
lcd.println();
delay(1000);
}
```

The status bar at the bottom right indicates "Arduino Uno on /dev/cu.usbmodem147501".

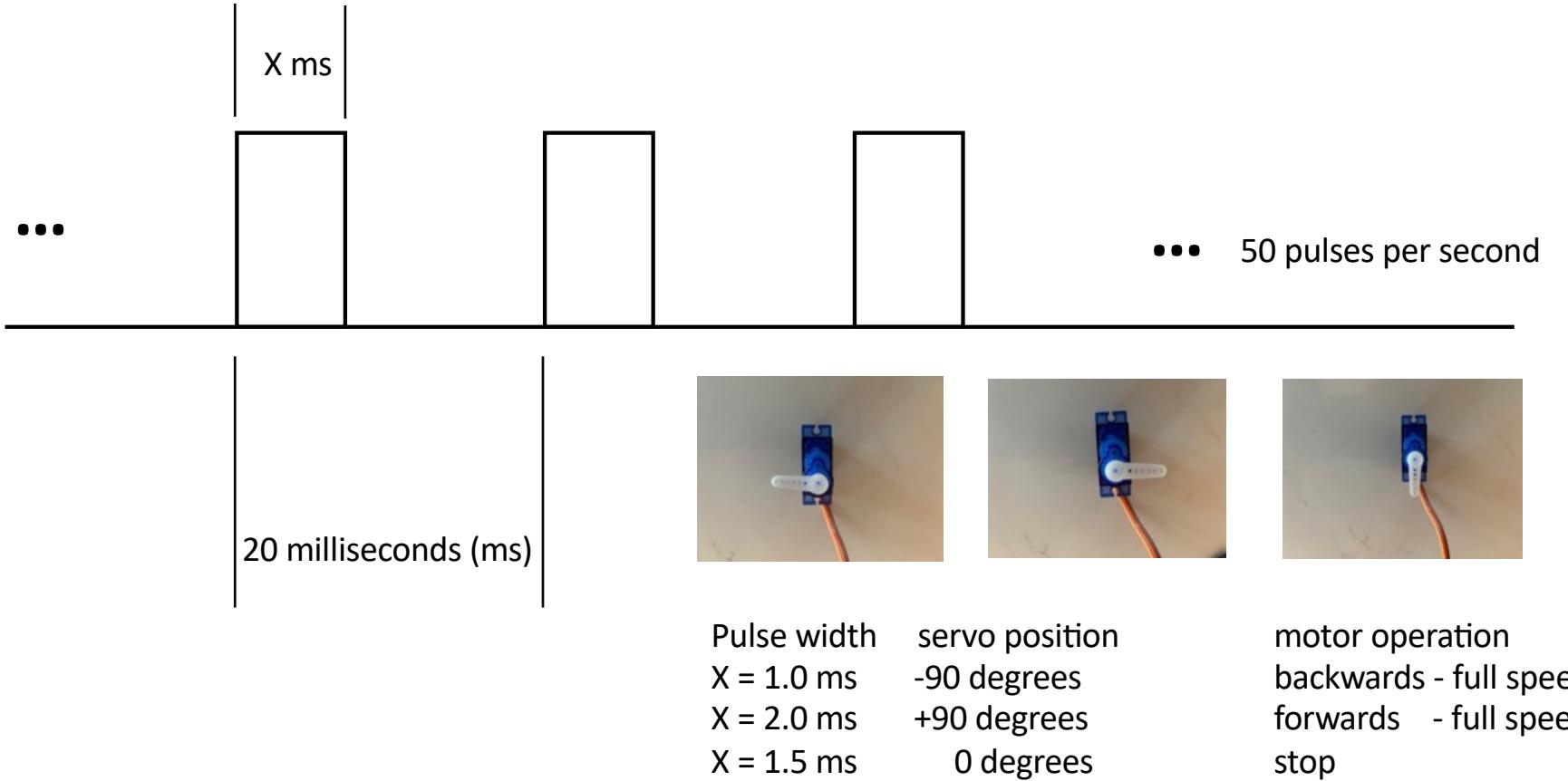
Sample Output of LCD display of range



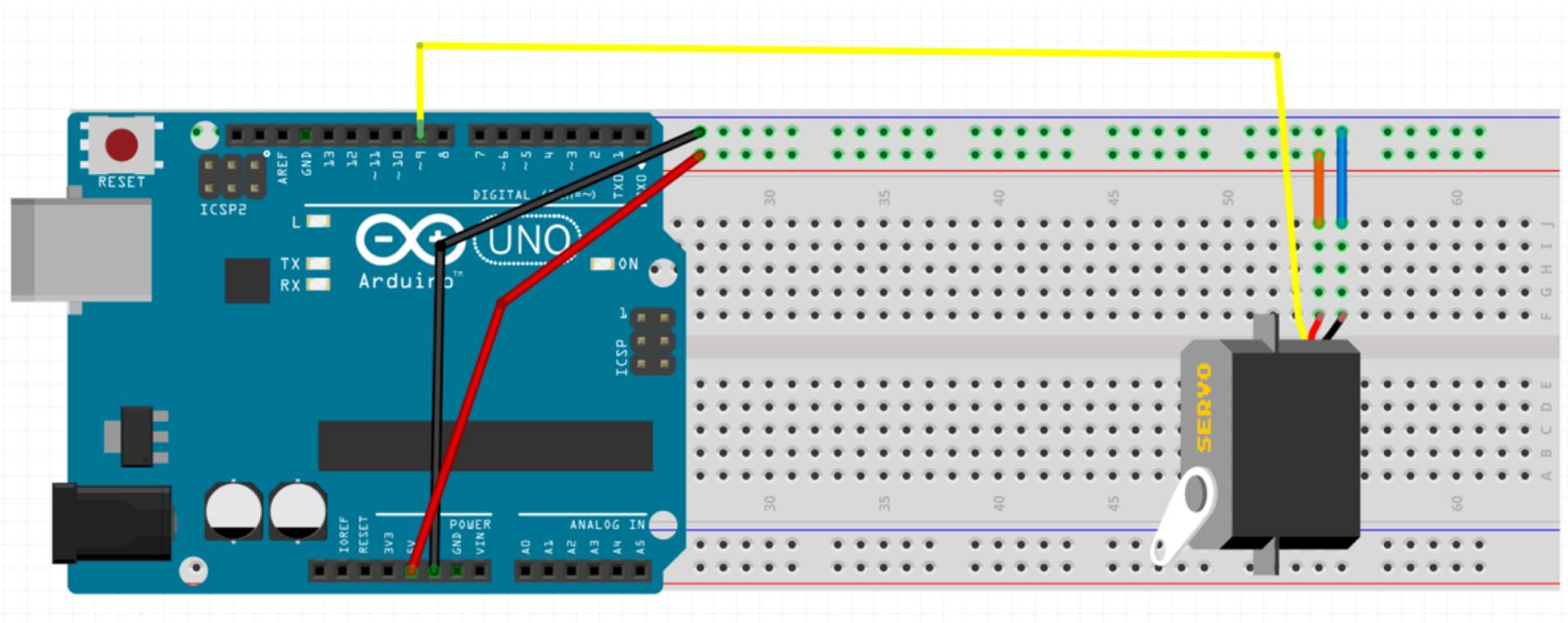
Example controlling a Servo

- Servos are used in radio controlled airplanes, cars and robots
- There is a standard way of controlling the servo
- Most Servos meet this standard
- Some deviate slightly from the standard
- Servos change their position based on a signal sent to them
 - The position can range from -90 to 0 to +90 degrees
- Some servos are modified to continuously rotate and thus become motors
 - The range of speed is from full backwards to stop to full forward
- The Arduino has commands to control servos

Standard Servo Operation



Servo Test Setup



Servo Sweep Code



The screenshot shows the Arduino IDE interface with the title bar "Sweep | Arduino 1.8.3". The code editor contains the "Sweep" sketch, which controls a servo to sweep from 0 to 180 degrees and back again. The code includes comments explaining the purpose of each section.

```
/* Sweep
by BARRAGAN <http://baraganstudio.com>
This example code is in the public domain.

modified 8 Nov 2013
by Scott Fitzgerald
http://www.arduino.cc/en/Tutorial/Sweep
*/
#include <Servo.h>

Servo myservo; // create servo object to control a servo
// twelve servo objects can be created on most boards

int pos = 0; // variable to store the servo position

void setup() {
  myservo.attach(9); // attaches the servo on pin 9 to the servo object
}

void loop() {
  for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
    // in steps of 1 degree
    myservo.write(pos); // tell servo to go to position in variable 'pos'
    delay(15); // waits 15ms for the servo to reach the position
  }
  for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservo.write(pos); // tell servo to go to position in variable 'pos'
    delay(15); // waits 15ms for the servo to reach the position
  }
}
```

Arduino/Genuino Uno on /dev/cu.usbmodem14731

Questions?

Post Class Projects

- If you haven't completed any of the experiments in class, try them after the class
- Look at the experiments for:
 - Buzzer
 - Speaker
 - Relay
 - RGB LED
- Try to:
 - If you know C, do the experiment
 - Try to combine individual experiments in one experiment