Mechatronics Tutorials Report

Ana Jaramillo, Brandon Lee, Miina Anvelt, Hannah Ramsperger
ME 14

Tutorial #1: Getting Started with Arduino

Guide followed: https://howtomechatronics.com/tutorials/arduino/getting-started/

1. *Is there a polarity to LEDs? If so, how would one distinguish the positive and negative terminals?*

Yes, LEDs have polarity. To distinguish between the positive and negative terminals, one can notice that the longer end is positive and the shorter end is negative. In this tutorial, the negative terminal was connected to the ground on the Arduino and the positive terminal was connected to the 5V power from Arduino.

2. *Please explain the functionality of the setup function.*

As was described in the tutorial video, the setup function runs the function once after each power-up or reset of the Arduino board. It initializes the various pin numbers and variables for different parts which can be used on the board.

3. *Please explain the functionality of the loop function.*

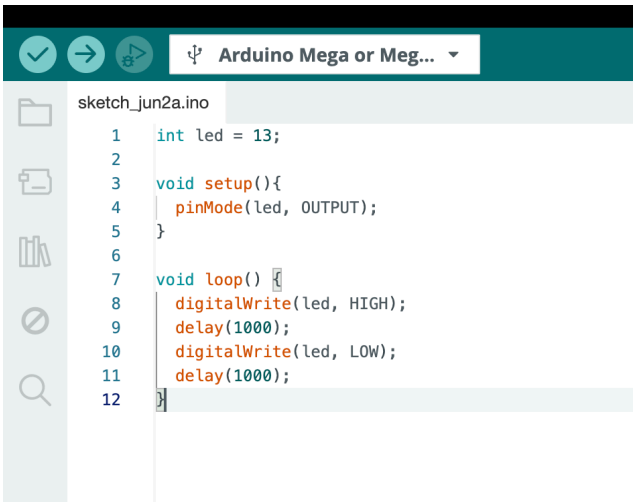
The loop function contains the main code of the program for the Arduino. It runs repeatedly and repeats the function which is desired to be run. For example, in this tutorial the loop turned on the LED, waited, and then turned it off and this was repeated continuously.

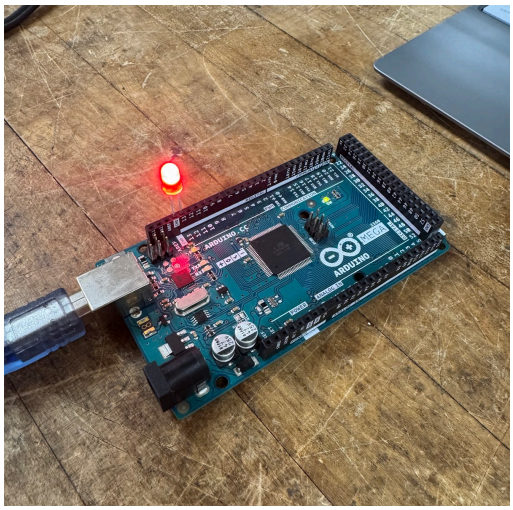
4. *What is the duty cycle? How can one change the duty cycle of the LED's blinking? What would you do so the LED blinks at a frequency of 30 times every minute with a 75 percent duty cycle?*

- A duty cycle is the percentage of the time an object is on such as the LED being on versus the time it is off during a period of time during the tutorial.
- The duty cycle of the LED's blinking can be changed by changing the delay time in the loop function code. By changing the length of the delay, there would be a longer amount of time on vs. off depending on whether the first or second delay is adjusted.

- To have a 75% duty cycle then 75% of the time the LED would be turned on and 25% of the time, the LEd would be turned off.
    - The frequency desired is 30 cycles per minute so the period of the motion would be 60 seconds/30 which gives a period of two seconds.
    - Then, to change the duty cycle to 75%, we can change the delay times in the Arduino code to 1500ms for the (led, HIGH) mode when the LED is on and 500 ms for the (led, LOW) mode when the LED is turned off.

Screenshot of Arduino code and Image of Arduino and circuit setup:





Tutorial #2: Buttons and PWM

Complete the following tutorial: https://howtomechatronics.com/tutorials/arduino/buttons-pwm/
Directions are provided video and textual formats. You should watch the entire video and periodically pause it to simulate the Arduino coding on a separate laptop or to physically replicate the mechatronics exercises
and demonstrations.
NOTE: The circuit diagram for this tutorial is incorrect. The resistor should connect the button (on the side connected to the Arduino's power) to ground to prevent shorting the Arduino's power to ground. Also, connect another resistor from the negative terminal of the LED to ground to prevent frying the LED.

1. *Is the button provided to you in a normally ON or OFF position?*

The button is normally in the OFF position.

2. *Why is it necessary to pair the LED with a resistor? Is it in series or parallel with the LED?*

It's necessary to pair the LED with a resistor because the LED has very low resistance internally itself, so the current in the system might get too high. Adding a resistor is necessary to limit the current that flows through the circuit, which is needed because otherwise, excess current will damage the LED. The LED is in series with the resistor.

3. *If the button is pressed, does the pin read LOW or HIGH? Will the LED be turned ON or OFF?*

When the button is pressed, the LED turns ON and the pin will read HIGH.

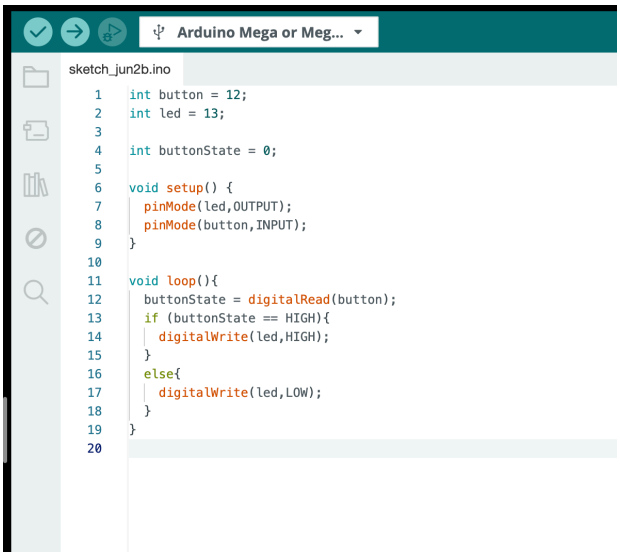
4. *What is a PWM signal? If the HIGH voltage of the signal was 10 V and 8 V was needed, what should the duty cycle be?*

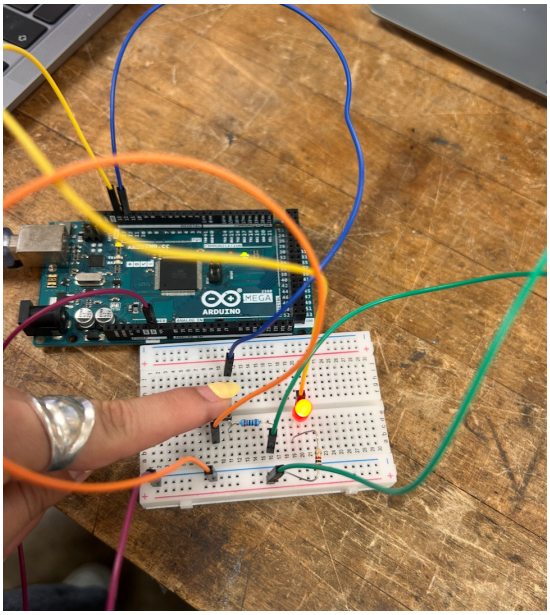
PWM stands for Pulse Width Modulation. It is a square digital signal that averages the power delivered and in the tutorial it is involved due to the write function in the code. If the HIGH voltage of the signal is 10 V and 8 V then the duty cycle is 80% since 8/10 gives a 80% value which indicates that 80% of the high voltage is being used.

5. *What is a digital signal? What is an analog signal? How can the brightness of the LED be changed in real-time with each type of signal?*

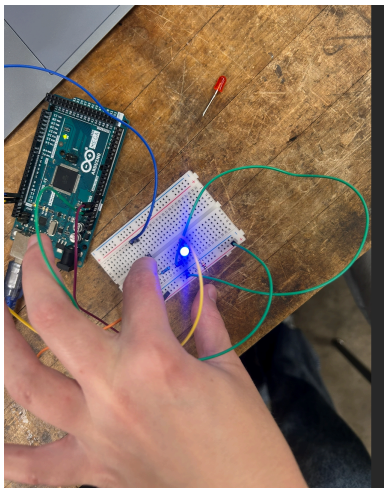
- A digital signal shows the information in discrete values. So in this case for the tutorial, it only contains two discrete states, HIGH and LOW which are being read and interpreted by the Arduino.
- An analog signal is a continuous signal that will vary in voltage between 0V to 5V. These values are read and interpreted by the analog Arduino pins and can be read into the system using Analogread (such as in tutorial #3).
- The brightness of the LED can be changed with an analog signal by using analogWrite which may be able to pass in different brightness values as an integer is converted into a voltage that is being sent to the LED. A low voltage would mean a low brightness or a higher voltage leading to a higher brightness.
- If we decided to use digital signals instead to change the brightness of the LED, then we can vary the duty cycle, or the amount of time the LED is set to HIGH or LOW in the Arduino code program.

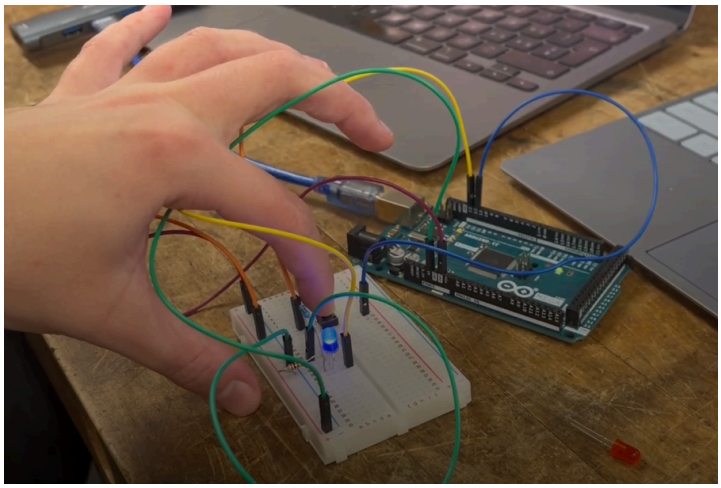
Example 1: The Arduino with the button and LED subcircuits and the code associated:

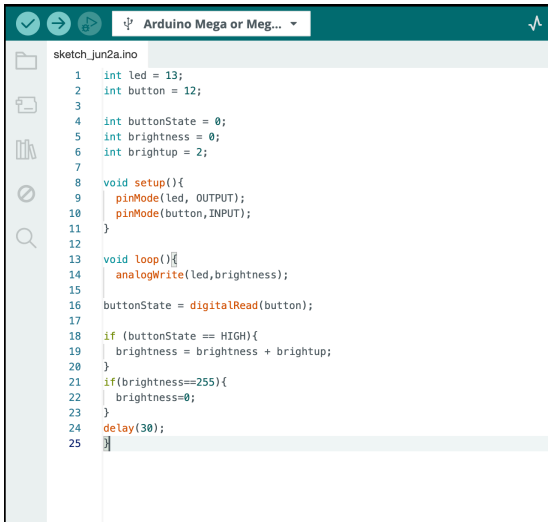




Example 2: LED at different brightnesses and Arduino Code:







Tutorial #3: Analog Inputs

Complete the following tutorial: https://howtomechatronics.com/tutorials/arduino/analog-inputs/ Directions are provided in video and textual formats. You should watch the entire video and periodically pause it to simulate the Arduino coding on a separate laptop or to physically replicate the mechatronics exercises and demonstrations.

1. *What is a voltage divider? Why is it useful?*

- A voltage divider is a type of circuit that produces an output voltage that is a fraction of the input voltage.
- A voltage divider is useful because it lets you create a lower voltage from a higher voltage. It also lets you provide different voltages to different components of a circuit from one common voltage source.

2. *Please explain the reasoning behind mapping analogRead values when using them to write an analog signal.*

The analogRead is used to receive the different voltage levels and then it reads the input voltage to convert it into the integer values. The conversion goes from 0 to 5 V to 0 to 1023 integer values that can be read by the digital arduino board. In the tutorial, the analog read function is what allows the board to take in the voltage differences from the potentiometer or photo-resistor and then it converts them into integer values which we were reading from the serial monitor output on the arduino app.

3. *Aside from the photoresistor in the video, please name at least two other types of variable resistors that exist.*

Thermosistors (which would take in a temperature input to analog), potentiometers (which were also used in the tutorial and involve a rotating piece), and force-sensing resistors (such as something which could detect force changes in a beam).

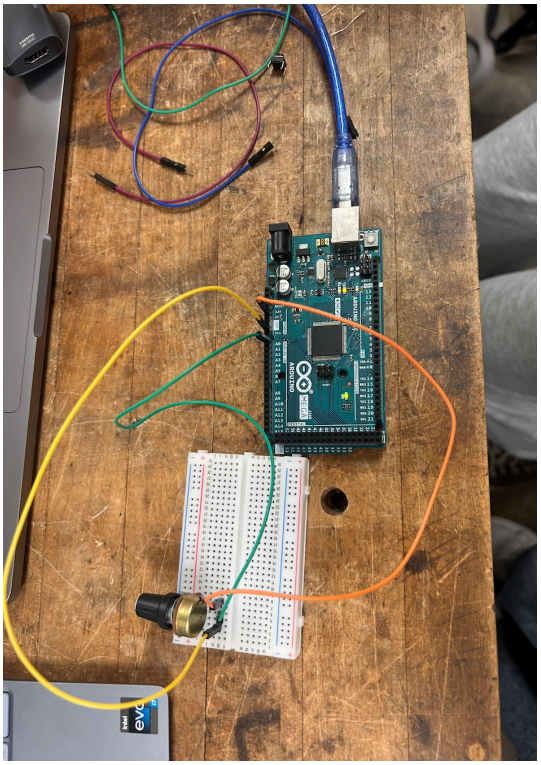
4. *Please record the values of the sensor value on the serial monitor when: covering the photoresistor with your hand, in a normally lit room, and with the provided flashlight.*

On average, the values of the sensor read about in the 90-96 integer value range for the normally lit lab room, about 7-10 when fully covering the photoresistor with a finger, and then about 290-296 range when a phone flashlight was used near it.

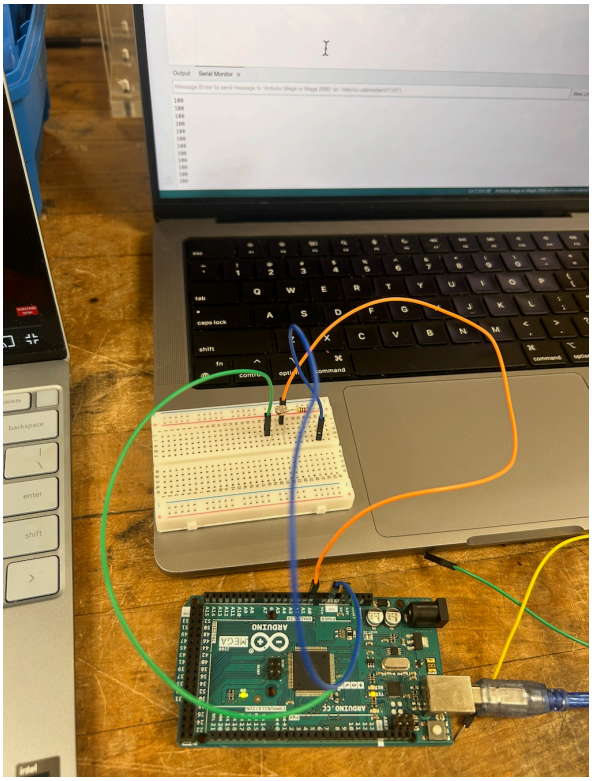
Video of our circuit board with the photoresistor:
https://drive.google.com/file/d/1T4ikknmU7SDiA8AKJ8N9HJyiFrpWT1Jg/view?usp=sharing

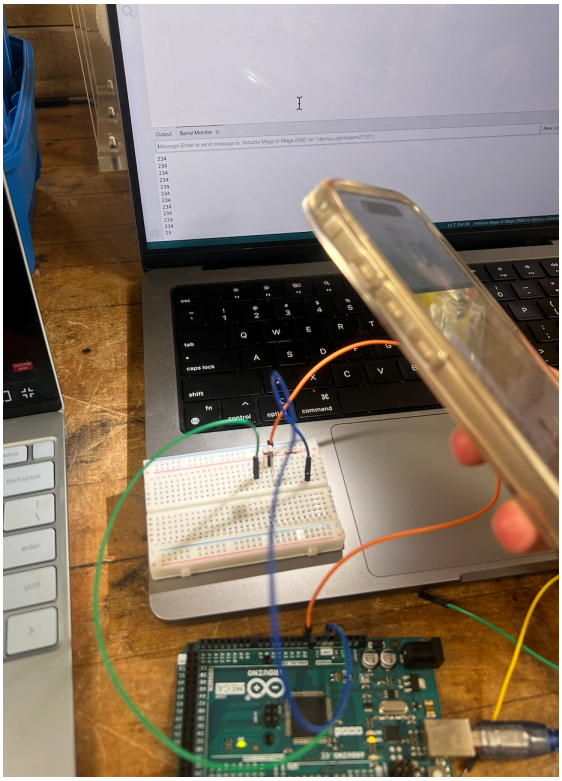
Please take a screenshot of your circuit for each of the three examples. For the potentiometer-controlled LED circuit, please provide two screenshots at different brightnesses. Please also provide a screenshot of your Arduino code for each example as part of your document.

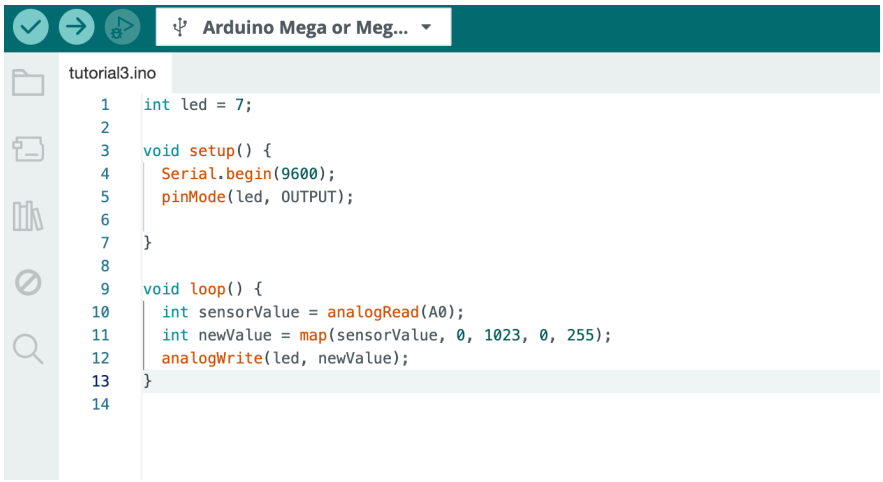
Example 1: Potentiometer (with no LED)



Examples 2: Photoresistor

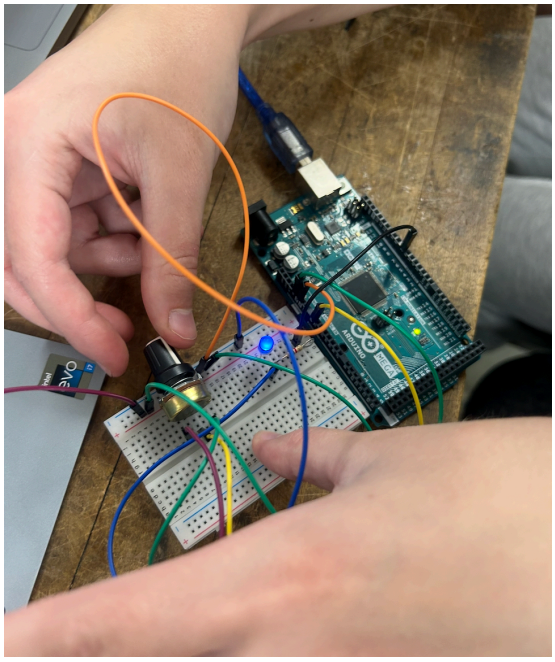


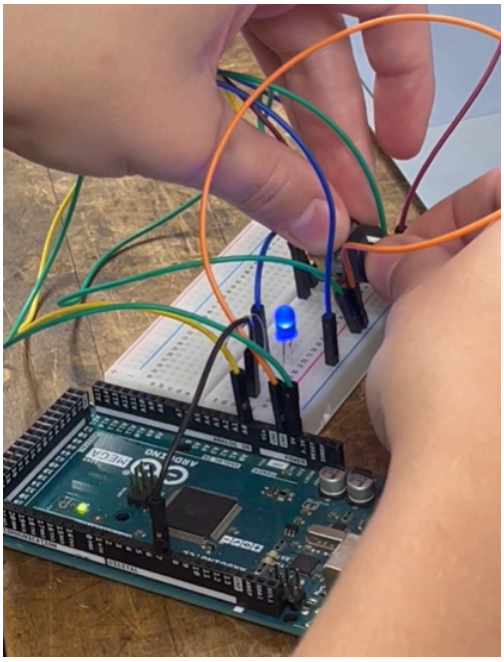


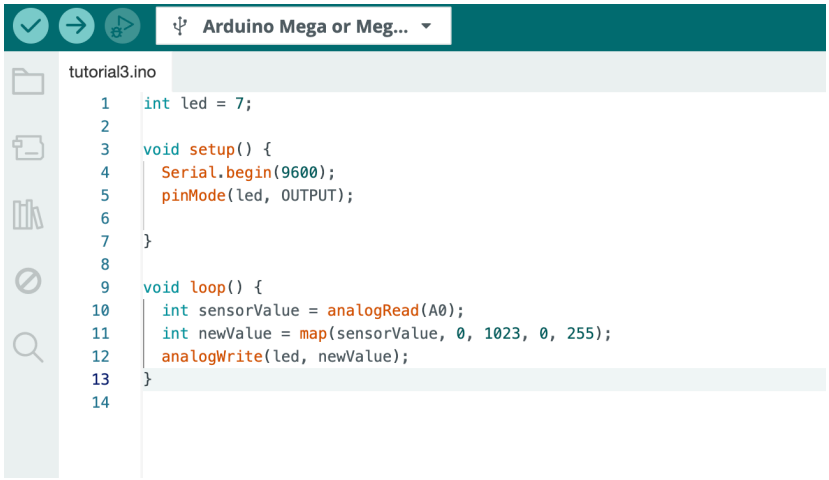


Example 3: Potentiometer with the LED

Video of our circuit: https://drive.google.com/file/d/1z_80TD9yb0ElGUYoNVjN2es0V79C6Szy/view?usp=sharing







Tutorial #4: DC Brushed Motors

Complete the following tutorial: https://howtomechatronics.com/tutorials/arduino/motors/ Directions are provided in video and textual formats. You should watch the entire video and periodically pause it to simulate the Arduino coding on a separate laptop or to physically replicate the mechatronics exercises and demonstrations.

1. *What is a transistor? Which direction does current run in an NPN transistor? How would one change the current value through the transistor?*

- A transistor controls a signal based on the signal sent to the gate. The current runs from the input signal, collector, to the output signal location, the emitter. As a subsystem, the transistor is a small semiconductor device that helps control the flow of current. It is made up of three different terminals which help it control the current: the base, collector, and emitter.
- In a NPN transistor, current runs from the collector where the signal is being input to the emitter where the signal is being output. This happens because typically, the collector is at a higher voltage than the emitter so the current wants to move from a high voltage area to a lower voltage area.
- To change the current value through the transistor, you could increase or decrease the base current which determines the overall current flow in the transistor section.

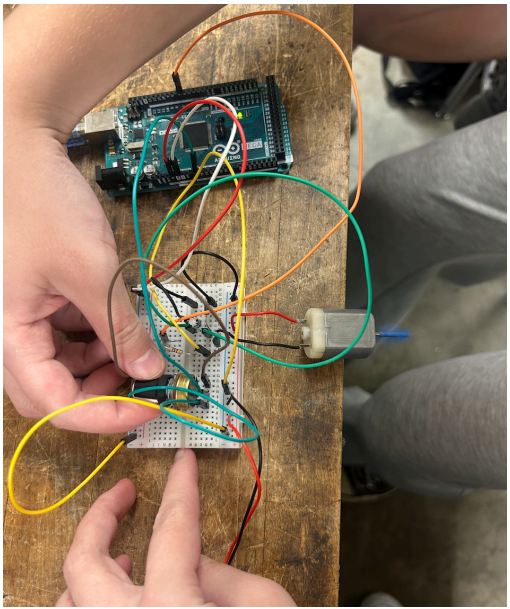
2. *Why is a 1 kΩ resistor connecting the pin to the base of the transistor? What purpose do the diode and the capacitor serve?*

- The 1kΩ resistor connects the pin to the base of the transistor to limit the base current and protect the Arduino from the 12 volts.
- The diode protects electronic components by giving it a path for the current when the transistor switches off since the motor acts like an inductor. The capacitor helps protect from noise by suppressing voltage changes that happen suddenly such as voltage spikes that might reach higher or lower levels than would be safe for the circuit and they overall are helpful to the smoothing of the electrical signals.

3. *What is the difference between a DC motor and a servo motor? Think of possible use cases for each.*

   A DC motor spins continuously when you apply high voltage while the servo motor spins based on the command or signal sent. The direction of the DC motor depends on polarity and the DC motor uses two wires in the Arduino tutorial setup. A good example for a DC motor is for fans or wheels on a robot since they are continuous motions. The servo motor consists of three wires when it is used in the tutorial and there is feedback that happens inside the motor while it is working. A good example of a servo motor in use is with a robotic arm when a more specific angle of motion is desired in the action such as a rotation by 45 degrees.

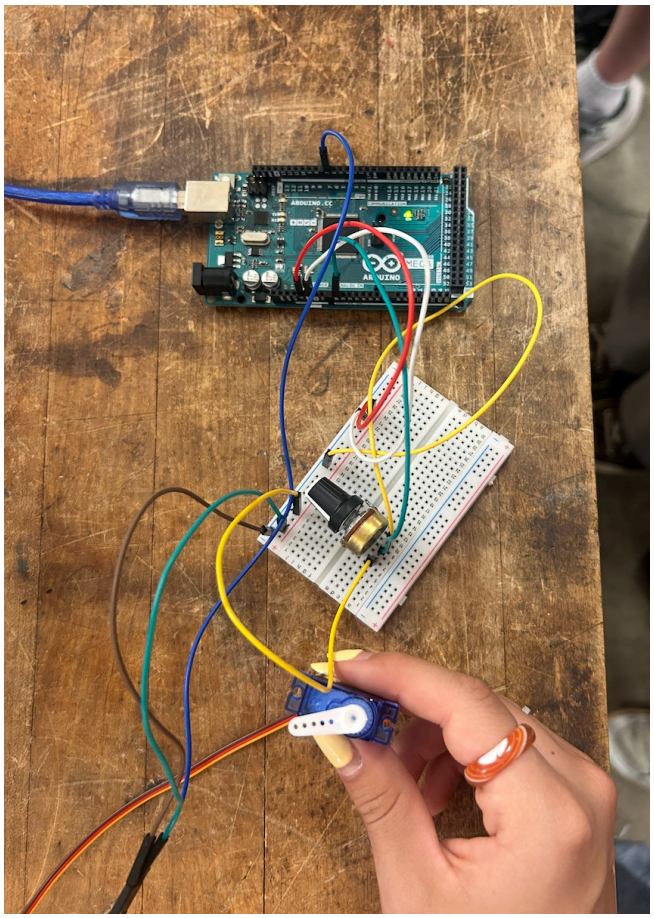
DC motor circuit and Arduino code:
Video of our circuit:
https://drive.google.com/file/d/1ey_uBFxRShy_Ltlbwv7du6lsxST-GVyd/view?usp=sharing



dc_brushed_motor.ino

```
1  int pwmPin = 7;
2
3  void setup() {
4    Serial.begin(9600);
5    pinMode(pwmPin, OUTPUT);
6  }
7
8  void loop() {
9    int potValue = analogRead(A0);
10   int newpotValue = map(potValue, 0, 1023, 0, 255);
11   analogWrite(pwmPin, newpotValue);
12 }
13
```

Servo Motor circuit and Arduino code:
Video of our circuit:
https://drive.google.com/file/d/1pnl0udCQWu3t0asTSQ9plYSiURwDtiE6/view?usp=sharing



servo_motor.ino

```
#include <Servo.h>

Servo myServo;

void setup() {
  myServo.attach(7);
}

void loop() {
  int potValue = analogRead(A0);

  int angleValue = map(potValue, 0, 1023, 0, 180);
  myServo.write(angleValue);
  delay(10);
}
```

Tutorial #5: Stepper Motor with A4988 Driver and Arduino

Complete the following tutorial:
https://howtomechatronics.com/tutorials/arduino/how-to-control-stepper-motor-with-a4988-driver-and-arduino/ Directions are provided in video and textual formats. You should watch the entire video and periodically pause it to simulate the Arduino coding on a separate laptop or to physically replicate the mechatronics exercises and demonstrations.

1. *What voltage does the Arduino provide? Why can the stepper motor not run on this voltage?*

The Arduino provides 5 volts with the power pin that has 5V on it and it also has a pin that can output 3.3 volts. The stepper motor cannot run on this voltage as it requires a higher minimum voltage in order to run and according to the data provided in the tutorial video, the minimum operating voltage is 8V. The arduino board alone would not be able to provide the full 8V which is why we use the additional microstepping board.

2. *What are some essential characteristics when choosing an appropriate stepper motor? What are they for your stepper? What does NEMA XX stand for?*

Some characteristics for choosing a stepper motor are making sure that the voltage is within the voltage provided in the circuit, what steps/smoothness is desired from the motor, and the efficiency that is given by the motor. The NEMA XX stands for the type of stepper motor that is being used and more specifically the xx represents a dimension on the motor at its front.[1] The following are the specifications for the stepper motor in our kit box.

[1] Source: https://www.electromate.com/mechatronic-automation/mechatronic-automation-components/stepper-motors/nema-stepper-motors/

3. *Determine which colored wire pairs on the provided stepper motor are the same connection internally (e.g., 1A and 1B are the same wire). How can you tell physically?*

1A and 1B are the same connection internally and the 2A and 2B are the same connection internally. For both this means they are being connected to the same coil and when following the tutorial, it was possible to tell which are the same connection as they are output from the motor in 4 pins where the two pins next to each other were 1A and 1B and then 2A and 2B in sequential order. While the motor indicated slightly different lettering, by pairing the pins the wire order was maintained.

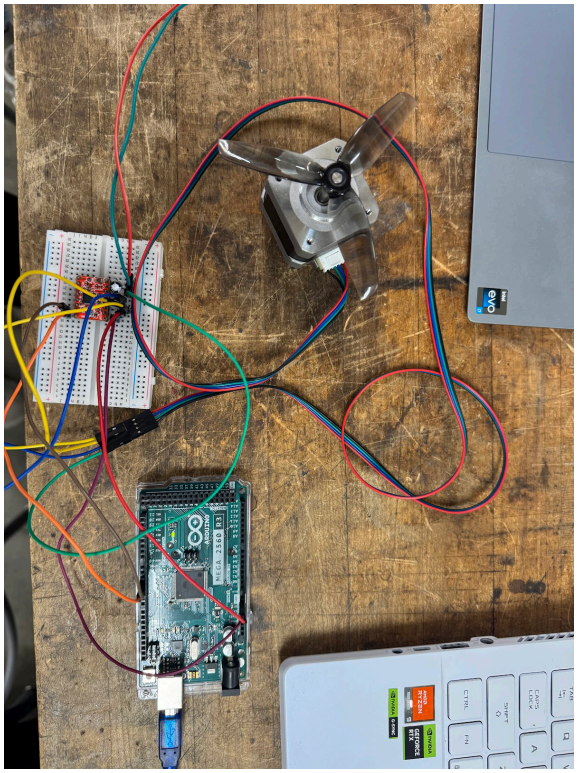
4. *If the stepper turns 1.8 ° operating at full step, how much does it turn at a quarter step? Explain why fractional steps are useful. How can one control the fractional step setting of the stepper?*

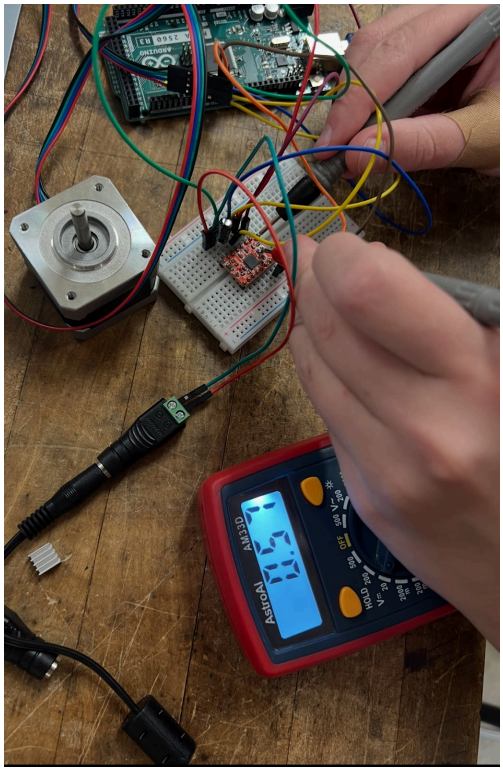
If a full step is 1.8 degrees turns, when a quarter step would be 25% of the the full step which is 1.8*0.25 = 0.45 degrees turns. The fractional steps may be useful for more accurate positioning as the total number of steps of the stepper motor increases with the smaller step degrees turn size (since more steps are now necessary to meet the full step). This is controlled by the pins as the tutorial has a chart that explains how different combinations of the microstepping MS1, MS2, and MS3 pins give different fractional steps.

5. *Why is it essential to place a capacitor between terminals of a power supply when feeding power to your circuit?*

A capacitor is an important component to ensuring that sudden voltage changes do not negatively impact the circuit. By having the capacitor after the power supply but before feeding into the circuit it ensures that if the voltage were to drop suddenly from the battery or increase, these changes don’t immediately go into the circuit but rather they go gradually.

Circuit with the stepper motor and the Arduino Code:





stepper_motor.ino

```ino
const int stepPin = 3;
const int dirPin = 4;

void setup() {
  pinMode(stepPin, OUTPUT);
  pinMode(dirPin, OUTPUT);
}

void loop() {
  digitalWrite(dirPin,HIGH);
  for(int x = 0; x < 200; x++) {
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(500);
    digitalWrite(stepPin, LOW);
    delayMicroseconds(500);
  }
  delay(1000);

  digitalWrite(dirPin,LOW);
  for(int x = 0; x < 400; x++) {
    digitalWrite(stepPin,HIGH);
    delayMicroseconds(500);
    digitalWrite(stepPin,LOW);
    delayMicroseconds(500);
  }
  delay(1000);
}

```

Tutorial #6: Brushless Motors

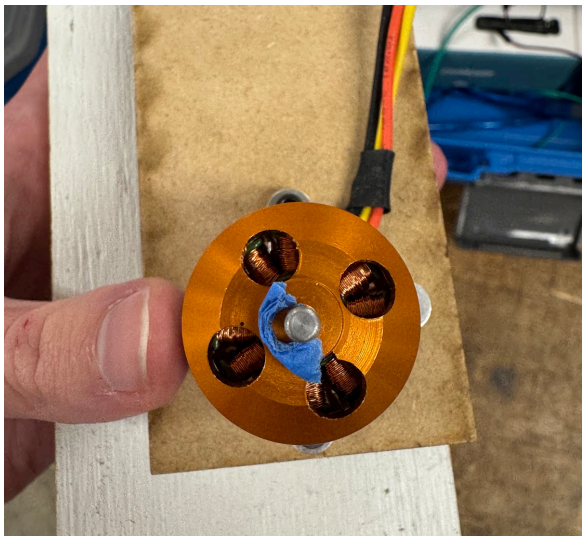
Complete the following tutorial at the URL:
https://howtomechatronics.com/how-it-works/how-brushless-motor-and-esc-work/
Directions are provided video and textual formats. You should watch the entire video and periodically pause it to simulate the Arduino coding on a separate laptop or to physically replicate the mechatronics exercises and demonstrations.

1. *Briefly explain the theory behind the rotation of a brushless motor. How is this different from a brushed motor?*

The basic idea of the rotation of the brushless motor is that there is a permanent magnet inside the motor and when current is sent into the coils inside the motor (the stator), the coils become magnets as well that interact with the permanent magnet. The interaction leads to the continuous motion inside the motor and this is different from the brushed motor where the contact of the commutator and the coils resulted in the rotation.

2. *How many poles does your brushless motor have? How can you physically tell?*

When counting the number of coils on our motor, we had 12 coils which means that if three coils are connected in their action, there would likely be 12/3 = 4 poles inside the motor. A more precise way to tell this would be if we were able to remove the outer section and look inside the motor to count the number of magnets in the motor (which would be at its center). Another way to tell is the Hall effect sensors (discussed in question 4), where the number of pulses per revolution is the number of poles.



3. *What does ESC stand for? Why is it needed in combination with a brushless motor? Why is it important to pair the correct ESC with the appropriate motor and battery?*

The ESC stands for Electronic Speed Controller and it is used with the brushless motor since it consists of mosfets and various electronic components that provide the necessary electrical current to the motor. More specifically, it takes in the position, power, and other circuit information to send correct amounts of current into the coils which then determines the rotation of the coils. The ESC we used had to match the motor and battery as it would be unable to accurately send a signal to the motor as it is specifically designed for a certain motor and battery current inputs and outputs.

4. *What are Hall Effect sensors? What do they do and what are their benefits in their inclusion?*

The Hall Effect sensors are in the brushless motor and the purpose of the sensors is keeping track of the magnetic field inside the motor. By keeping track of the polarity, the sensors can tell when it is a negative or positive pole which is beneficial as the information is sent to the ESC which can then control which coil it would need to control next in the sequence of the coils involved.

5. *What is the meaning of a KV rating? If used in a drone, should the motors have a higher or lower KV rating? If used in an electric bike, should the motors have a higher or lower KV rating? Please briefly explain.*

A KV rating for a motor is by definition the speed of the rotating motor over the voltage that is being applied to the motor.[2] Since a drone uses propellers in order to generate to overcome its own weight, we would expect that the propellers would have to be rotating very fast. In order to be an effective motor, this means the drone would likely need a high KV rating. On the other hand, the electric bike would not need a rotational speed that is as fast as the drone propeller speeds so we would expect that the KV rating would be lower.

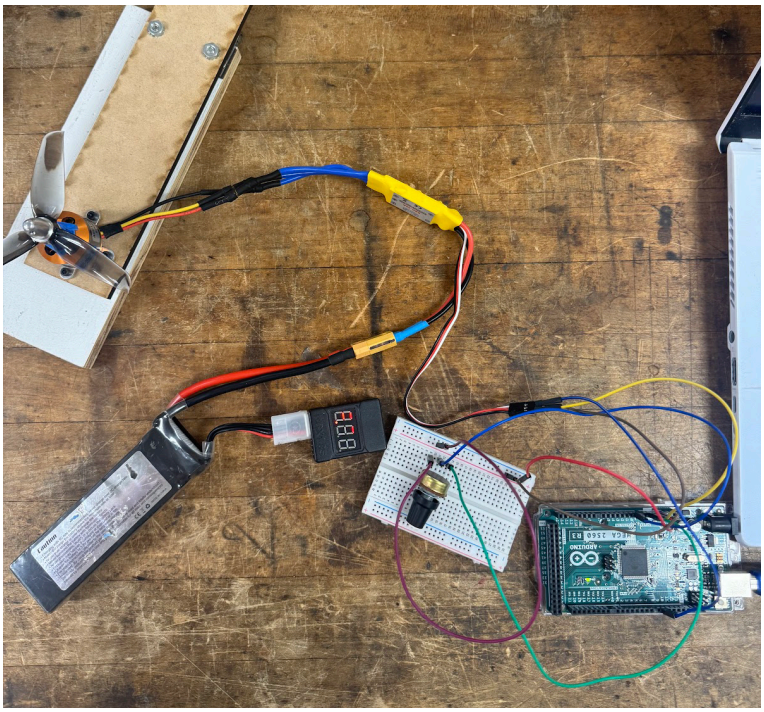
[2] Source: https://dronenodes.com/brushless-motor-kv-rating-explained/

6. *What is meant when a battery is 10S4P? How much voltage is this battery supplying? How low can this battery go before it is considered "dead"? What approximate maximum voltage can this battery be charged to?*

The designation of 10S4P is based on how the cells inside the battery are connected so 4 of them would be in parallel and 10 would be in series. As discussed in the LiPo battery tutorial, a battery cell has a nominal voltage that is around 3.7V so the 10 in series would be supplying a voltage of up to around 37 V. The critical minimum voltage is 3.0 V per cell which would mean that a 10 cell series would be considered at a minimum voltage or nearly dead state at around 30 V. The maximum voltage each cell can be charged to is 4.2 V per cell which would be an absolute maximum voltage of 42 V when charging the whole battery fully.

Circuit and Code:

Note: we attached the brushless motor to the mounting mechanism from the start of the tutorial



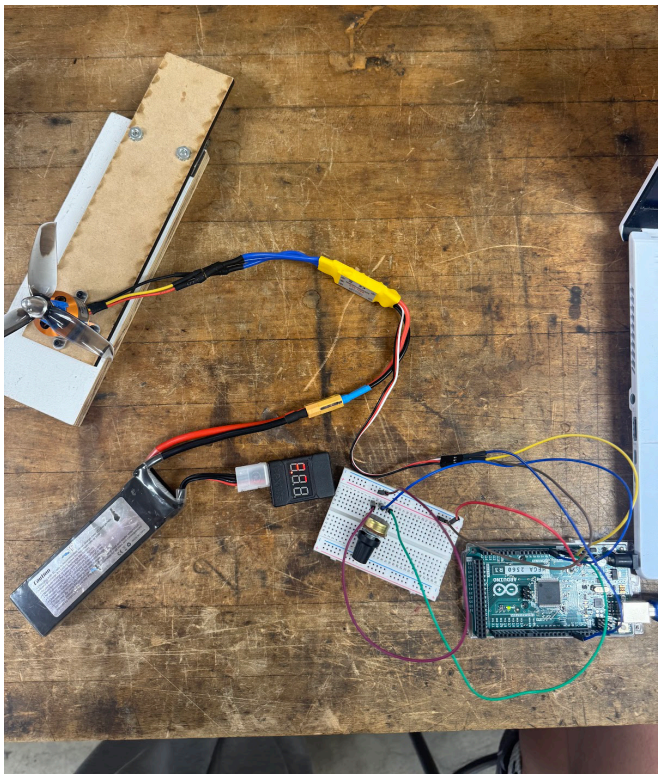
brushless_motor.ino

```
#include <Servo.h>

Servo ESC;

int potValue;

void setup() {
  ESC.attach(9,1000,2000);
}

void loop() {
  potValue = analogRead(A0);
  potValue = map(potValue, 0, 1023, 0, 180);
  ESC.write(potValue);
}
```

8. (Extra Credit) Attach the brushless motor to the provided mounting mechanism with the propeller. At what voltage (and hence RPM) does the motor's lift to weight ratio surpass 1?

For our setup we noted that the motor's lift to weight ratio did not end up surpassing one as the propeller was unable to provide enough lift to lift up the full wooden structure it was mounted to. In order to figure out this final ratio we would have had to plot the analogread values and determine the value when the motor lift became greater than the weight.

Video of our circuit:
https://drive.google.com/file/d/1SiNs52QXBdDIAhI3L2-DQ_BY6X_nUKTG/view?usp=sharing



brushless_motor.ino

```
#include <Servo.h>

Servo ESC;

int potValue;

void setup() {
  ESC.attach(9,1000,2000);
}

void loop() {
  potValue = analogRead(A0);
  potValue = map(potValue, 0, 1023, 0, 180);
  ESC.write(potValue);
}
```

Tutorial #7: Arduino DC Motor Control Tutorial – L298N | PWM | H-Bridge

Complete the following tutorial: https://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/ Directions are provided video and textual formats. You should watch the entire video and periodically pause it to simulate the Arduino coding on a separate laptop or to physically replicate the mechatronics exercises and demonstrations. Please only complete the exercises discussed up to the 6:25 mark in the video and skip the portion at the end about building a small car.

1. *How does one control the direction that a DC motor rotates? How is this implemented?*

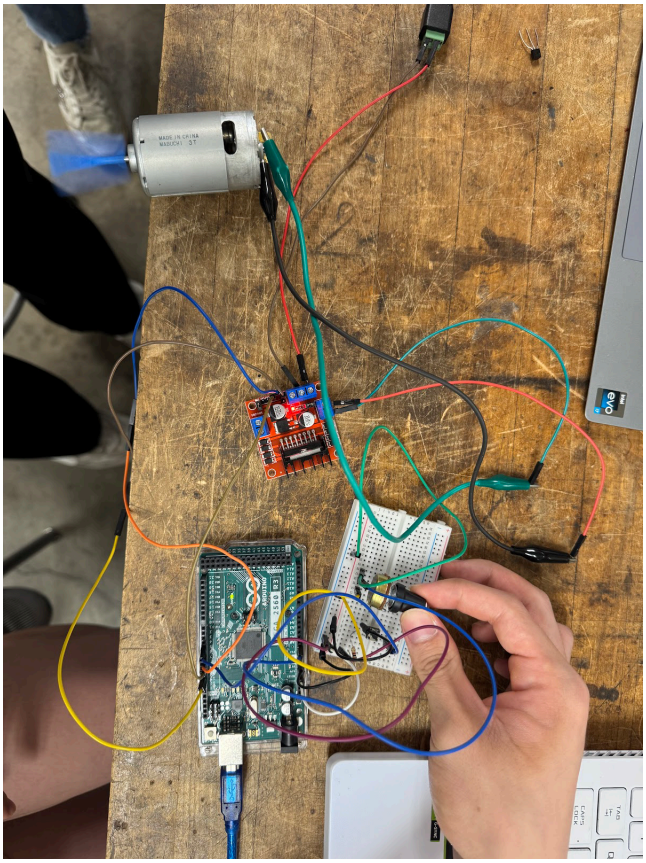
The DC motor determines its direction of motion based on how the current is coming into the motor so by switching the positive and negative wire connections on the motor it is possible to switch its direction. This is due to the structure of the DC motor internally when it is rotating. The switching of the positive and negative is implemented by the H-bridge. For the H-bridge described in the tutorial, when one input is set high and the other low the motor rotates in one direction and switching the high to low and the low to high gives opposite direction.

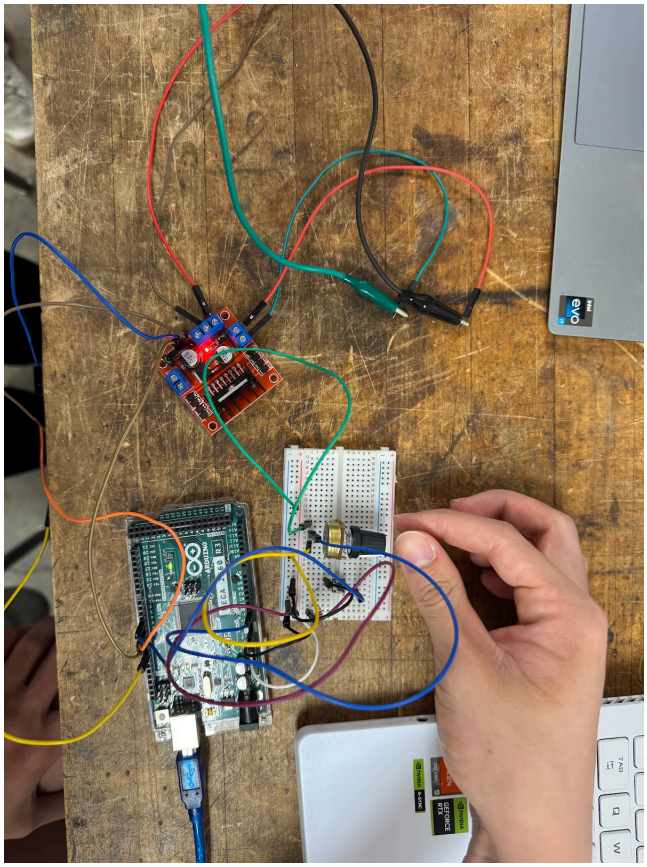
2. *How are Li-ion batteries different than LiPo batteries? What is the most common form factor for Li-ion batteries?*

The main difference in LiPo or Li-ion batteries is the material that they are made of internally: Li-ion batteries rely on electrolytic chemical reactions with lithium while the lipo uses a different solid material. The most common form factor for the Li-ion batteries, in terms of their shape, is cylindrical.[3]

[3] Source for most common type of lithium battery: https://www.eblofficial.com/blogs/blog/lithium-ion-battery-sizes-uses#:~:text=What%20are%20the%20standard%20lithium,chart%20to%20compare%20options%20easily.

Image of the circuit and the Arduino Code:





h_bridge.ino

```ino
#define enA 9
#define in1 6
#define in2 7
#define button 4

int rotDirection = 0;
int pressed = false;

void setup() {
  pinMode(enA, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(button, INPUT);
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
}
```

```ino
void loop() {
  int potValue = analogRead(A0);
  int pwmOutput = map(potValue, 0, 1023, 0, 255);
  analogWrite(enA, pwmOutput);

  if (digitalRead(button) == true) {
    pressed = !pressed;
  }
  while (digitalRead(button) == true);
  delay(20);

  if (pressed == true  & rotDirection == 0) {
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    rotDirection = 1;
    delay(20);
  }

  if (pressed == false & rotDirection == 1) {
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
    rotDirection = 0;
    delay(20);
  }
}
```