Benzon Carlitos Salazar
CS412 Week 9 Assignment

Task 1: Servo Motor and Potentiometer

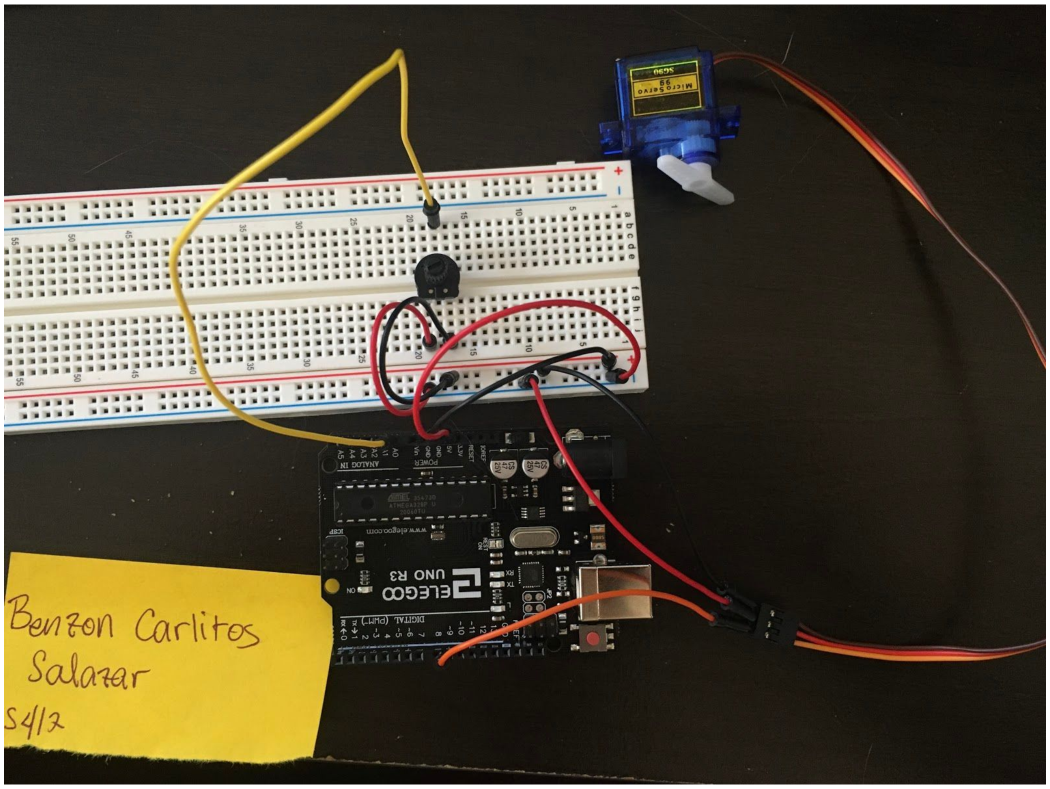
Summary:

This task involves integrating the servo motor and potentiometer. The goal of this is to control the speed of the Servo Motor. We will accomplish this by controlling the delay speed between each `write()` function of our servo, where the servo will move from 0°-180°.

Hardware Requirements:

- 1 x Arduino UNO Controller Board
- 8 x Breadboard Jumper Wires
- 1 x Tie-points breadboard
- 1 x Servo Motor
- 1 x Potentiometer 10K

The Circuit:



Circuit Explanation:

All the **red jumper wires** are the **power connections**, this means that the Servo motor’s red wire and the potentiometer’s red wire are connected to the Arduino’s 5V power pin. For the potentiometer, the power pin controls the voltage of the system, and for the Servo the power is constantly applied. The **black jumper wires** are the **ground connections** from both the Servo and potentiometer to the Arduino.

The **yellow jumper wire** connects the potentiometer’s middle pin to the analog input in the Arduino. This is important because by turning the shaft of the potentiometer, we can change the amount of resistance on either side of the wiper that’s connected to the middle pin of the potentiometer. This can change the relative “closeness” of the middle pin to 5V and ground, which gives us different analog inputs. These changes in analog inputs is what we’re going to take advantage of when changing the speed at which the Servo motor changes angles. When we turn the shaft of the potentiometer, we change the voltage to 0V, and our Ardunio reads 0. And when turned the other way, we change the voltage to 5V, and our Arduino reads 1023. We divide the range of 0-1023 by 4, giving us a range of 0-255 instead, for our delay speed.

The **orange jumper wire** connects the Servo motor’s control signal (PWM) to the Arduino’s digital pin 9. We use pin 9 so that we can program a PWM (pulse-width modulation) signal to the Servo, causing a loop of pulses of variable width controlled by the `delay()` function. We will set the position angle of the Servo of 0° and 180°. The range of voltage that we control from the potentiometer will be the delay control of our Servo motor.

Code:

```
#include <Servo.h>
Servo myservo;

int potpin = 0;
int servopin = 9;
int val;

void setup() {
  myservo.attach(servopin);
}

void loop() {
  val = analogRead(potpin);

  myservo.write(180);
  delay(val);
  myservo.write(0);
  delay(val);
}
```

Task 2: Servo Motor and Button

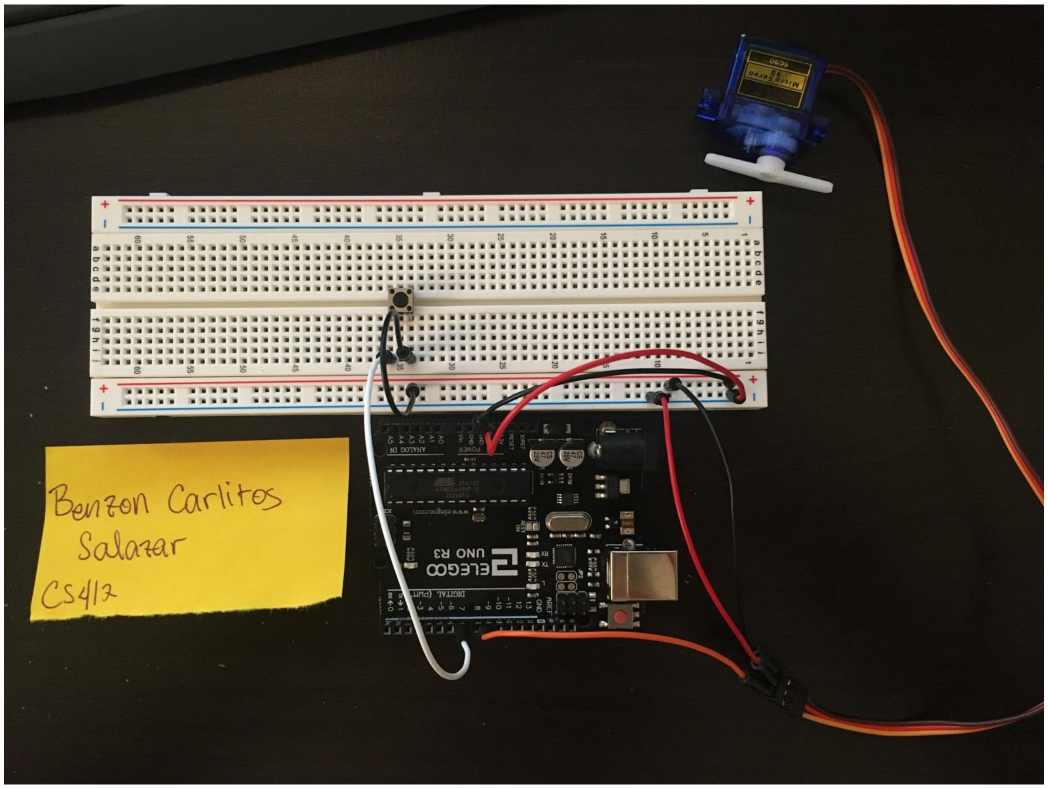
Summary:

This task involves integrating the button with the Servo motor. The goal is when the button is pushed, the Servo's angle will move to 90°, and when the button is released, we change the Servo's angle back to 0°.

Hardware Requirements:

- 1 x Arduino UNO Controller Board
- 7 x Breadboard Jumper Wires
- 1 x Tie-points breadboard
- 1 x Servo Motor
- 1 x Button

The Circuit:



Circuit Explanation:

All the **red jumper wires** are the **power connections**, this means that the Servo motor's red wire is connected to the Arduino's 5V power pin. The **black jumper wires** are the **ground connections** from both the Servo and button to the Arduino.

The **white jumper wire** connects one of the button's legs to the Arduino's digital pin 7. While the **orange jumper wire** connects the Servo motor's control signal (PWM) to the Arduino's digital pin 9. We use pin 9 so that we can program a PWM (pulse-width modulation) signal to the Servo. Instead of causing a variable delay, we take advantage of the boolean values applied to the button when the button is pressed and unpressed. This boolean value can be used to tell the Servo, "If we receive a signal from the button, turn 90° and stay there. Otherwise, when the button is released causing no signal, go back to 0° and stay there."

As for the code, we can see that the setting of our `pinMode()` is set to `INPUT_PULLUP`. This inverts the behavior of the `INPUT` mode, which means that HIGH == off, and LOW == on.

Code:

```
#include <Servo.h>

int servoPin = 9;
int switchPin = 7;

Servo servo1;

void setup() {
  servo1.attach(servoPin);
  pinMode(switchPin, INPUT_PULLUP);
}

void loop() {
  setServoMotor(digitalRead(switchPin));
}

void setServoMotor(boolean reverse) {
  if(reverse) {
     servo1.write(90); // 90 degrees
  }
  else{
     servo1.write(0); // 0 degrees
  }
}
```