Operation: SWTCHCTRL

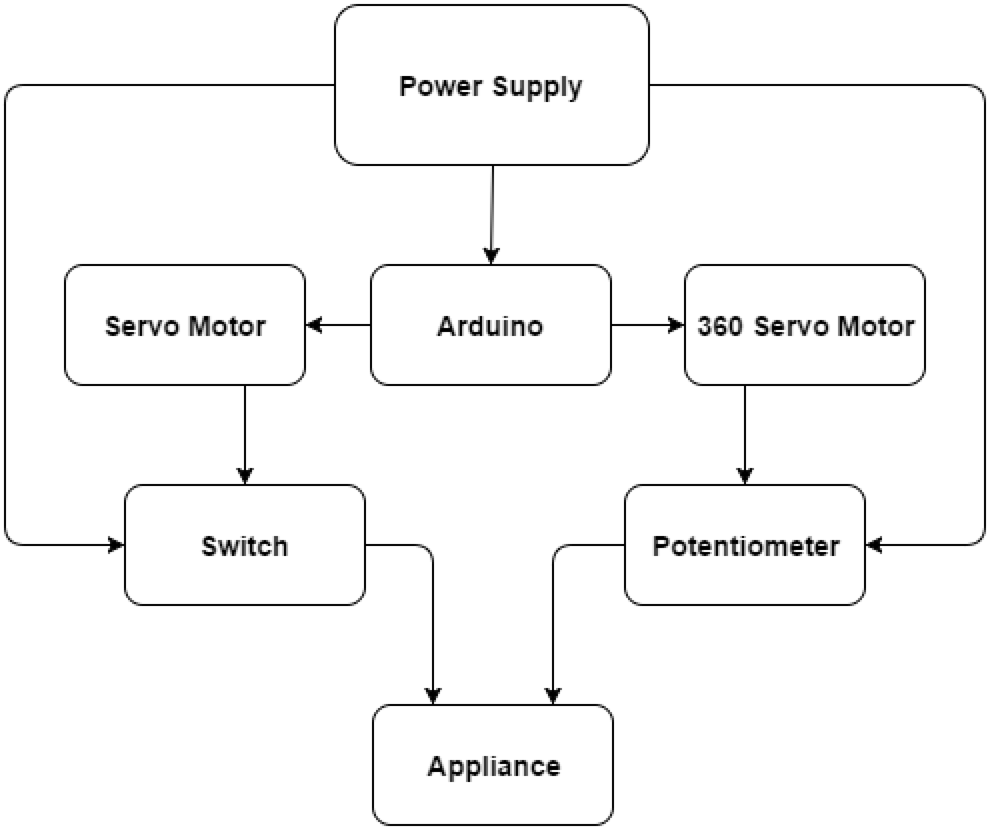
Introduction

The problem being addressed in our project was the need for more control when using a voltage dependent device, such as a lamp or a fan. The goal of our project was to create a method of controlling the brightness of a lamp and toggle its state through the use of an arduino and eventually through an app on a phone.

The solution we proposed was to use a common wall switch and dimmer, such as those that could be found in one's house, and wire them to an extension cord. This would provide the mechanics of controlling the voltage through the cord. In order to control the switches' positions, we would attach servos to each switch, which we could then program to move to the desired position.

Design

Block Diagram:



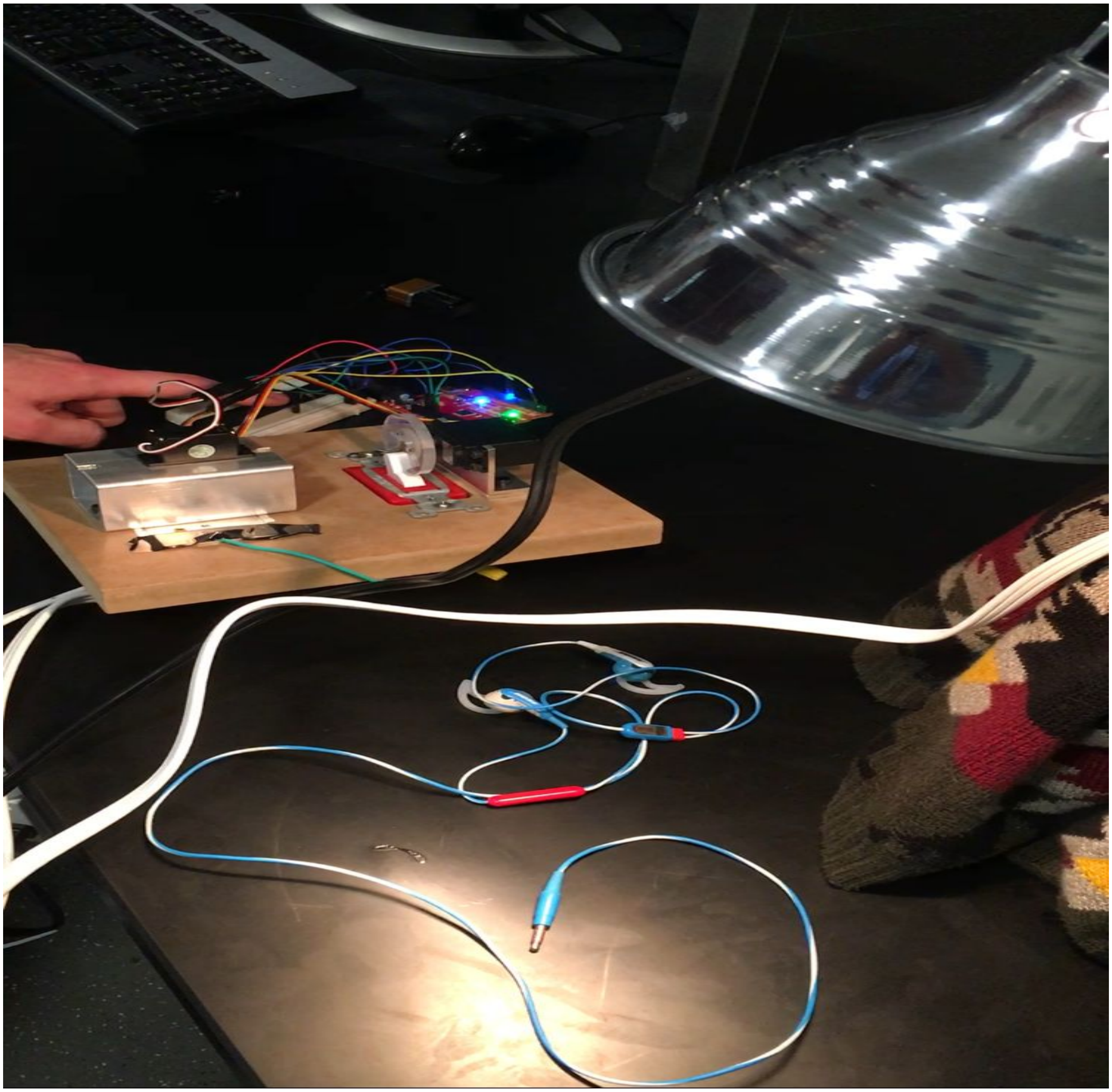
Power Supply: 110v wall outlet for switches, 9v battery for arduino

Arduino: controls servo position based on external input (we used a button for demo purposes)

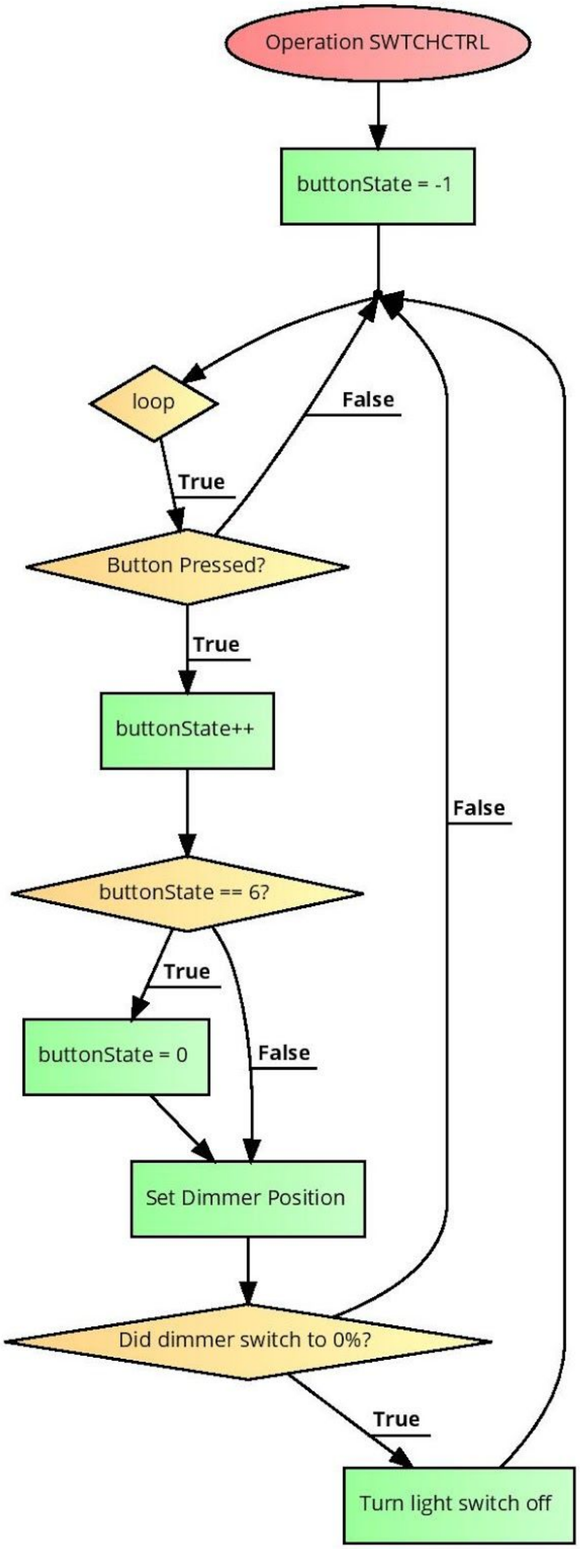
Servo Motors: move the switch and potentiometer to the correct positions

Switch: toggles voltage to appliance
Potentiometer: limits voltage to appliance

Picture of Demo:



Flow Chart of Software:


Operation SWTCHCTRL
buttonState = -1
loop
True
False
Button Pressed?
True
buttonState++
buttonState == 6?
True
False
buttonState = 0
Set Dimmer Position
False
Did dimmer switch to 0%?
True
Turn light switch off

Code Used:

See appendix.

Results

We accomplished the main part of what we wanted to do, which was to control appliances remotely. Our finished project allows us to dim lights or turn on and off devices using an arduino, which we can control using a wide variety of inputs. Our results were qualitatively better than manually controlling the switches but not as good as using transistors to accomplish the same thing. Our servos were fast and accurate but could never be near instantaneous or super precise like a transistor. Our results don't really have any data involved because our goal was to automate a task, so a quantitative analysis would not be very meaningful.

Future Work

The next step that we would take with this project is devising a better method of inputting the desired location. With our current setup, we only had a button. This required us to cycle through predefined states until we reached the one we wanted. In the future, we would like to be able to control the system with an app on one's phone, allowing us to change directly to the desired state and have more control over the project.

Conclusions

The circuit we designed was really simple and easy to implement due to servo motors only requiring power and a PWM signal. Our toggle switch worked flawlessly because the servo attached to that switch allowed to us to set the position it was rotated to, and therefore we could just alternate between two angles with a little a bit of code. The continuous servo attached to the potentiometer did not work as well because we could only control the speed and direction of this motor. This caused the servo to sometimes rotate past the maximum and minimum angles the potentiometer could handle and damage the velcro connection. If possible, it would have been better to use a servo that is not continuous but is capable of rotating more than 180 degrees that way it could operate like the servo we used on the toggle switch.

I think the most important thing we learned was to not overplan and rather to start tinkering and prototyping as soon as possible. We spent a lot of time planning out the project but didn't leave much time to actually build it, and when problems arose that left even less time for troubleshooting which turned out to be the bulk of our work. In the future we will still set out a plan but we will not try to plan super far ahead because you can't predict what problems will arise and change the plan.

Appendix

```
#include <Servo.h>  // servo library

Servo servo1;  // servo control objects
Servo servo2;

int currentPercent = 0;
int buttonState = -1;
int stopVal = 88;
const int buttonPin = 2;     // the number of the pushbutton pin

/*  0 = 100%
 *  1 = 80%
 *  2 = 60%
 *  3 = 40%
 *  4 = 20%
 */

void setup()
{

  servo1.attach(9);   //light switch
  servo2.attach(10);  //Dimmer
  servo2.write(stopVal);
  Serial.begin(9600);
  pinMode(13, OUTPUT);
  pinMode(buttonPin, INPUT);
  changeSwitch(0);
}


void loop()
{

  int b = isButtonPressed();

  Serial.println(buttonState);

  if(b) {
    switch(buttonState){

      case 0: //100%
            setDimmer(100);
            break;
      case 1: //80%
            setDimmer(90);
            break;
      case 2: //60%
            setDimmer(80);
            break;
      case 3: //40%
            setDimmer(70);
            break;
      case 4: //20%
            setDimmer(60);
            break;
      default: //0%
            setDimmer(0);
            break;

    }
  }

  //delay(1000);



}

void setDimmer(int percent)
{
  /* Don't do anything if we're already at the proper position */
  Serial.print("percent: ");
  Serial.println(percent);
  if(percent == currentPercent)
    return;

  //Serial.println(percent);
  if((percent < 0) && (percent > 100))
    return;

  //turn switch on / off
  if((percent > 0) && (currentPercent == 0))
    changeSwitch(1);
  else if((percent == 0) && (currentPercent > 0))
    changeSwitch(0);


  //decrease percent - counter-clockwise
  if(percent < currentPercent)
  {
    //changeSwitch(0);
    int turn_delay = 1011 * ((double)(currentPercent-percent)/100);
    servo2.write(102);  // 88 makes it stop
    delay(turn_delay);
    servo2.write(stopVal);
    delay(200);  //prevent errors due to jerking at end of motion
  }

  /* Turning clockwise 100% (scaled by percent)*/
  else if(percent > currentPercent)
  {
    int turn_delay = 1010 * ((double)(percent-currentPercent)/100);
    servo2.write(74);  // 88 makes it stop
    delay(turn_delay);
    servo2.write(stopVal);
    delay(200);  //prevent errors due to jerking at end of motion
  }

  currentPercent = percent;
}

/*
void loopLightswitch(){
  int position;

  servo1.write(100);   //on
  delay(1000);

  servo1.write(50); //off

  delay(1000);

  for(position = 50; position < 100; position += 2)
  {
    servo1.write(position);  // Move to next position
    //Serial.print("loop #1: ");
    //Serial.println(position);
    delay(20);               // Short pause to allow it to move
  }

  delay(500);

  for(position = 100; position >= 50; position -= 2)
  {
    servo1.write(position);  // Move to next position
    //Serial.print("loop #2: ");
    //Serial.println(position);
    delay(20);               // Short pause to allow it to move
  }

}
*/

void changeSwitch(int status) {
  if(status)
  {
    servo1.write(99);   //on
    digitalWrite(13, HIGH);
  }
  else
  {
    servo1.write(53);   //off
    digitalWrite(13, LOW);
  }
}


int isButtonPressed()
{
  int buttonVal = digitalRead(buttonPin);
  if(buttonVal){
    while(digitalRead(buttonPin)){
        delay(1);
    }
    buttonState++;
    if(buttonState == 6)
      buttonState = 0;
    return 1;

  }

  return 0;
}
```