Sleep Switch

Zach Eaton, Matt Hansen

CSE 321, University at Buffalo

Executive Summary -

Our project is essentially a sensor that controls a cover for your light switch that will flip the light switch off if the room has been determined to be empty. We decided to undertake this project because a large amount of energy goes to waste due to active lights, TVs, etc. in empty rooms, which hurts the environment and your wallet. We thought this would be a good opportunity to deploy a realtime embedded system that can address this problem. In addition to being a helpful product for turning off unused lights, it can also act as a "clapper" that turns on/off the light when you clap. Our product would have the advantage of being cheap to install compared to many home automation products because it can be used over existing light switches without having to play with wiring. This product would be great for anyone that wants to implement some home automation but doesn't have great technical skills and doesn't know how to start. We designed this using an Arduino microcontroller and some components like a sound sensor and a piezo buzzer. We designed and implemented this using a breadboard and jumper wires as a proof of concept but we'd like to solder this and package it better in the future. The realtime and embedded portion of the project lies in the Arduino which is a microcontroller that runs in realtime on embedded scripts. Our script is deadline driven based on time, from the motor flipping the switch to waiting for user response to the buzzer.

Project Objectives -

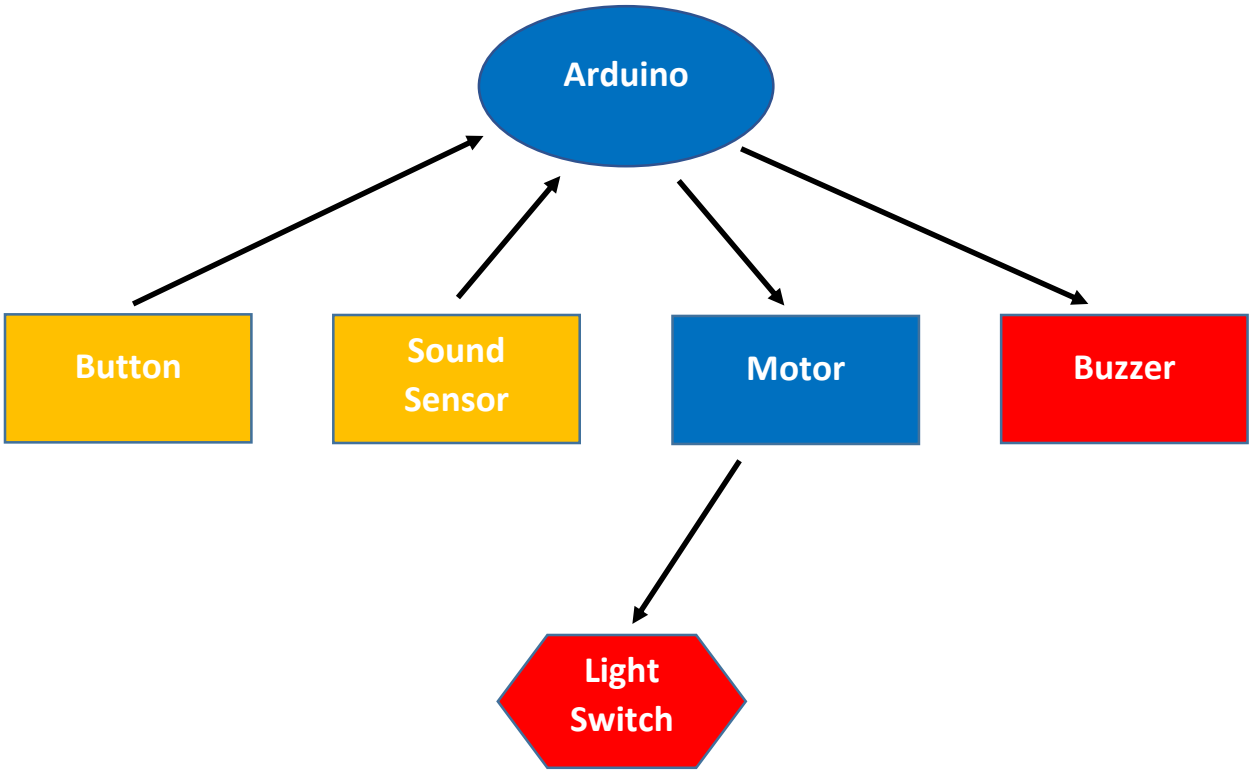
Sleep Switch will meet the following objectives:

- Work as a traditional light "clapper" that turns on and off lights with a clap.
- Listen for background noise and initiate a countdown to a buzzer if no sound is detected.
- Listen for a clap or a button press that cancels the light switching sequence if someone is in a room that was thought to be empty.
- Provide a cheap and easy entry into home automation.

Project Approach -

We first began development when we came up with the idea, and planned what we would need. We ordered these parts and then began software development in the Arduino IDE while we were waiting for shipment. The software was developed in C++ and used the Servo.h library to interface with the motor that flips the switch.

Project Description -



The diagram shows how each component interacts with the "brain" of the device - the Arduino. To start, the Arduino sends a signal to the Buzzer so that it sounds off every 30 minutes to give warning that the device will turn the light off in one minute. By pressing the Button or clapping for the Sound Sensor to "hear", this action is interrupted and the Arduino is informed that the lights are still in use. If the Arduino is not interrupted after one minute, it sends a signal to the Motor which then activates to flip the switch. The Sound Sensor is also constantly "listening" so that clapping at any moment will send a signal to the Arduino and then the Motor, flipping the switch on/off.

Pictured to the left is what the device would look like once installed (note that in this example both the switches have motors attached, and there is no buzzer).

User's Manual -

Once the device has been properly installed onto the light switch, utilizing the product is fairly simple. Every 30 minutes, the buzzer will sound off 3 times indicating the light is about to be turned off to save energy. Once the buzzer has finished, you have one minute to clap (make any loud noise) or press the button on the device to stop it from turning the light off. The device also works as a traditional light switch "clapper"- at any time you can clap to make the device turn the switch on or off.

Programmer's Manual –

All you need on the software end is to install the Arduino IDE on your personal computer, this will include the Servo.h library and allow you to compile and transfer the software to your Arduino.

Parts list

Electronics:

- *Mini Servo Motor*- A small motor that acts as a rotary actuator (link)
- *Arduino Kit*- A kit that includes an Arduino, mini breadboard, jumper wires, resistors (link)
- *Piezo*- the buzzer (link)
- *Sound Sensor*- the microphone (link)

Once built using materials listed and installed- see Assembly for details

Nuts and Bolts:

- 3mm x 10mm bolt (x3)
- 3mm nut (x2)
- 3mm locknut (x1)
- 3mm washer (x3)
- 5mm washer (x4)
- 6-32 x 1.25" machine screws (x2*)*

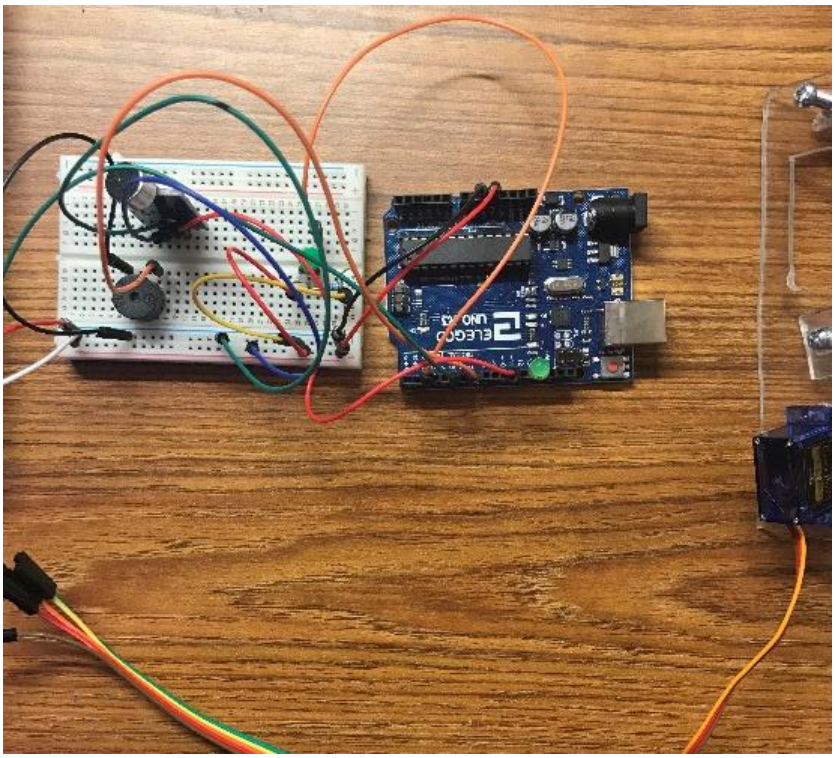
Lexan (Polycarbonate- 10cm x 15cm):

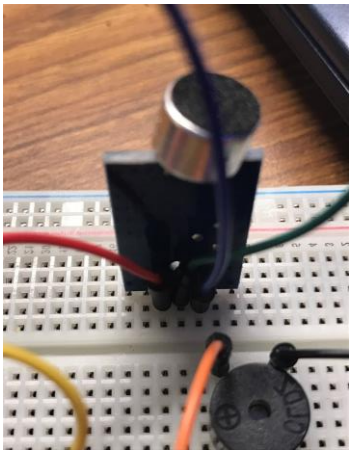
- *Cut your own*- You will need a drill and saw (jigsaw/scroll saw or laser cutter) *See Design pdf*

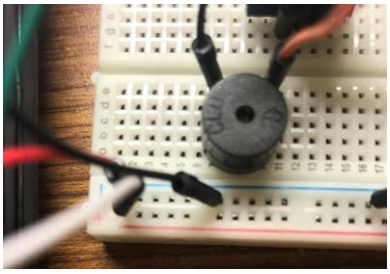
Assembly

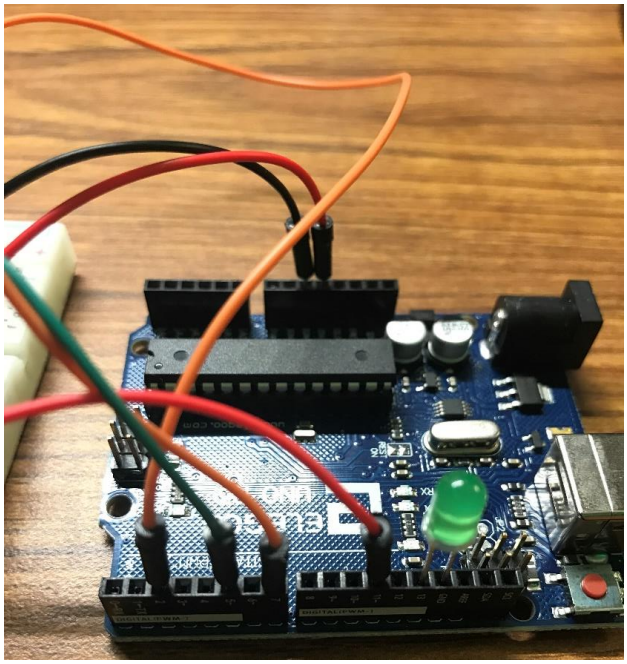
Use the instructions found in the *Assembly pdf* to build the light switch attachment

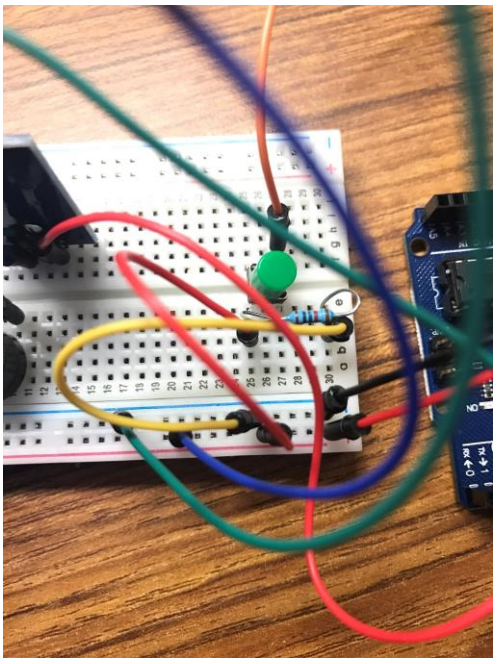
Setup the Breadboard and Arduino with the electronic components according to the following pictures:

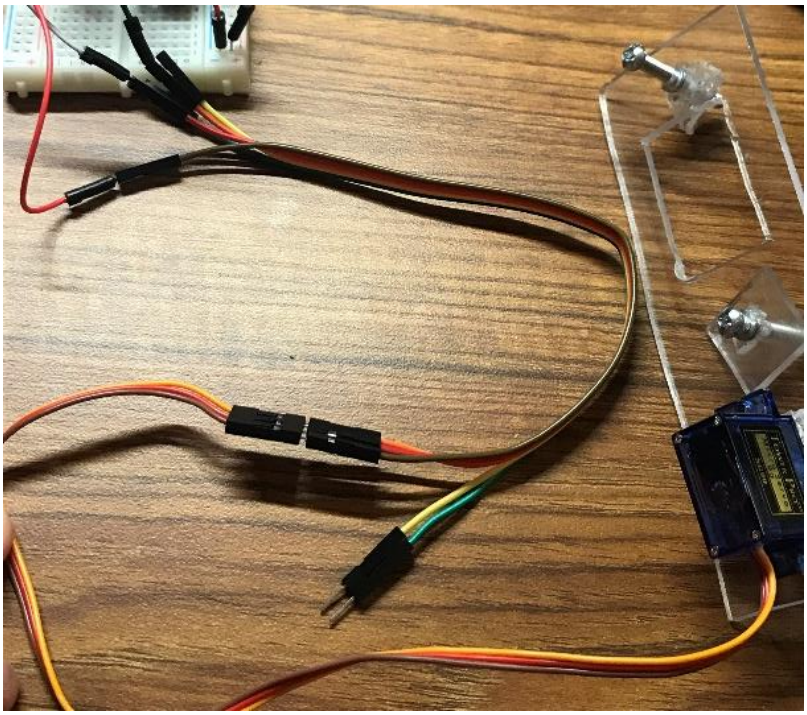


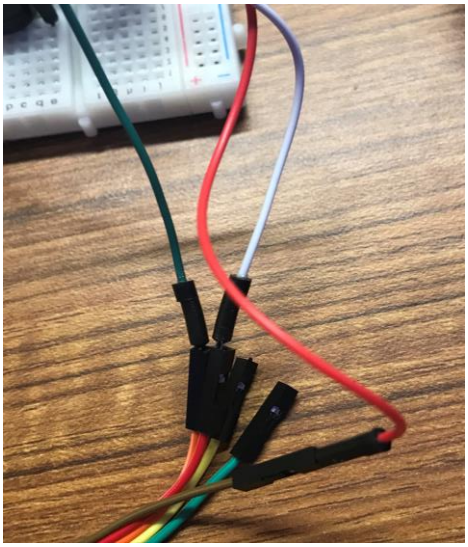


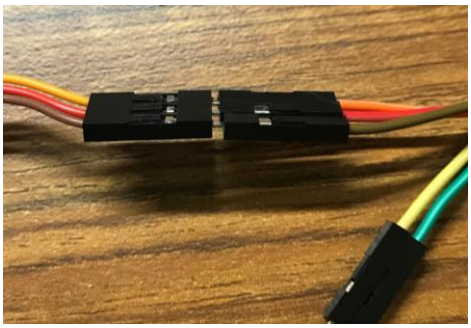












Programming

Use the following code to program the Arduino.

```
#include <Servo.h>
#define motorPin 5
#define buttonPin 2
#define buzzerPin 7
#define sensorPin 11
#define ledPin 13
#define OFF 180
#define NEUTRAL 110
#define ON 20

Servo myservo;  // create servo object to control a servo
long start = 0, timer = 0;
```

```
boolean val = 0, pushed = 0, switchOn;


void setup(){
 //Serial.begin(9600); Serial monitoring for debugging
 myservo.attach(motorPin);  // attaches the servo on pin 9 to the servo object
 pinMode(buttonPin, INPUT); // sets the sensors to Arduino pins
 pinMode(buzzerPin, OUTPUT);
 pinMode(sensorPin, INPUT);
 myservo.write(ON); //turn the switch on so we know what state it's in
 delay(1000);
 switchOn = 1;
 myservo.write(NEUTRAL);
 delay(1000);
 myservo.detach(); // turn the servo off when not in use to save power
}


void loop (){
 //delay(1800000); //check every 30 minutes
 timer = millis();
 while(millis() - timer > 1800000){//wait for 30 mins but continually listen
   if(digitalRead(sensorPin)){//if there's noise
     if(switchOn){//if the switch was last turned on
       myservo.attach(motorPin); //turn on the motor again
       myservo.write(OFF); //turn the switch off
       delay(1000); //wait for the motor to move
       switchOn = 0; //keep track of the last move the motor made
       myservo.write(NEUTRAL);//go to the neutral position so the switch can be manually flipped
       delay(1000);
       myservo.detach(); // turn off he motor again
       return; //reset
     }
     else{
```

```
      myservo.attach(motorPin);
      myservo.write(ON);
      delay(1000);
      switchOn = 1;
      myservo.write(NEUTRAL);
      delay(1000);
      myservo.detach();
      return;
    }
  }
}

for(int x = 0; x < 3; x++){  //sound buzzer alarm 3 times
  //Serial.println("Buzz");  //debugging
  tone(buzzerPin, 2000, 500); //turn on the buzzer
  delay(1000); // wait a second
  noTone(buzzerPin); // turn off the buzzer
  delay(1000);
}

start = millis();//start the response timer
pushed = 0;
while(millis() - start < 30000){ //repeat this loop for 30 seconds
  int volume = digitalRead(sensorPin); //check if there's sound
  if(volume == HIGH){ // if there's sound over the threshold (clap)
    //digitalWrite(ledPin, HIGH); //Turn ON LED for debugging
    pushed = 1; //acknowledge a person's presence in the room
    //digitalWrite(ledPin, LOW); //Turn OFF LED
    break; //break out of the response loop after response
  }
  else if(digitalRead(buttonPin) == HIGH){ //if a person pressed the buzzer
    pushed = 1;
```

```
    break;
  }
 }

 if(pushed){ //if there was a user detected
  //Serial.println("returning"); // for debugging
  return; // reset
 }
 else{
  //Serial.println("moving"); // for debugging
  myservo.attach(motorPin);
  myservo.write(OFF); //turn off the light if no one is in the room
  delay(1000);
  myservo.write(NEUTRAL);
  delay(1000);
  myservo.detach();
  return; //reset
 }
}
```

References

http://www.instructables.com/id/Easy-Home-Automation-using-servo-switches/

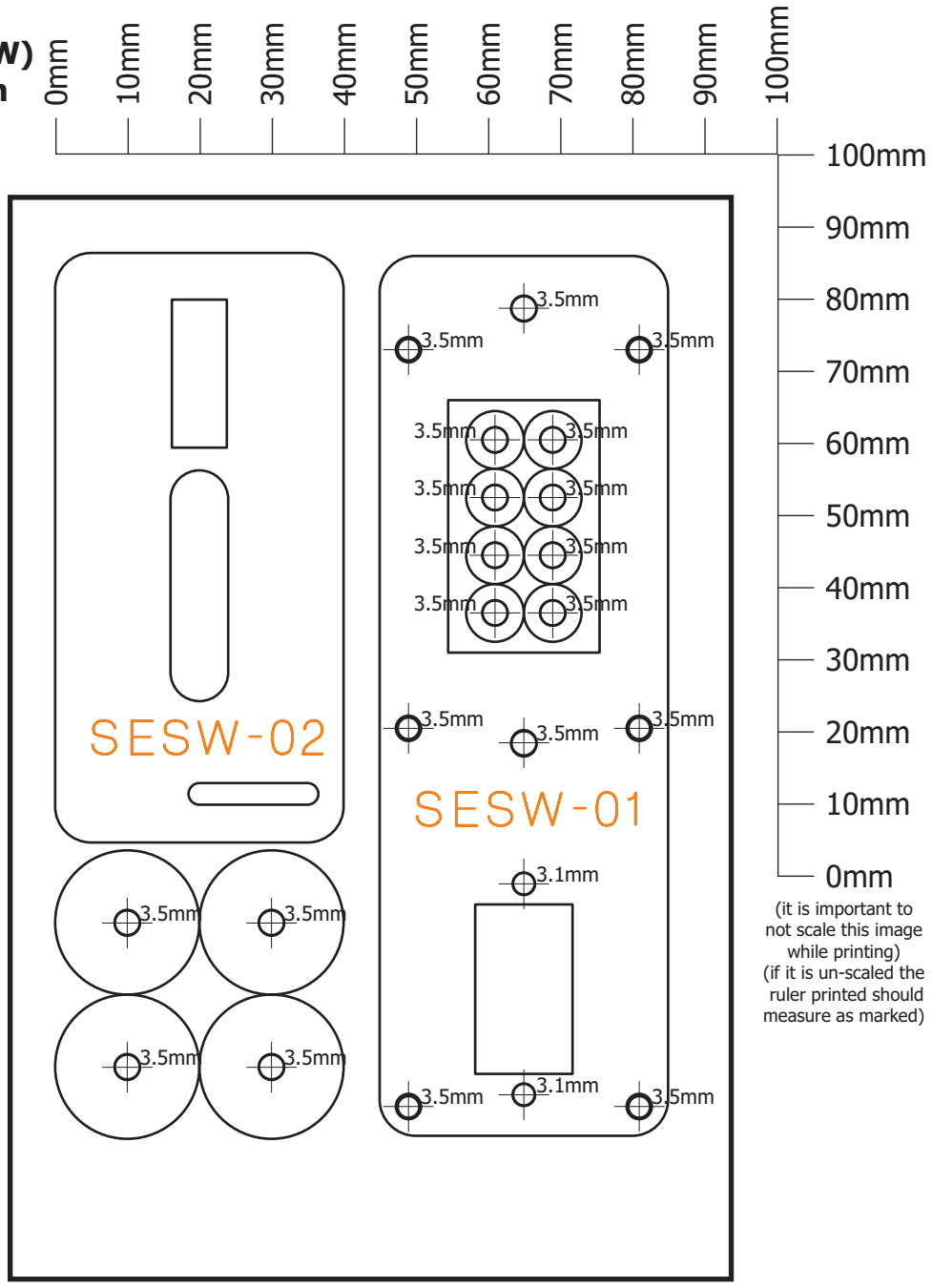
https://www.arduino.cc/en/Reference/Servo

https://learn.sparkfun.com/tutorials/sik-experiment-guide-for-arduino---v32/experiment-11-using-a-piezo-buzzer

http://www.instructables.com/id/Arduino-Sound-Sensor-with-LED/

https://www.arduino.cc/en/tutorial/pushbutton

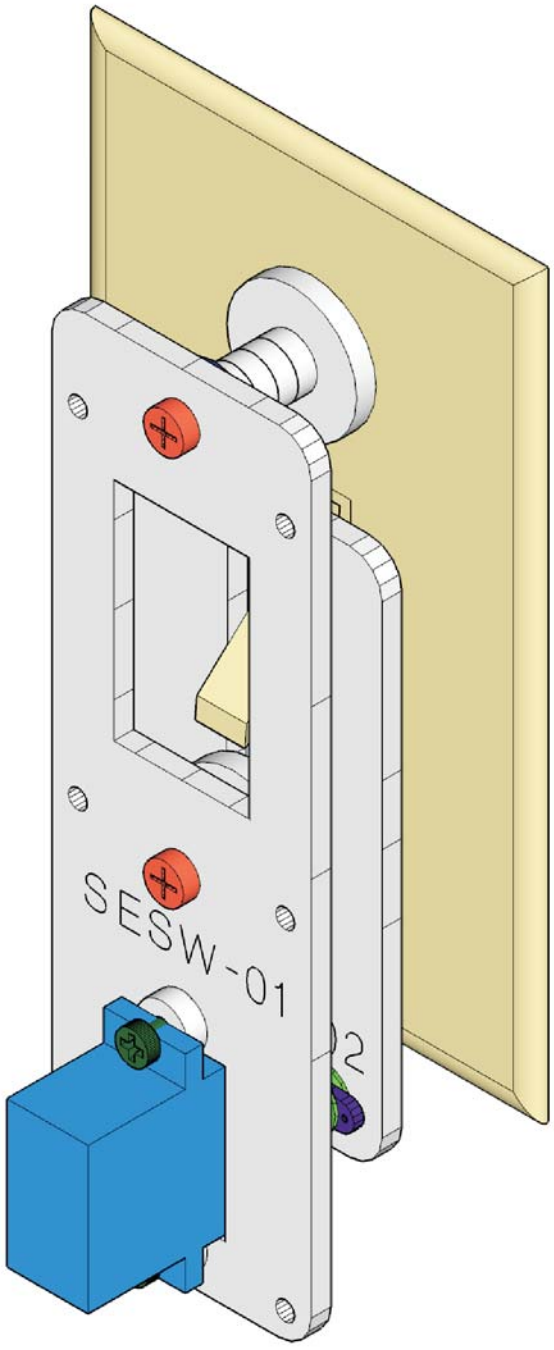
Servo Switch (SESW)
ScrollSaw Pattern
Page (1 of 1)



(it is important to not scale this image while printing)
(if it is un-scaled the ruler printed should measure as marked)

Acrylic Sheet Size
100mm x 150 mm

Servo Switch Assembly Guide



(SESW)

www.oomlout.com

1.

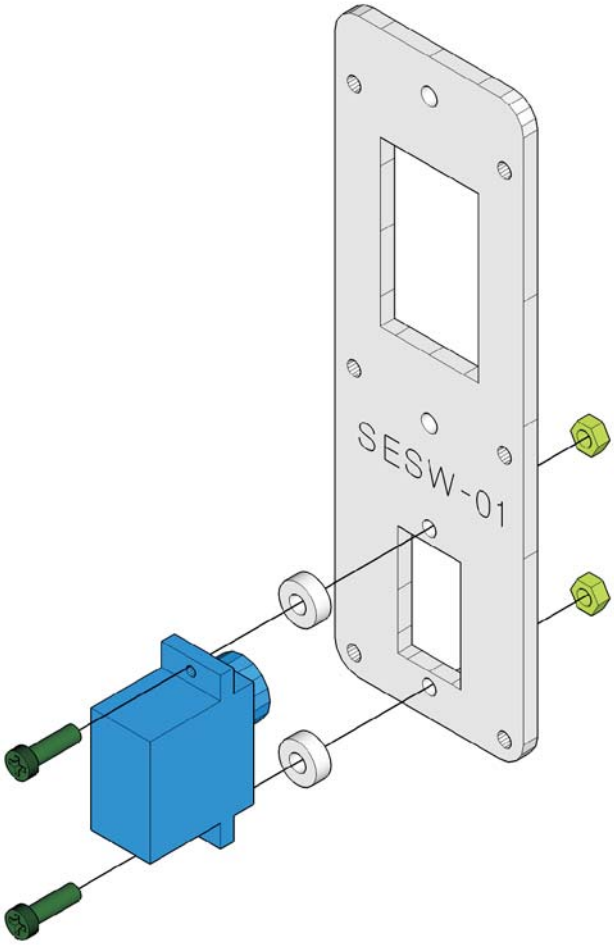
3mm x 10mm bolt
x2

3mm nut
x2

mini servo
x1

SESW-01
servo holder
x1

SESW-03
8mm spacer
x2



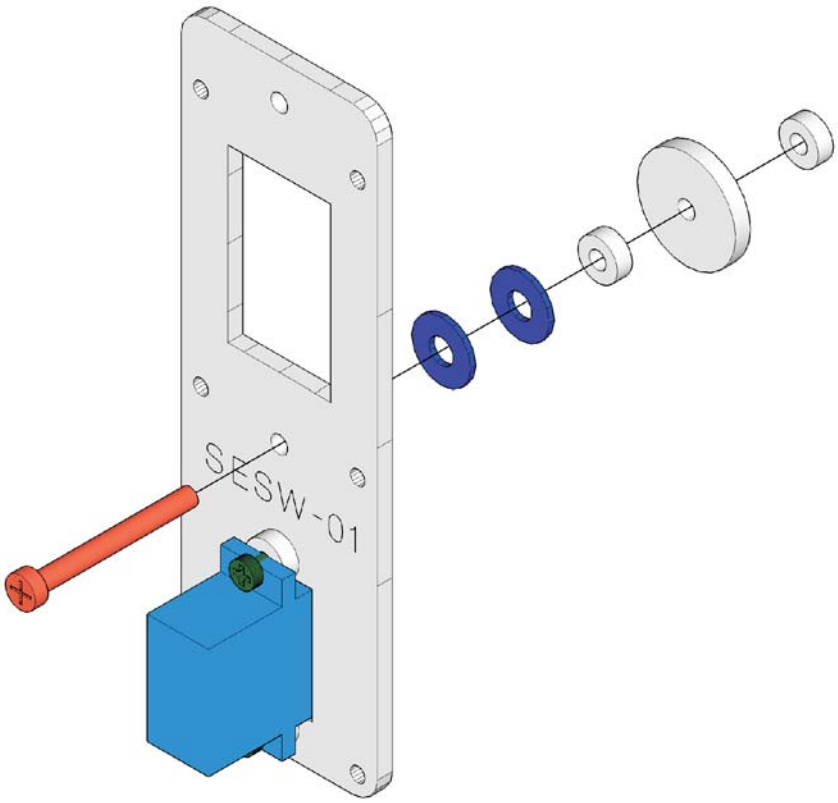
2.

5mm washer
x2

6-32 x 1.25”
bolt
x1

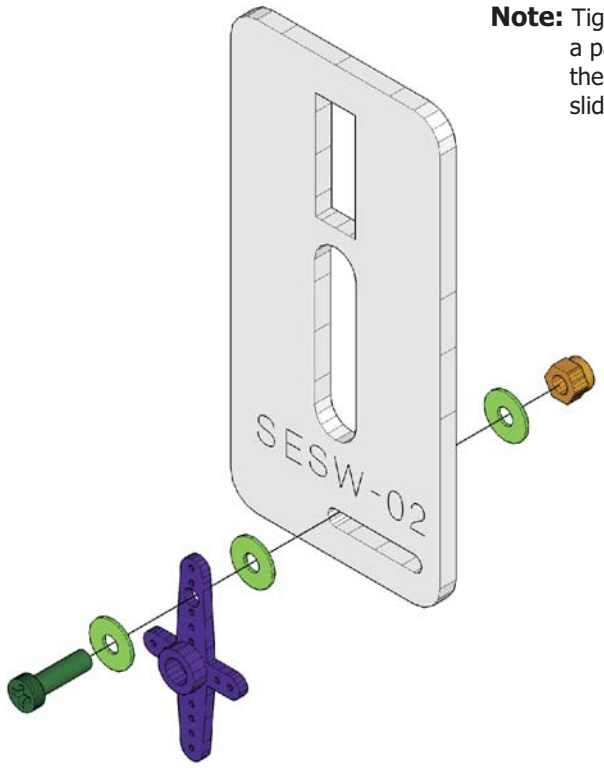
SESW-03
8mm spacer
x2

SESW-04
20mm spacer
x1



3.

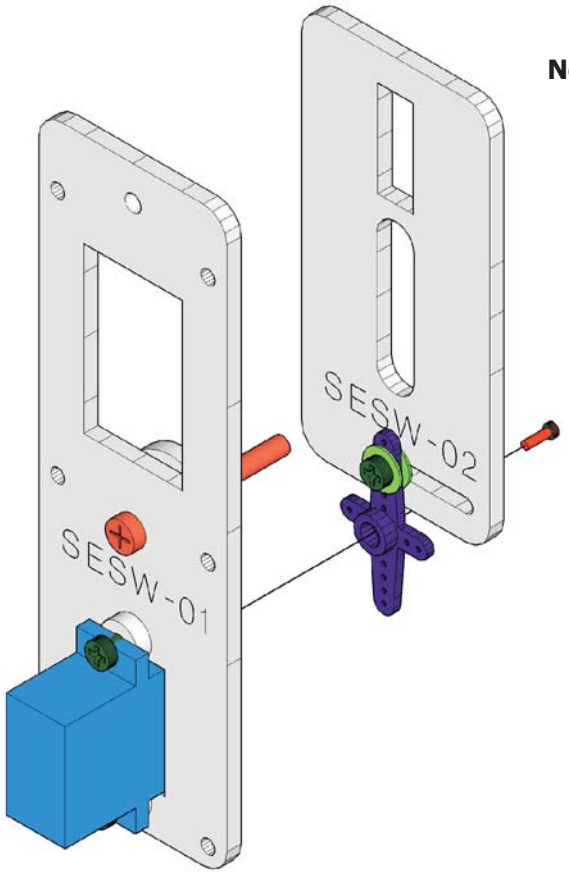
- 3mm x 10mm bolt **x1**
- 3mm lock nut **x1**
- 3mm washer **x3**
- servo horn **x1**
- SESW-02 switch carriage **x1**



Note: Tighten holding the locknut in a pair of pliers. Tighten until the bolt is snug but can still slide freely along the track.

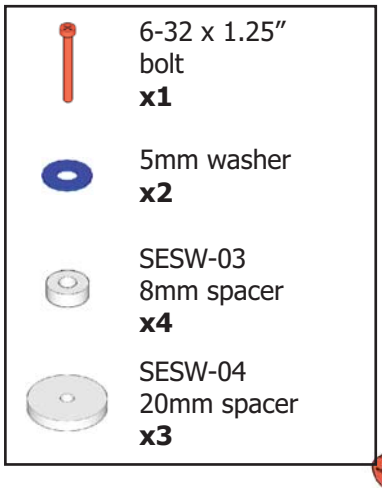
4.

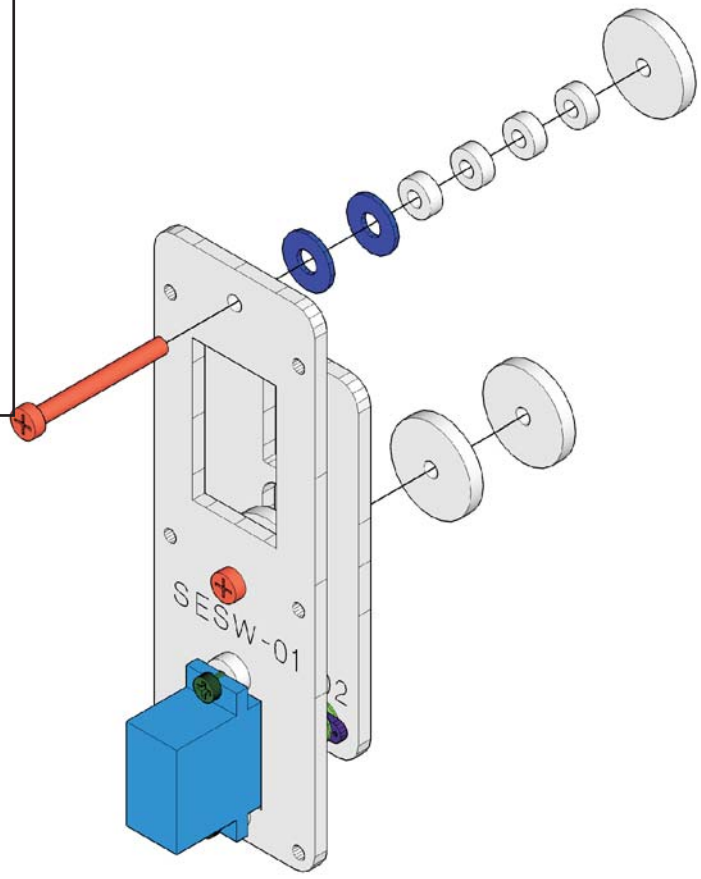
- servo screw **x1**



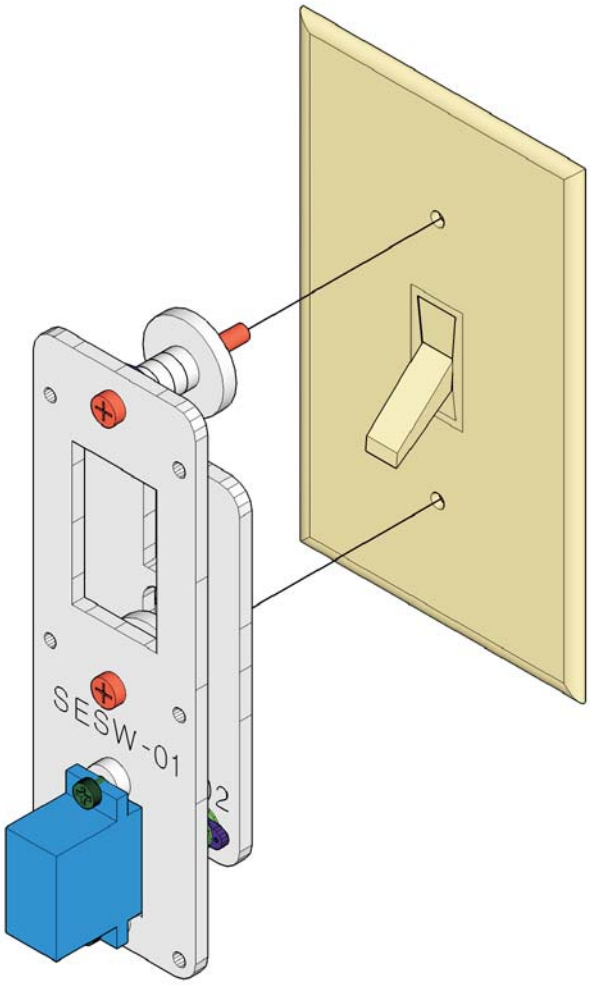
Note: To orient the servo use the servo horn to turn the servo as far clockwise as it will allow you to go, then attach.

5.





6.



Note: Tighten the bottom bolt so the servo switch is held tight but loose enough so the carriage slides freely.

Wiring

Pin 1 - black or brown - **ground**
Pin 2 - red - **power** (4.8v-6v)
Pin 3 - white or orange - **signal**
(1-2 millisecond pulse every 25 ms)
(or use the arduino Servo library)

www.oomlout.com