# Arduino with Relay Shield

By: Matthew Jourden
Brighton High School
Brighton, MI

Relays: Arduinos only allow for 5 volts to be passed to a circuit, which does not allow a user to run very many peripherals; a relay connects a secondary power source to a controller (i.e. Arduino) to connect the peripheral devices. The relay receives a High/Low signal from a micro-controller to open the circuit to a the secondary power source. NOTE: it is important to know how much voltage the relay can handle. It is typically written on the relay box.


Arduino

Peripherals
(i.e. Servo, DC Motors, Lights, Sensors, etc.)

Relay has two groups of pins Low Voltage Group (Controller) and High Voltage Group (Peripherals)

Low Voltage Group

4. GND: Ground Wire on Arduino
5. VCC: Connect to 5V Pin on Arduino
6. IN: Receives control signal from Arduino

High Voltage Group

4. COM: is the Common Pin used in both normally open or closed mode (Typically Ground Wire from Peripheral Device)
5. NO: Normally Open (Starts in the Closed (OFF) State (Low) moves to the Open (ON) State)
6. NC: Normally Closed (Starts in the Open (ON) State (Low) moves to the Closed (OFF) State)

## Arduino Relay Shield (4 Relay Channel Device)

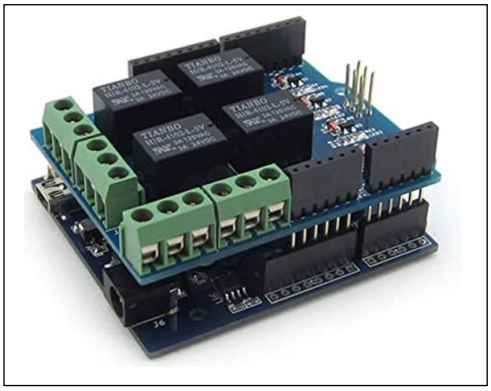
Relay Voltage Max 24 Volts per relay

Shield links the Relays to defined digital ports

NOTE: Digital Ports may not be used for signals if the Relay is being used

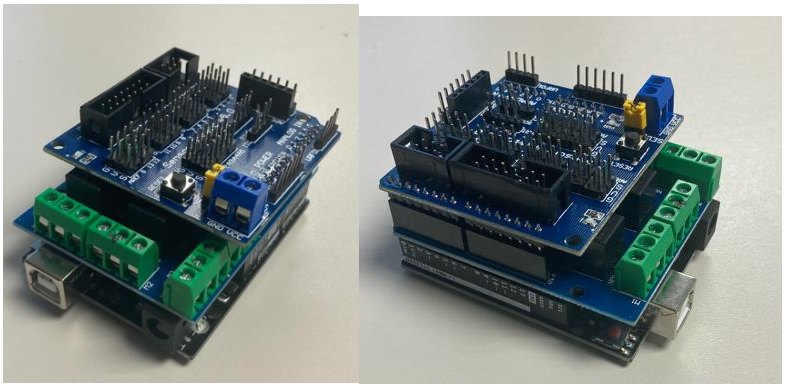
Digital Ports

Relay 1 is controlled by digital pin 7
Relay 2 is controlled by digital pin 6
Relay 3 is controlled by digital pin 5
Relay 4 is controlled by digital pin 4

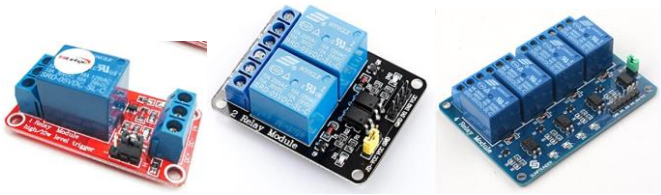



Terminal Ports are setup the same as the single relay shown above

NOTE: a Sensor Shield maybe placed on Top of the Relay Shield to provide Ground, Voltage and Signal Ports for all Pins



## Or Use a Single, Double, 4 or 8 Channel Relay



**Program:** Arduino with Relay Shield and Multiple Servo Motors to calibrate a robotic arm

Robotic Arms have 6 Joints in which to move and rotate in the X, Y and Z Planes/Axis

Major Joints: 1, 2, and 3 Move Parallel with X, Y and Z Planes

Minor Joints 4, 5, and 6 Rotate Around the X, Y, and Z Axis

Joint 1: X Plane

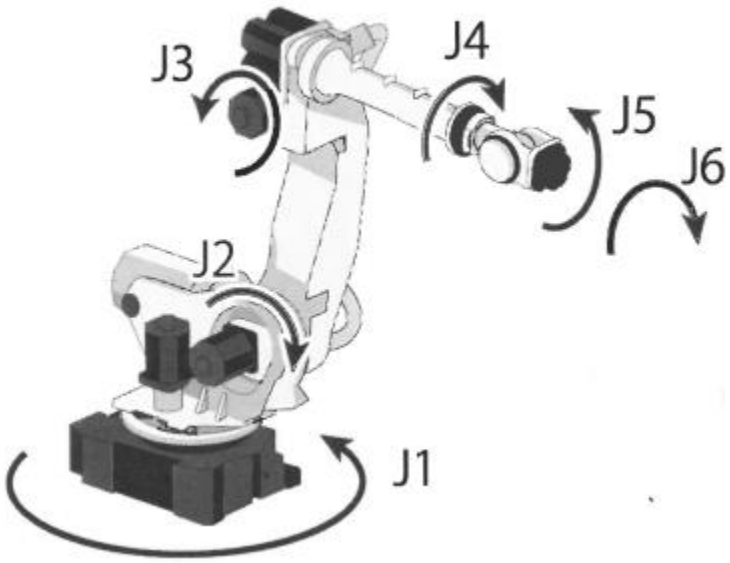
Joint 2: Y Plane

Joint 3: Z Plane

Joint 4: X Axis

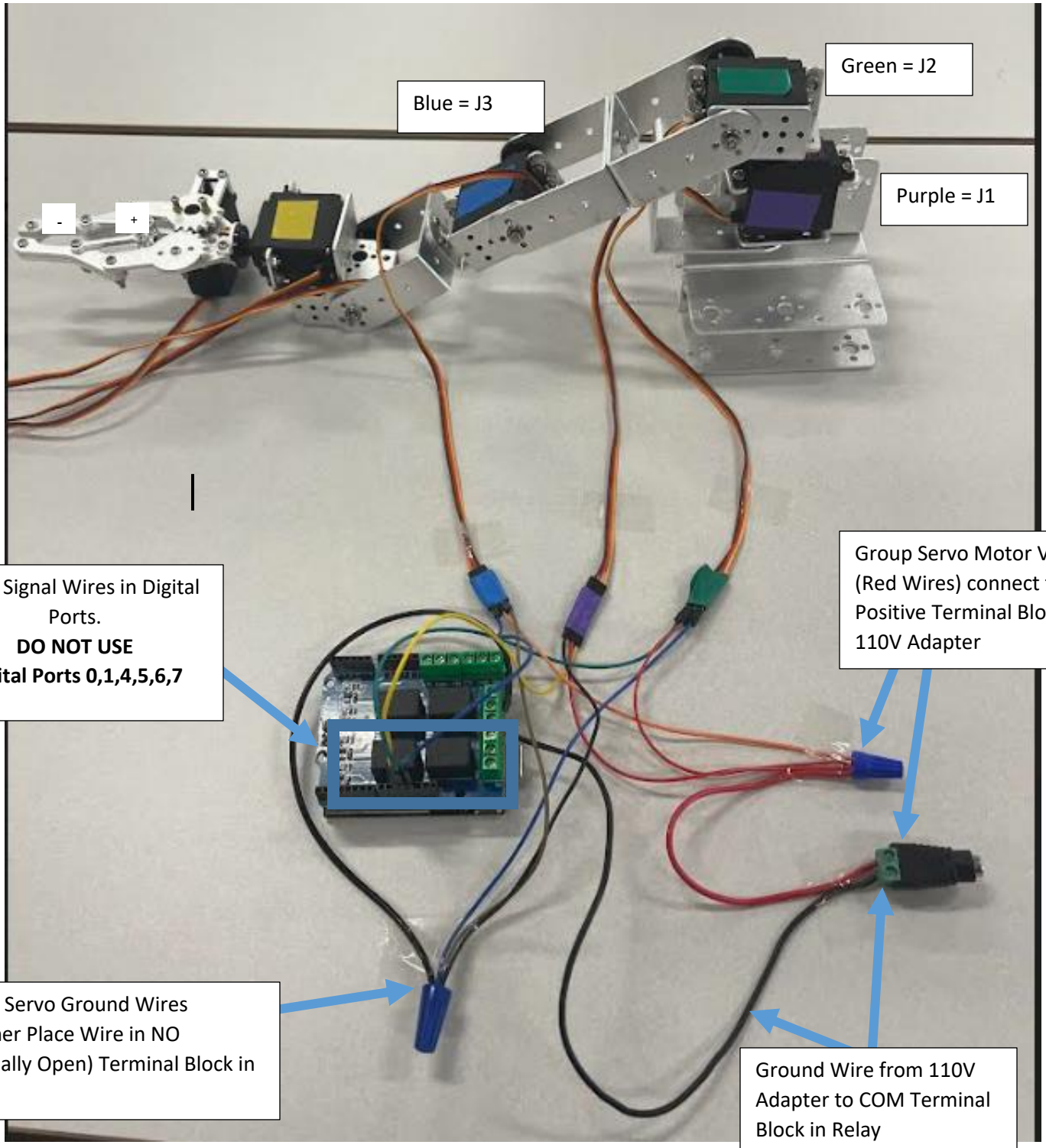
Joint 5: Y Axis

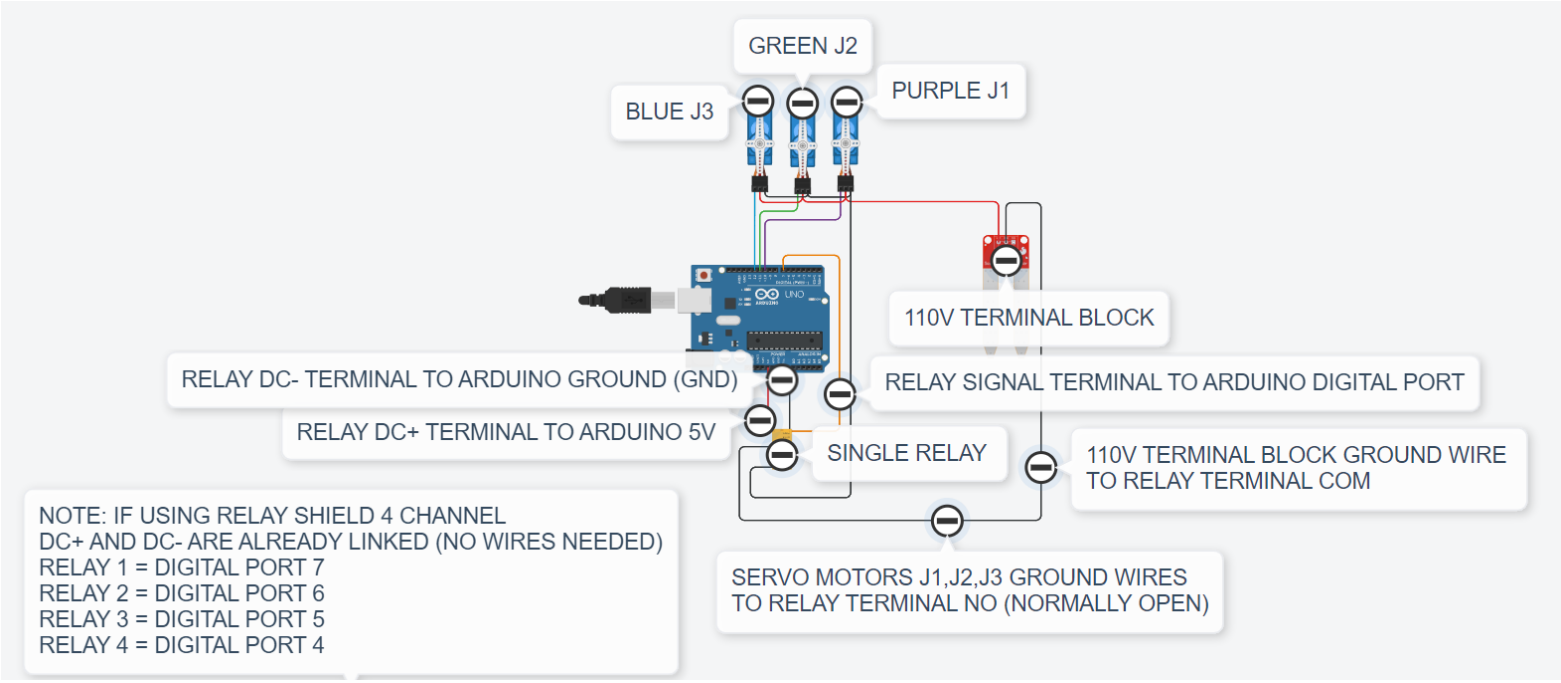
Joint 6: Z Axis or End Effector like a Claw

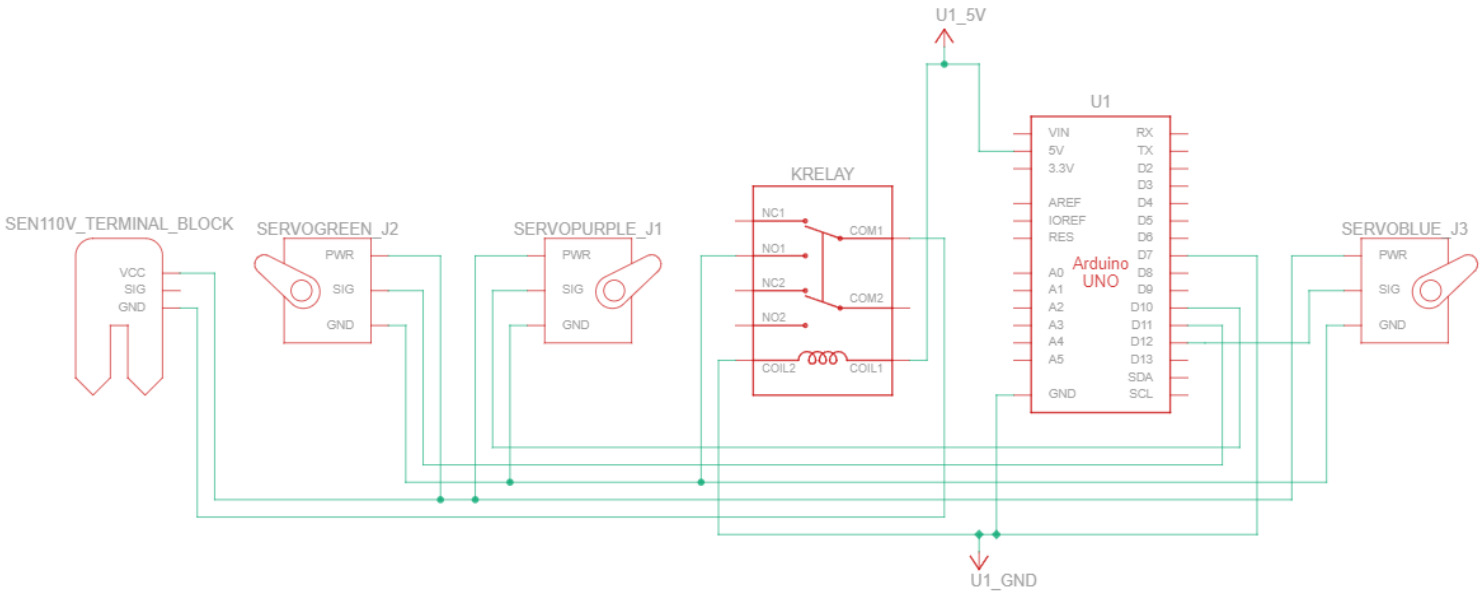


**Objective:** Wiring 3 servo motors to 1 relay to supply 110 volts Wall Outlet of electricity. This will allow the three motors to have enough voltage and current to operate and not lose Digital Port.

**Wire the following Circuit**




GREEN J2
PURPLE J1
BLUE J3
110V TERMINAL BLOCK
RELAY DC- TERMINAL TO ARDUINO GROUND (GND)
RELAY SIGNAL TERMINAL TO ARDUINO DIGITAL PORT
RELAY DC+ TERMINAL TO ARDUINO 5V
SINGLE RELAY
110V TERMINAL BLOCK GROUND WIRE TO RELAY TERMINAL COM
NOTE: IF USING RELAY SHIELD 4 CHANNEL
DC+ AND DC- ARE ALREADY LINKED (NO WIRES NEEDED)
RELAY 1 = DIGITAL PORT 7
RELAY 2 = DIGITAL PORT 6
RELAY 3 = DIGITAL PORT 5
RELAY 4 = DIGITAL PORT 4
SERVO MOTORS J1,J2,J3 GROUND WIRES
TO RELAY TERMINAL NO (NORMALLY OPEN)


U1_5V
U1
SEN110V_TERMINAL_BLOCK
VCC
SIG
GND
SERVOGREEN_J2
PWR
SIG
GND
SERVOPURPLE_J1
PWR
SIG
GND
KRELAY
NC1
NO1
NC2
NO2
COM1
COM2
COIL2
COIL1
VIN
5V
3.3V
AREF
IOREF
RES
A0
A1
A2
A3
A4
A5
GND
Arduino UNO
RX
TX
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
SDA
SCL
SERVOBLUE_J3
PWR
SIG
GND
U1_GND

# Calibration Program

**Write the following Program**

## **Part 1:** Find the Limits for Joint 1

Directions: Right the follow Program.

NOTE: Servo Names and Servo Pin Names maybe different based on the color of the motors on a chosen Robotic Arm

1. Turn Relay On
2. Attach PURPLEJ1
3. Move or Rotate Joint 1 between 0 -180 Degrees. NOTE Motors may not be able to reach the 0 or 180 due to interference from the robotic arm frame. Adjust degrees that are written to the robotic arm to find the limits of the Joint.
4. Detach Joint 1 Servo
5. Turn Relay Off

```
#include <Servo.h>
Servo PurpleJ1, GreenJ2, BlueJ3;                        //NOTE: Colors tape placed on the motors and wires to identify between servos; Users maybe different

int PurpleJ1pin = 10, GreenJ2pin = 11, BlueJ3pin =12;   //NOTE: Colors tape placed on the motors and wires to identify between servos; Users maybe different
int JointRelay1 = 7;                                    //Relay Shield Relay1 = Digital Port 7; See Tutorial for Relay2,3,4 Locations
void setup() {

pinMode (PurpleJ1pin, OUTPUT);
pinMode (GreenJ2pin, OUTPUT);
pinMode (BlueJ3pin, OUTPUT);
pinMode (JointRelay1, OUTPUT);

digitalWrite (JointRelay1, HIGH);

delay (5000);
PurpleJ1.attach (PurpleJ1pin);
PurpleJ1.write (0);
delay (2000);
PurpleJ1.write (180);
delay (2000);
PurpleJ1.Detach();

digitalWrite (JointRelay1, LOW);                    //Turns Relay OFF.  CAUTION When Relay turned OFF all power is removed from the motor
                                                    //the motor will move to a position based on gravitational forces
}

}
void loop() {

}
```

# CAUTION: CLEAR AWAY SPACE AROUND THE ROBOT. ROBOT WILL MOVE AND KNOCK OBJECTS AWAY.

# CAUTION: BE CAREFULL OF PINCH POINTS FOR FINGERS AND HAIR

**NOTE: Leaving power attached to a servo motor that is affected by gravity will help keep the servo motor in a set position, once power is removed the weight of the servo motor will make it fall down.**

**Operator needs to be careful in leaving a constant relay open since energy is constantly being drawn and could overload a circuit or drain the power sources.**

6. **Upload and Test Program NOTE Be careful when uploading the robot will begin moving > Calibrate and Adjust degrees for Limits as needed**

**Recommend** **to add comments on the program to note where the true Limits of the motors are and other data points**

**Example**

```
15  delay (5000);                                   //delay provides time for the operator to setup and adjust wires
16  PurpleJ1.attach (PurpleJ1pin);
17  PurpleJ1.write (45);                             //Demo Robot 45 degree sets limit to Right
18  delay (2000);
19  PurpleJ1.write (180);                           //Demo Robot 180 degree sets limit to Left
20  delay (2000);
21  PurpleJ1.Detach();
```

**Part 2:** Find the Limits for Joint 2

Directions: Comment Out Code for Joint 1 > Add to the Program for Joint 2

NOTE: Servo Names and Servo Pin Names and locations maybe different based on the color of the motors on a chosen Robotic Arm

```
15 delay (5000);                        //delay provides time for the operator
16                                      // to setup and adjust wires
17 /*PurpleJ1.attach (PurpleJ1pin);
18 PurpleJ1.write (0);
19 delay (2000);
20 PurpleJ1.write (180);
21 delay (2000);
22 PurpleJ1.Detach();*/
23
24 GreenJ2.attach (GreenJ2pin);
25 GreenJ2.write (0);
26 delay (2000);
27 GreenJ2.write (180);
28 delay (2000);
29
30 digitalWrite (JointRelay1, LOW);    //Turns Relay OFF
31 }
32 void loop() {
33   // put your main code here, to run repeatedly:
34 }
```

# CAUTION: CLEAR AWAY SPACE AROUND THE ROBOT. ROBOT WILL MOVE AND KNOCK OBJECTS AWAY.

# CAUTION: BE CAREFULL OF PINCH POINTS FOR FINGERS AND HAIR

*****Upload and Test Program NOTE Be careful when uploading the robot will begin moving > Calibrate and Adjust degrees for Limits as needed****

**NOTE: Leaving power attached to a servo motor that is affected by gravity will help keep the servo motor in a set position, once power is removed the weight of the servo motor will make it fall down.**

**Operator needs to be careful in leaving a constant relay open since energy is constantly being drawn and could overload a circuit or drain the power sources.**

**Part 3:** Find the Limits for Joint 3

Directions: Comment Out Code for Joint 1 and 2 > Add to the Program for Joint 3

NOTE: Servo Names and Servo Pin Names and locations maybe different based on the color of the motors on a chosen Robotic Arm

```
14
15  delay (5000);                       //delay provides time for the operator
16                                      // to setup and adjust wires
17  /*PurpleJ1.attach (PurpleJ1pin);
18  PurpleJ1.write (0);
19  delay (2000);
20  PurpleJ1.write (180);
21  delay (2000);
22  PurpleJ1.Detach();
23
24  GreenJ2.attach (GreenJ2pin);
25  GreenJ2.write (0);
26  delay (2000);
27  GreenJ2.write (180);
28  delay (2000);*/
29
30  BlueJ3.attach (BlueJ3pin);
31  BlueJ3.write (0);
32  delay (2000);
33  BlueJ3.write 180);
34  delay (2000);
35
36  digitalWrite (JointRelay1, LOW);    //Turns Relay OFF
37  }
38  void loop() {
39    // put your main code here, to run repeatedly:
40  }
```

# CAUTION: CLEAR AWAY SPACE AROUND THE ROBOT. ROBOT WILL MOVE AND KNOCK OBJECTS AWAY.

# CAUTION: BE CAREFULL OF PINCH POINTS FOR FINGERS AND HAIR

**NOTE: Leaving power attached to a servo motor that is affected by gravity will help keep the servo motor in a set position, once power is removed the weight of the servo motor will make it fall down.**

**Operator needs to be careful in leaving a constant relay open since energy is constantly being drawn and could overload a circuit or drain the power sources.**

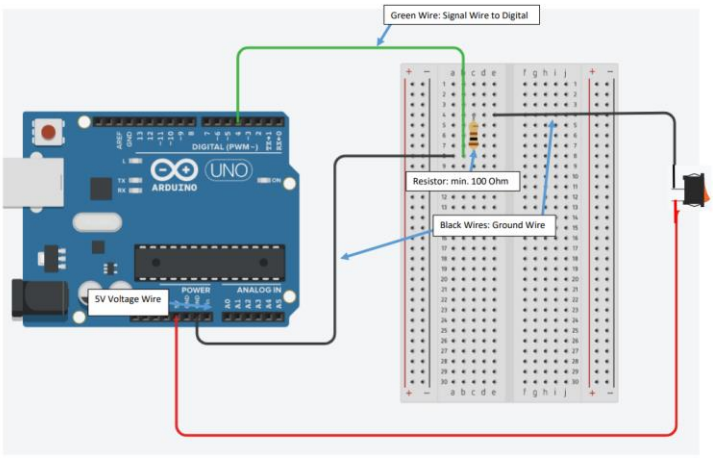
*****Upload and Test Program NOTE Be careful when uploading the robot will begin moving > Calibrate and Adjust degrees for Limits as needed****

## Part 4: Wire Toggle Switch

Modify the code from above to do the following

1. Remove the 5sec Delay on Line 15 of the tutorial image above
2. Wire a Toggle Switch
3. Toggle Switch will do the following
    a. Toggled On: Turn Activate the Code that Turns the Relay On
    b. Toggled Off: Activates New Code that will turn the Relay Off

**Toggle Switch Wiring Diagram**




Sample Code for Toggle Switch or Push Button

```
Program: Turns LED Light On/OFF
int ledPin = 13; // choose the pin for the LED
int inPin = 7;   // choose the digital input pin (for a pushbutton)
int val = 0;    // variable for reading the pin status 0 = LOW, 1 =HIGH

void setup() {
 pinMode(ledPin, OUTPUT);  // declare LED as output
 pinMode(inPin, INPUT);    // declare pushbutton as input
}
void loop(){
 val = digitalRead(inPin);  // read input value
 if (val == HIGH) {       // check if the input is HIGH (button released)
  digitalWrite(ledPin, LOW);  // turn LED OFF
 } else {
  digitalWrite(ledPin, HIGH);  // turn LED ON
 }
}
```

4. Move the Motor Detachments to the end of the program as part of turning the relay off.

**<u>CAUTION: Once Power is REMOVED from the Motors they will no longer be held in place. Be careful to not let the Robotic Arm CRASH into the Table or other OBJECTS</u>**

5. Upload and Test the Program

## Part 5: Wire and Test Joints 4,5,6

1. **Wiring:**
    a. Wire Remaining Servo Motors to a Relay Servo Shield or Standalone Relay
        i. Use a 2nd 110V DC Terminal Block Adapter. Operator should have 2 wall plugs for the robotic arm.
        ii. Group Servo Motors Joints 4,5 and 6 to a 2nd Relay Position
        NOTE: If using a Shield Digital Ports are as follows
            1. Digital Ports
                a. Relay 1 is controlled by digital pin 7
                Relay 2 is controlled by digital pin 6
                Relay 3 is controlled by digital pin 5
                Relay 4 is controlled by digital pin 4

## Repeat the Steps of above to wire > Calibrate Joints 4,5,6.

**<u>CAUTION: DO NOT Try to Calibrate all 3 Joints at the same time. Calibrate 1 Joint at a time</u>**

# Assignment: Block Pick and Place

# NOTE: DO NOT Overwrite the Calibration Program

1. Clamp Robot to Table
2. Tape off Location of Robot, Pick Location and Place Location
3. **Create a New Program** for the Robotic Arm to pick up a block and put it down.
   Toggle Switch: Used to Turn On and Off Relays so the robot can be setup and not moved the instant the program is updated. This gives the operator control over when the robot arm is begins operation
2. Write a program that will autonomously pick up a block from a defined location Pick Location > Move to defined location Place Location
3. Return to a Neutral Home Position
4. Pick Up Block from Place Location > Place in the Original Pick Location
5. Add Piezo Speaker to buzz when
    a. Claw grasps the Block from Pickup Location
    b. Claw releases the Block at Drop-Off Location
6. Add RGB LED
    a. Red = Not Moving
    b. Green = Moving Points
    c. Blue = Grasp or Release Block
7. Optional: Add Potentiometers and/or Push Buttons to manually control the robotic arm.