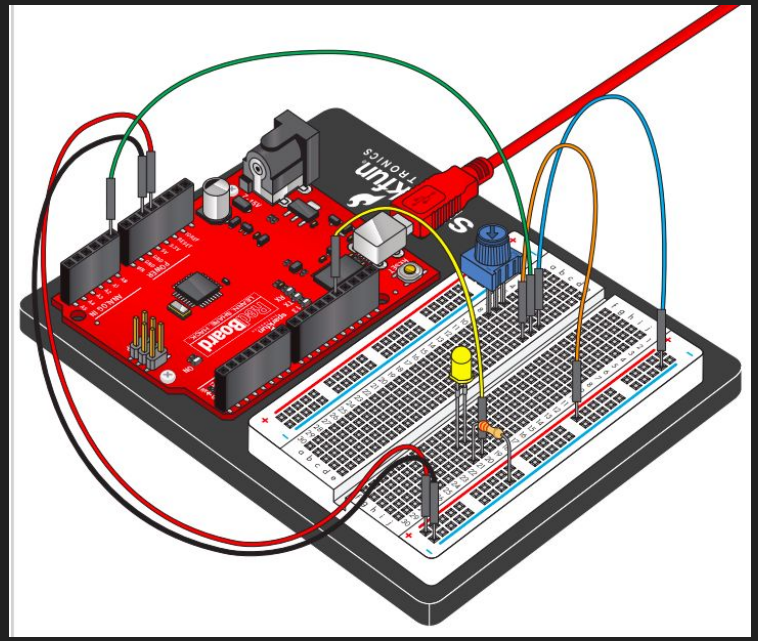
Arduino Concepts

By : Sharat

Credits to sparkfun for certain codes (for certain labs)

Lab 02 Potentiometer



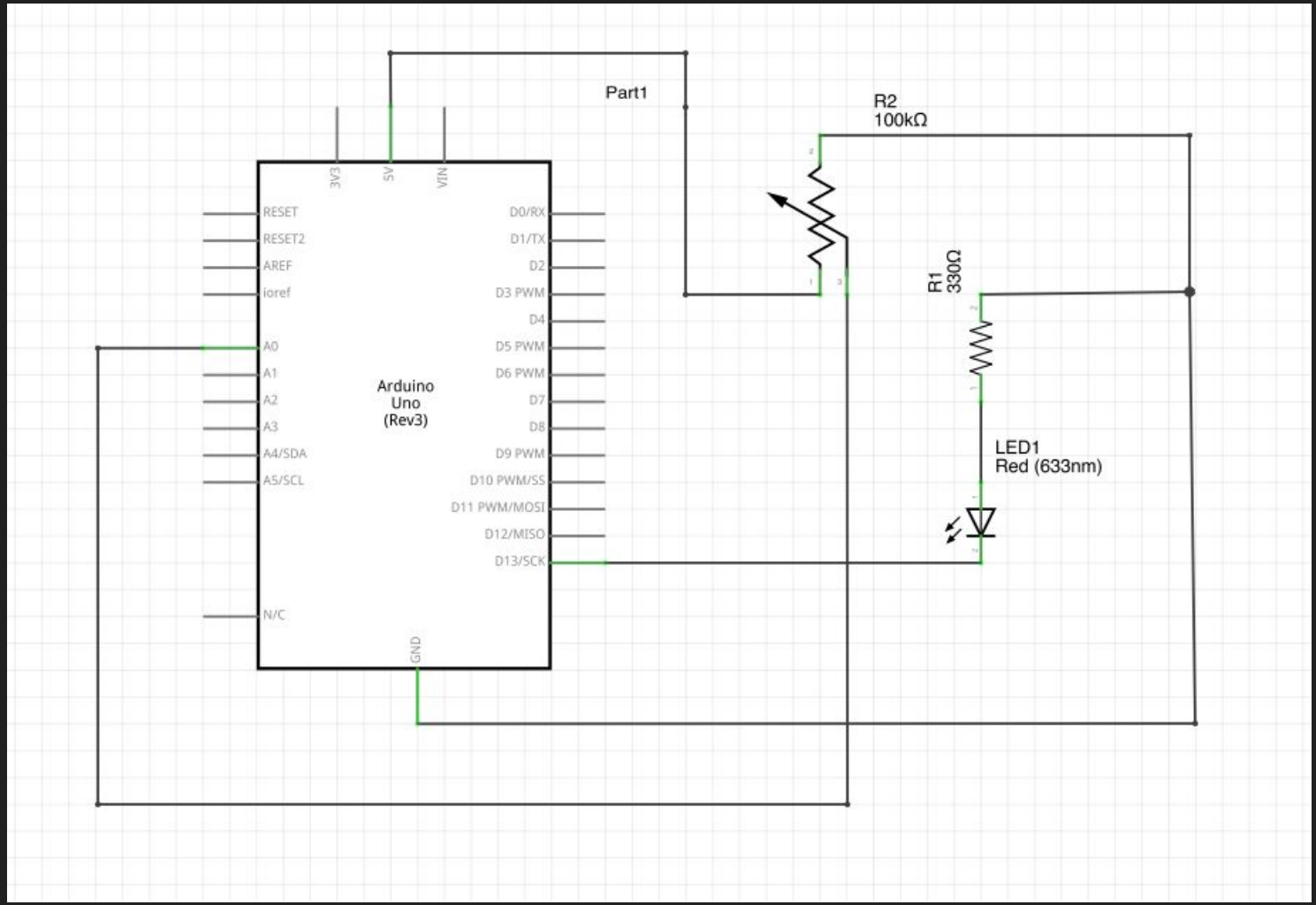
- Use of Potentiometer to change the speed of the blinking of the LED.

You should see the LED blink faster or slower in accordance with your potentiometer. If it isn't working, make sure you have assembled the circuit correctly and varied and uploaded the code to your board or see the troubleshooting tips below.

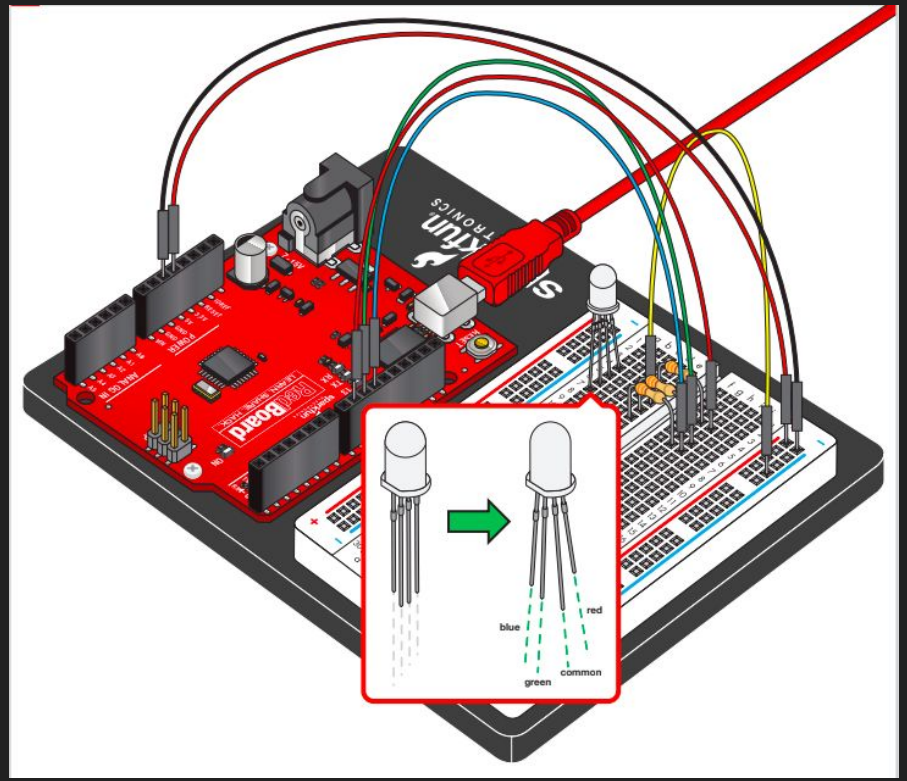
List of Components : Potentiometer (x1), LED (x1), 330 ohms resistor (x1), wires (x6)

Observations

What I observed is that as we increase the resistance of the potentiometer the LED started blinking slower because there was more resistance whereas when we decrease the resistance, the LED starting blinking faster and faster since there is less resistance. This let the LED blink faster and faster eventually to the point where it looks like it isn't blinking due to it blinking so fast as this is when the potentiometer is at its lowest resistance. The trend we really observe is how the resistance of the potentiometer really affects the pace or speed of a blinking LED as resistance played a key role. This essentially occurs due to the potentiometer controlling the voltage of the led as it is a variable resistor (resistance can be changed). The analog pin was also used instead of the digital pin in order to change the voltage of the LED and so that we can get a number between on and off (Analog Value) instead of it being only on and off (digital pin).


Part1
R2
100kΩ
R1
330Ω
LED1
Red (633nm)
Arduino
Uno
(Rev3)
3V3
5V
VIN
RESET
RESET2
AREF
ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
GND
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK

Lab 03 RGB Led

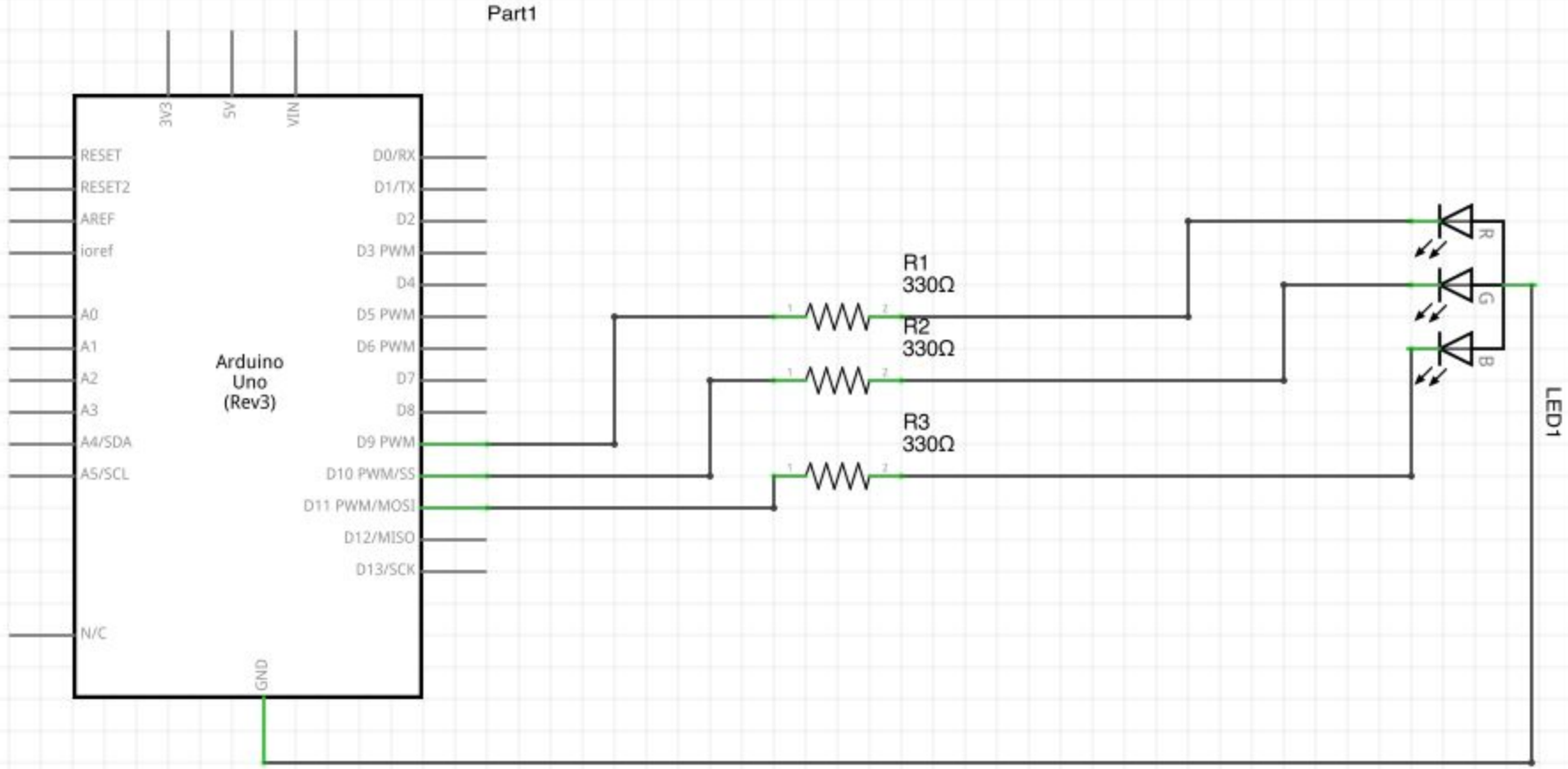


- An RGB led switching between colours through code (connected to digital pins)

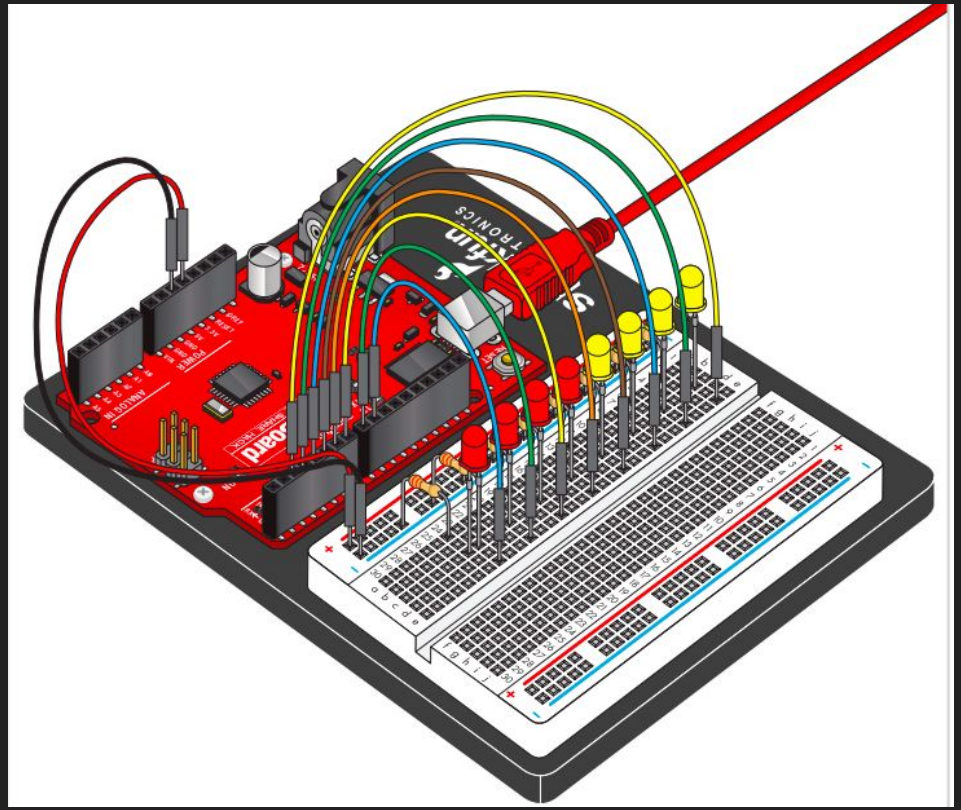
List of Components : RGB LED (x1), 330 ohms resistor (x3), wires (x6)

Observations

In this Lab the RGB led was coded to display various colours as the red pin, green pin and blue pin were all connected to the digital pins 9,10,11 of the arduino and we were able to code the rgb led through these digital pins to create different colours. Through coding the rgb values “(0,0,255) or (255,0,0) or (0,255,0) we can produce the basic three colours blue, red or green. Through using these colours and mixing them such as “(255,0,255) we can produce a colour like magenta as the rgb led is essentially mixing the green colour and red colour to produce this colour as the pins of the rgb led are coded to do so. Through this we were able to produce various colours as the digital pin values were essentially being coded as they were given values to get that amount of that colour such as red and a certain amount of green or blue to produce different colours. Like for magenta we used (255,0,255) as we used the red digital pin and blue digital pin values with no green values to get this colour.


Part1
3V3
5V
VIN
RESET
RESET2
AREF
ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
GND
Arduino
Uno
(Rev3)
R1
330Ω
R2
330Ω
R3
330Ω
R
G
B
LED1

Lab 04 Multiple LEDs

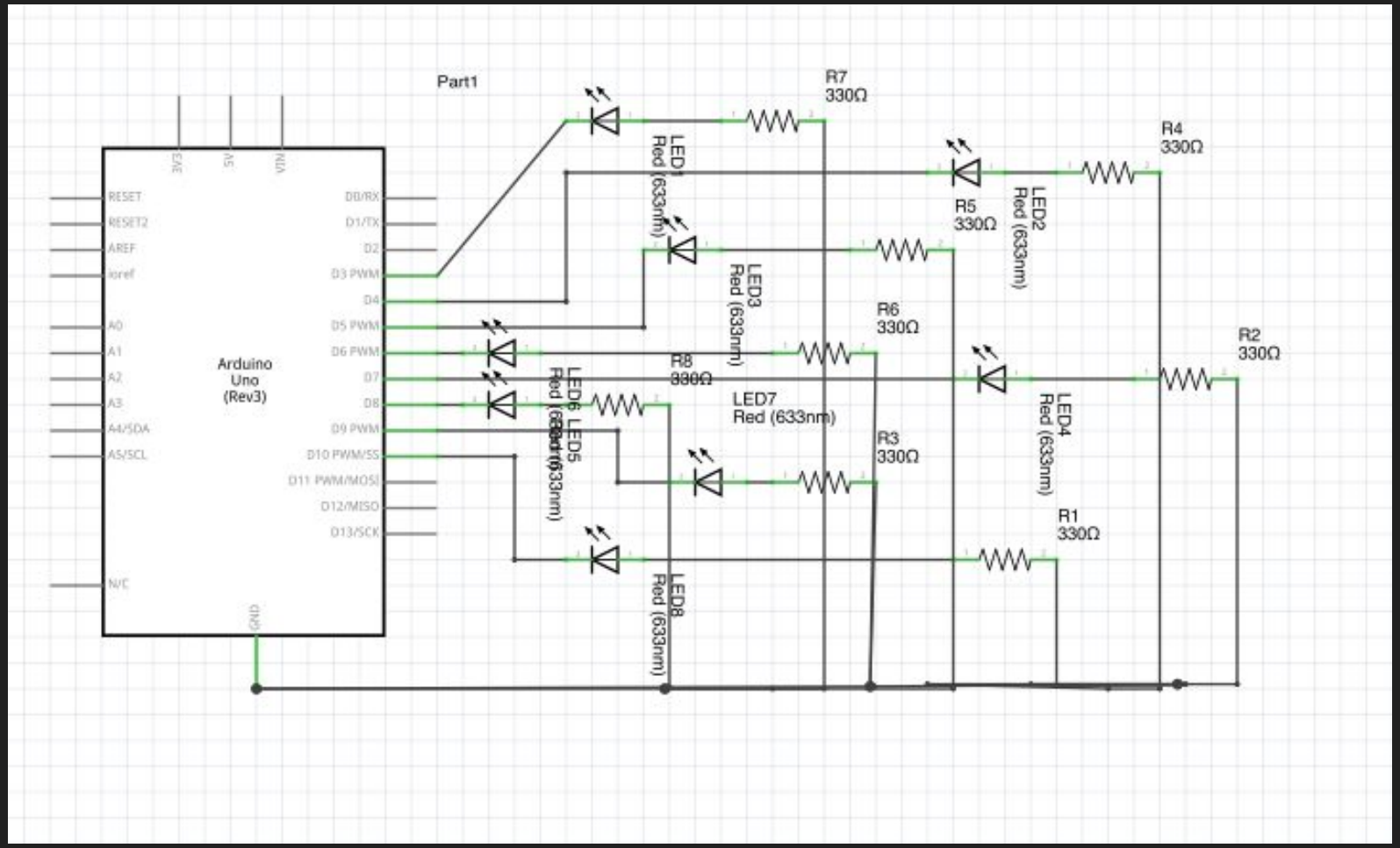


- Multiple Leds blinking one by one in parallel

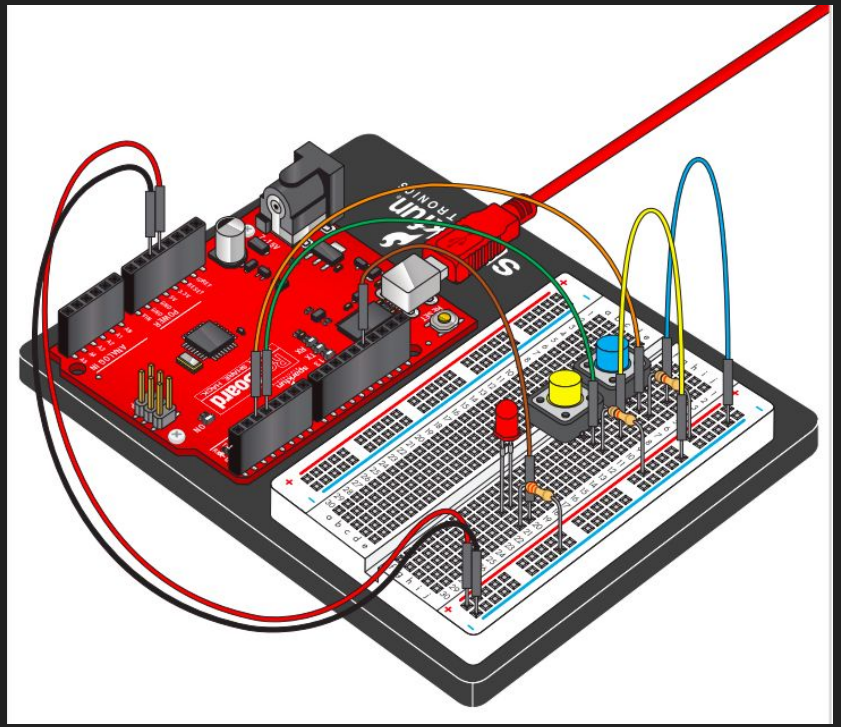
List of Components : LEDs (x8), 330 ohms resistor (x8), wires (x10)

Observations

We essentially saw how the LED started blinking one by one as they were in parallel from the 1st LED all the way to the 8th LED as they blinked one by one. There was a sort of delay going on as we see how there was a delay between the Led’s as it went one by one otherwise all of them would have went at the same time which isn’t what we want. The delay was essential as each LED essentially has a numerical standard and as it went up from 1 to the 8th there would be a delay so that way it looks like there’s a count up of the LED lights. We also observed how the LED were connected to digital pins as this is what essentially made this led light countup possible without a shift register as we were able to code the digital pins to make the count up go from 1st LED to 8th.


Part1
Arduino
Uno
(Rev3)
RESET
RESET2
AREF
Ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
GND
LED1
Red (633nm)
LED2
Red (633nm)
LED3
Red (633nm)
LED4
Red (633nm)
LED5
LED6
LED7
Red (633nm)
LED8
Red (633nm)
R1
330Ω
R2
330Ω
R3
330Ω
R4
330Ω
R5
330Ω
R6
330Ω
R7
330Ω
R8
330Ω

Lab 05 Push Buttons

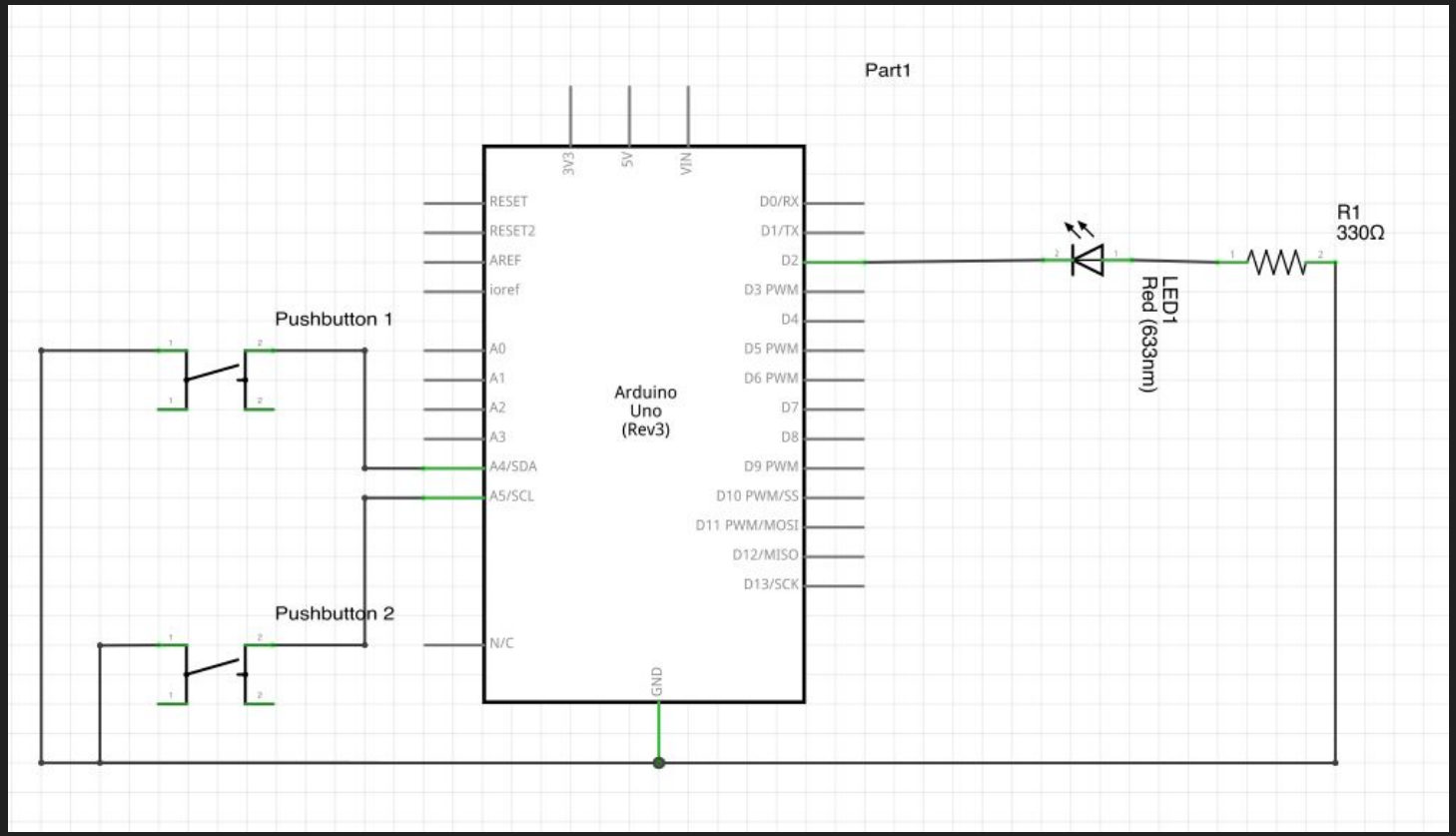


Essentially turning and LED off and on through the use of two push buttons.

List of Components : Push buttons (x2), LED (x1), 10k ohm resistor (x2), 330 ohm resistor (x1), wires (x9)

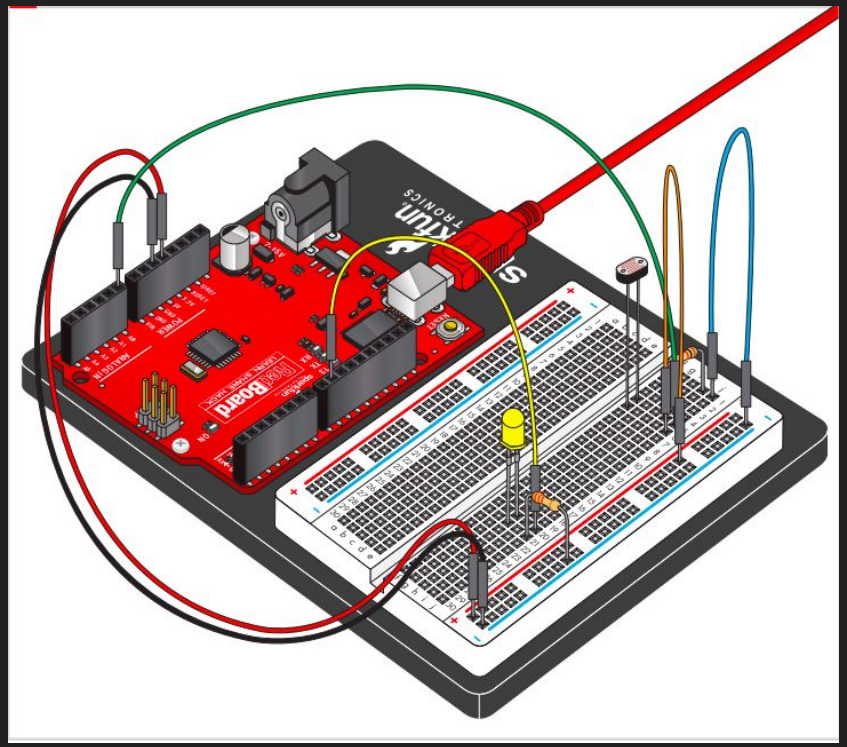
Observations

I observed how the push buttons essentially acted as switches as when we pressed both push buttons the LED turned off but when we pressed one of them the LED brightness was decreased as the LED became dim. When we didn’t press any push buttons the LED remained on but when we pressed them the LED turned off essentially acting like a switch as the push buttons were coded using digital pins along with the LED making the circuit possible through coding the push buttons to turn off the LED when pressed.



NOTE* READJUSTED SCHEMATIC SINCE WE changed it after video was RECORDED due to realization of a different method. There was only a vid of first one so I put this schematic.

Lab 06 Photoresistor

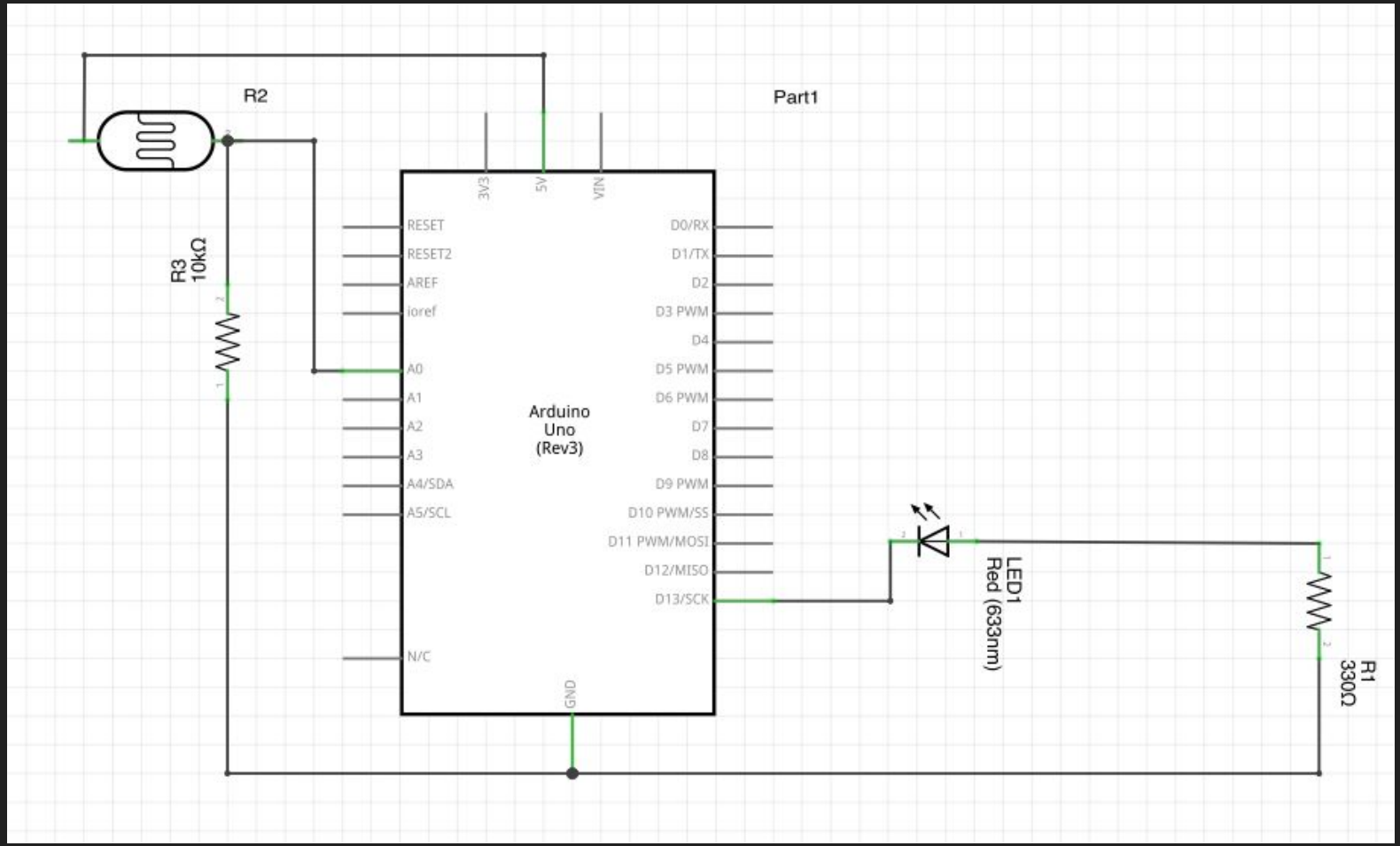


The LED turns on and off according to the analog value of the photoresistor.

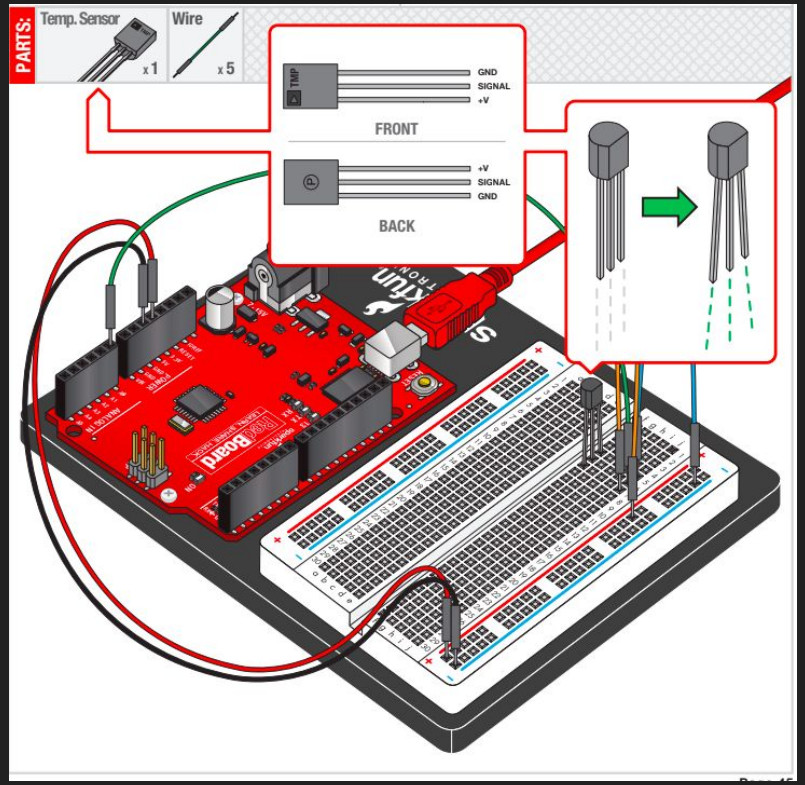
List of Components: Photoresistor (x1), LED (x1), 330 ohms resistor (x1), 10k ohms resistor (x1), wires (x6)

Observations

What we observed in this lab is that when we kept our hand at the LDR and blocked the light, the LED remained off but when we kept our hand away from the LDR the LED became bright as the LDR is essentially acting as a resistor as it's called a Photoresistor. Essentially we coded it so that when it doesn't sense light there's resistance for which reason it's connected to the analog pin as that Analog value we get can then be used to determine whether if there's light or not so that when there is light, the high analog value we get will turn the LED on as the led is connected to a digital pin. When we get a low analog value due to the LDR being blocked from light, the LED doesn't turn on as it connected to a digital pin. The Analog value is important as it essentially makes this condition possible through an on and off scenario like this one.


Part1
R2
R3
10kΩ
3V3
5V
VIN
RESET
RESET2
AREF
ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
GND
Arduino
Uno
(Rev3)
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
LED1
Red (633nm)
R1
330Ω

Lab 07 Temperature Sensor

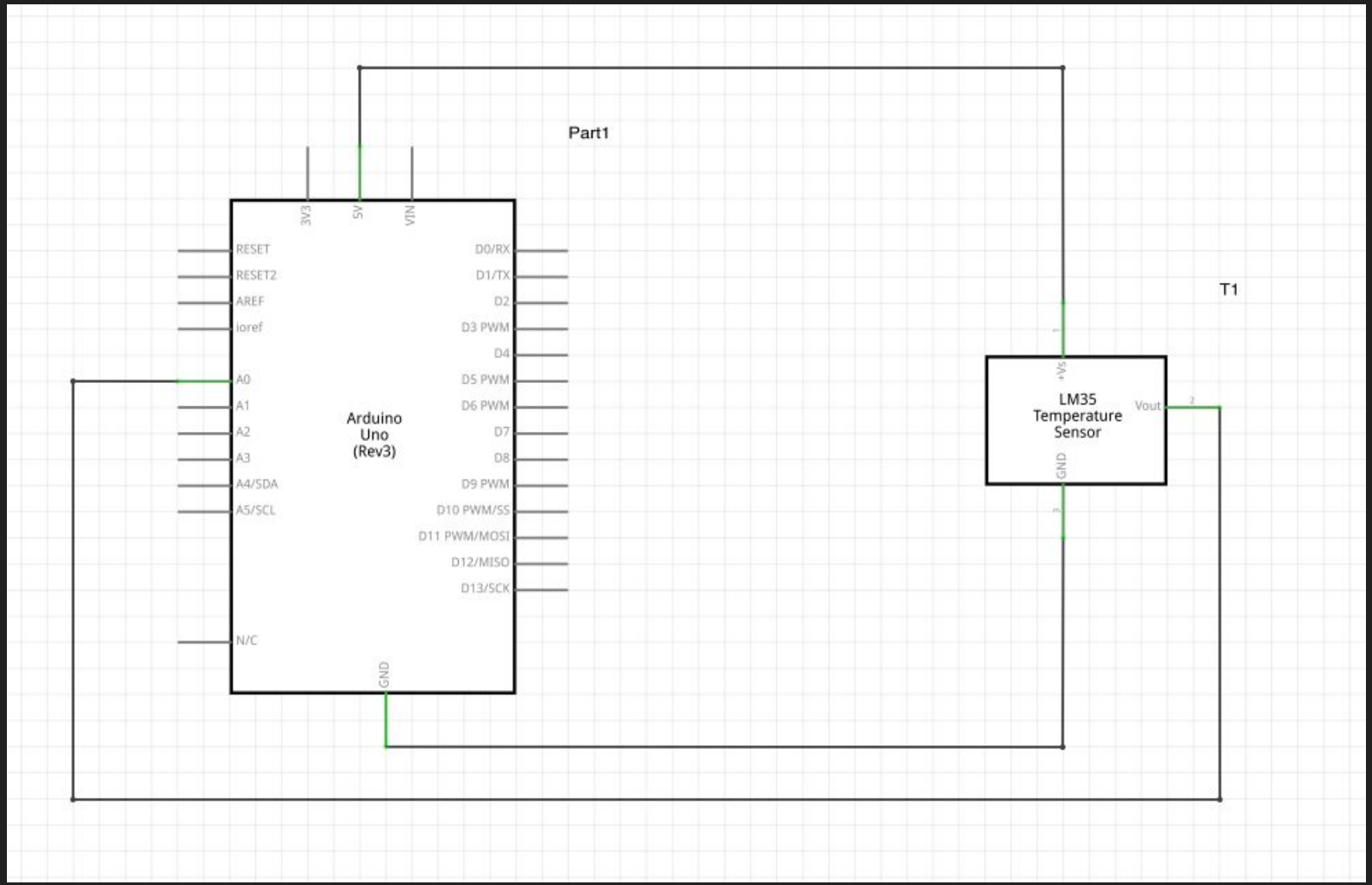


Essentially use a temperature sensor to detect temperature as reported in the serial Monitor

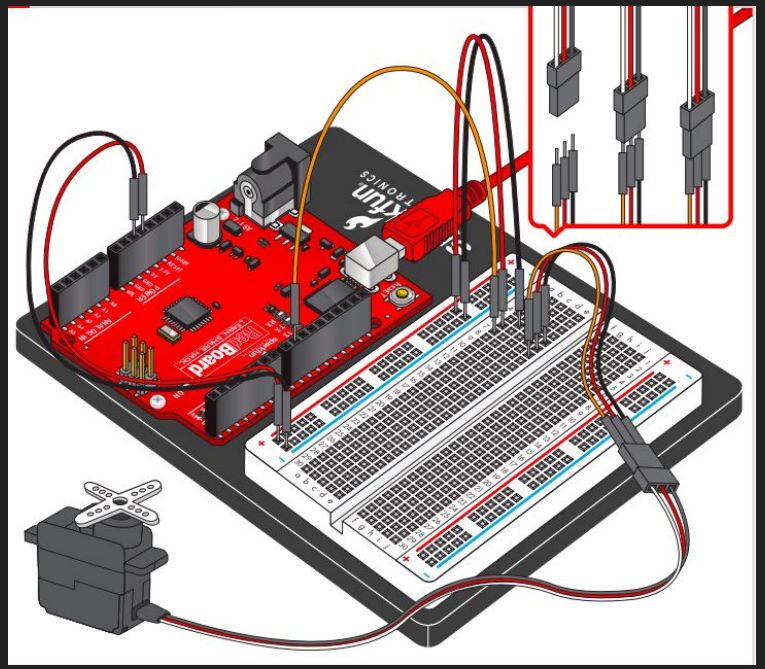
List of Components : Thermistor (x1), wires (x5)

Observations

We observed how the temperature sensor was using an analog pin as that analog value we get is essential for the equation we have the get the measure of temperature in degrees celsius and fahrenheit. It was coded so that the temperature sensor displays the voltage, temperature in degrees celsius and in fahrenheit. The Serial monitor displayed the temperature in both degree celsius and fahrenheit as it gave the room temperature to be around 23 degrees celsius and when I was close to touching it the temperature did sort of go up a couple degrees which made sense due to the warmth or heat gain.


Part1
3V3
5V
VIN
RESET
RESET2
AREF
ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
GND
Arduino
Uno
(Rev3)
T1
+Vs
LM35
Temperature
Sensor
Vout
GND

Lab 08 Single Servo



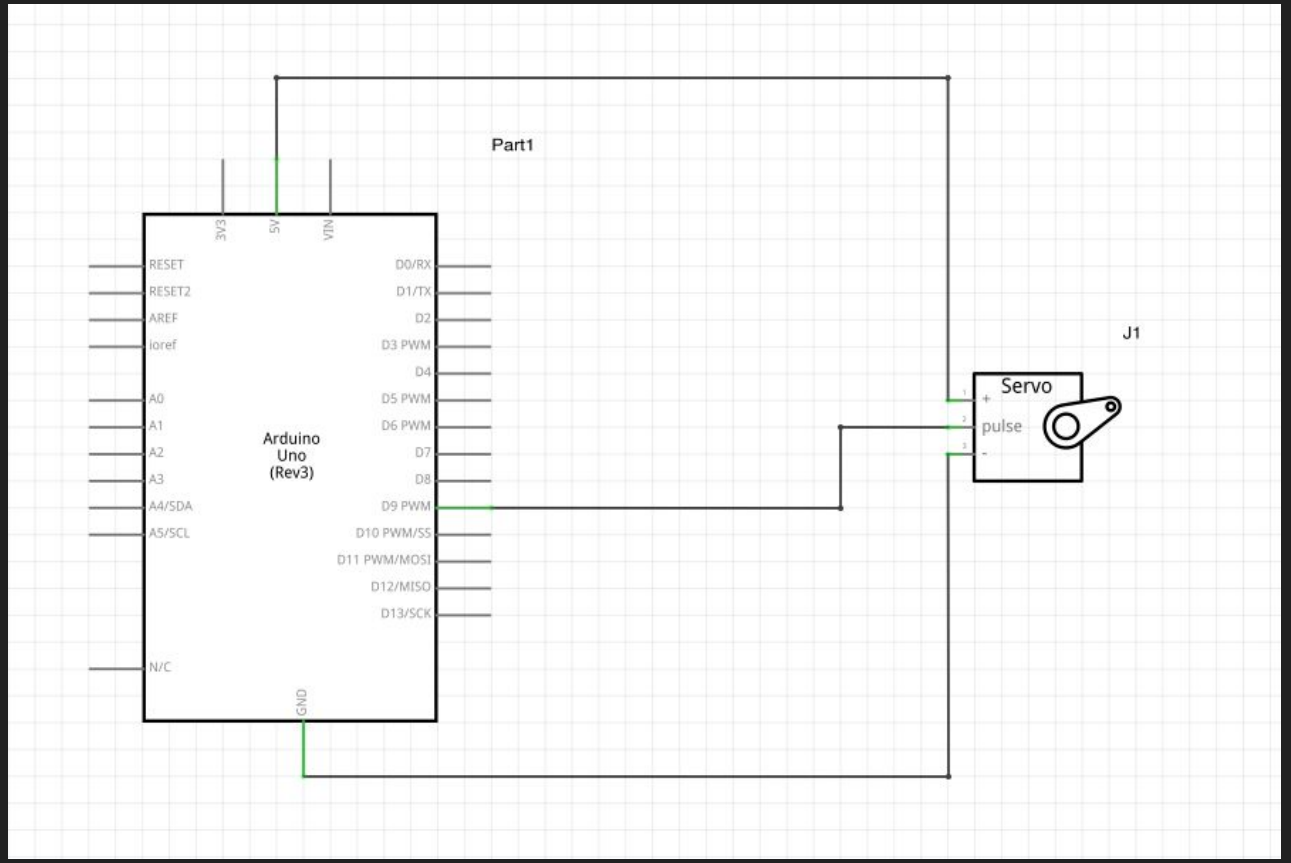
The Single Servo moves/turns in different angles as it goes left and right in different angles.

List of Components : Servo (x1), wires (x5)

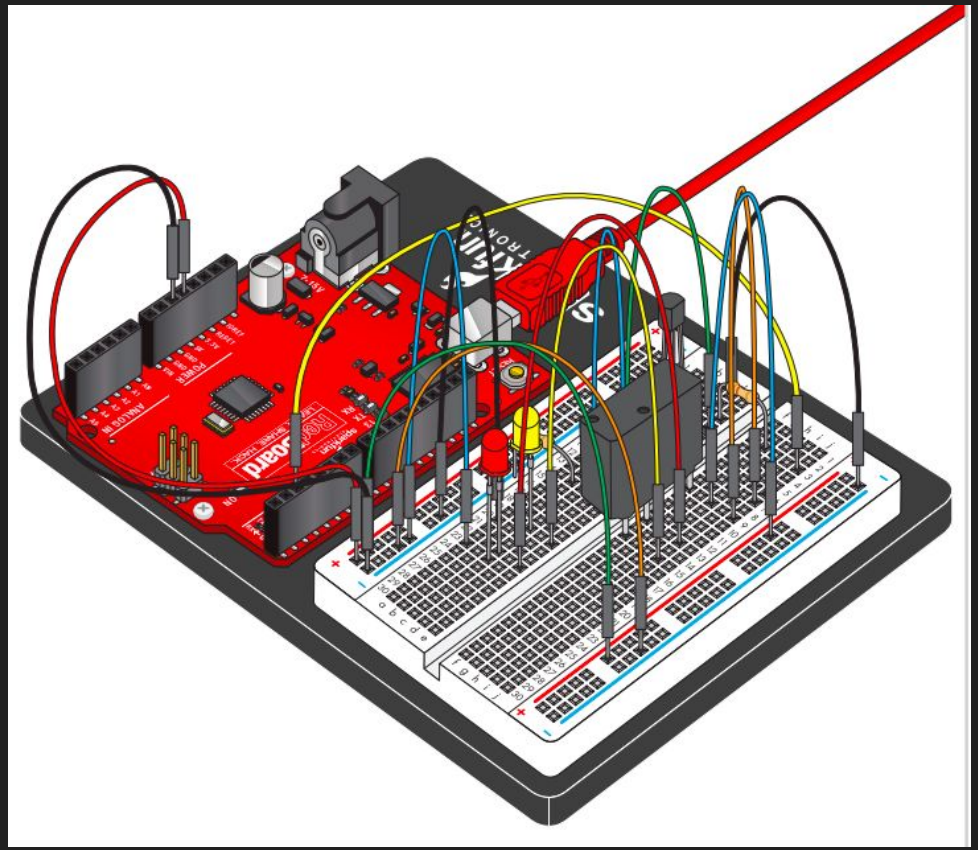
Observations

In this lab I observed how the single servo moved from 0° to 180° and back from 180° to 0° through clockwise and counterclockwise rotations. I also noticed that the fan on the servo would spin fast and slow at various speeds at different locations as a single servo consists a DC motor, potentiometer and a control circuit. So as the Dc motor rotates, the potentiometer resistance changes amd then the control circuit can then precisely regulate the movement there was and in what direction. These servos are practically controlled through sending an electric pulse of variable width (pulse width modulation) through the control wire as pin 9 was used as the PWM pin. The PWM that is sent to the motor essentially determines the position of the shaft of the servo and through the duration of the pulse, the rotor will essentially turn the position accordingly.

Schematic


Part1
3V3
5V
VIN
RESET
RESET2
AREF
Ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
GND
Arduino
Uno
(Rev3)
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
J1
Servo
+
pulse
-

Lab 13 Relay

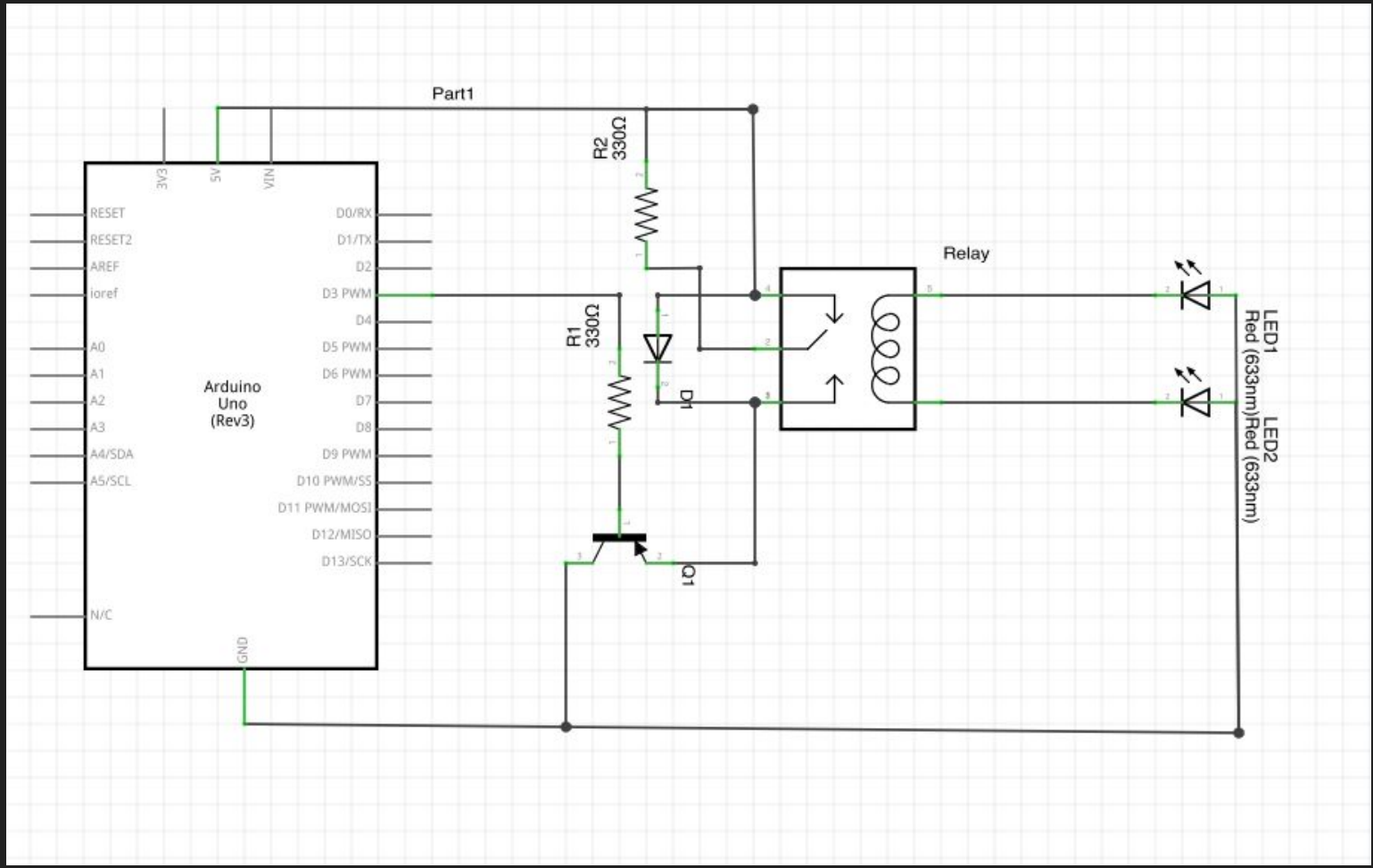


Making the LEDs blink one at a time at a delay (One off one on vice versa)

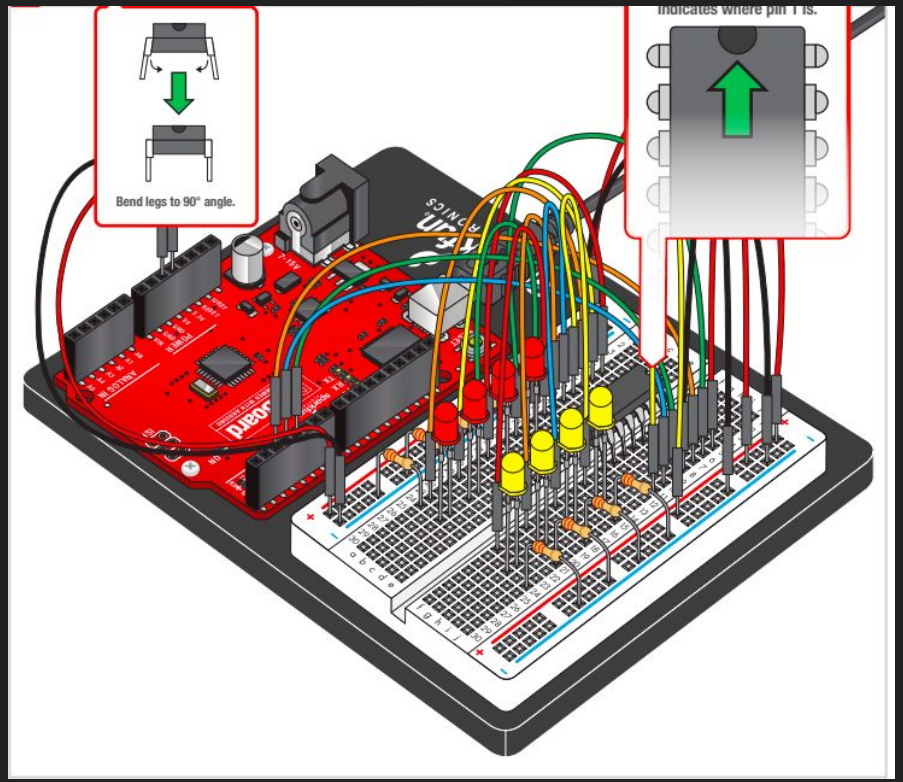
List of Components : Relay (x1), Transistor (x1), Diode (x1), 330Ω Resistor (x2), LEDs (x2), Wire (x14)

Observations

I observed how when the code was uploaded the relay started making a click sound and then I saw how the Led was blinking one by one; essentially they were alternating. Each time the LED would switch when one would turn off and one would turn on, the relay would make a click sound as this worked according to the delay set in the code. The intervals were around 1 second as they would alternate. This would seen applied in garage door openers as relays would be used in a similar concept. The blinking light intervals can also be used in various devices applications that have timed intervals.


Part1
R2
330Ω
R1
330Ω
Relay
D1
Q1
LED1
Red (633nm)
LED2
Red (633nm)
Arduino
Uno
(Rev3)
3V3
5V
VIN
GND
RESET
RESET2
AREF
Ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK

Lab 14 Shift Register

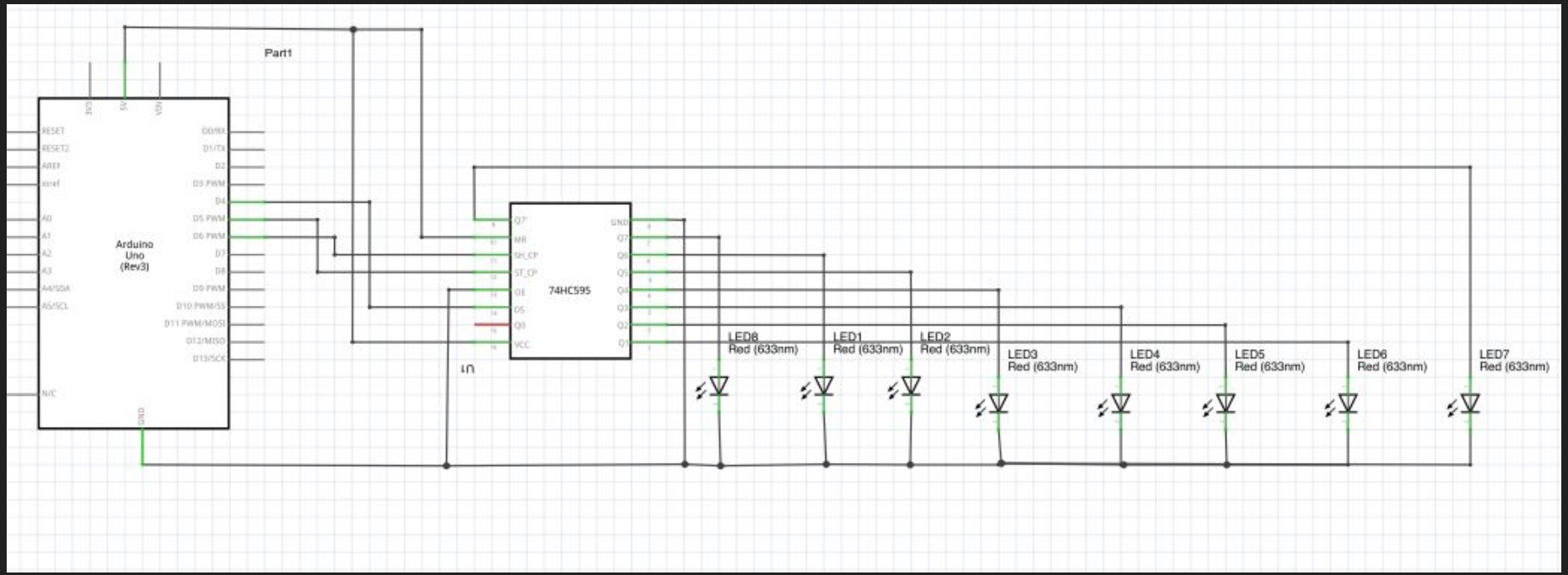


Essentially using the shift register to light up the led's in a count similar to that of multiple led's.

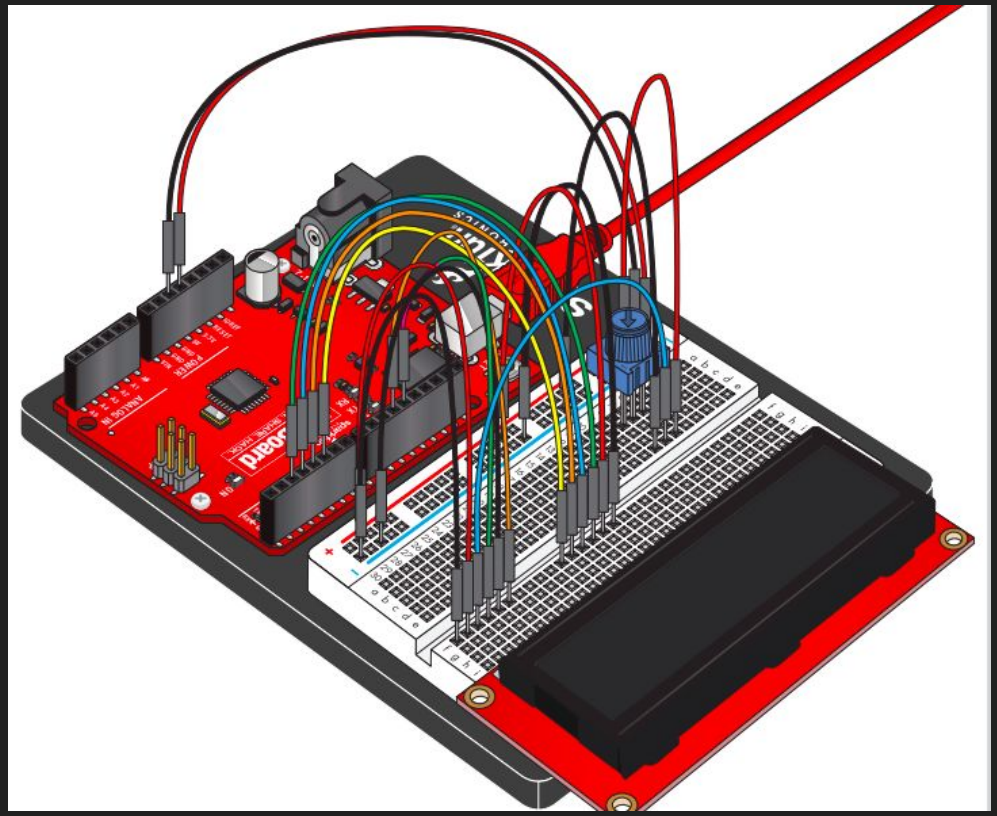
List of Materials: IC (x1), LED (x8), 330 ohms resistor (x8), wires (x19)

Observations

I observed how the LED’s sort of turn on in a numerical count to similar that of the Multiple LEd’s lab as the first LEd turned on and then turned off and then moved onto the second LED and so on all the way to the 8th led and reverse. We see how this is possible through the IC in contrast to the Multiple LED circuit. The Shift register holds the data in their memory which is moved or shifted to the positions that are required on each pulse. Through each clock pulse, the contents of the shift register shift one bit position to either the left or right. For which reason we see the count pattern occur with the LEDs similar to that of the multiple leds lab.


Part1
Arduino
Uno
(Rev3)
74HC595
LED8
Red (633nm)
LED1
Red (633nm)
LED2
Red (633nm)
LED3
Red (633nm)
LED4
Red (633nm)
LED5
Red (633nm)
LED6
Red (633nm)
LED7
Red (633nm)

Lab 15 LCD



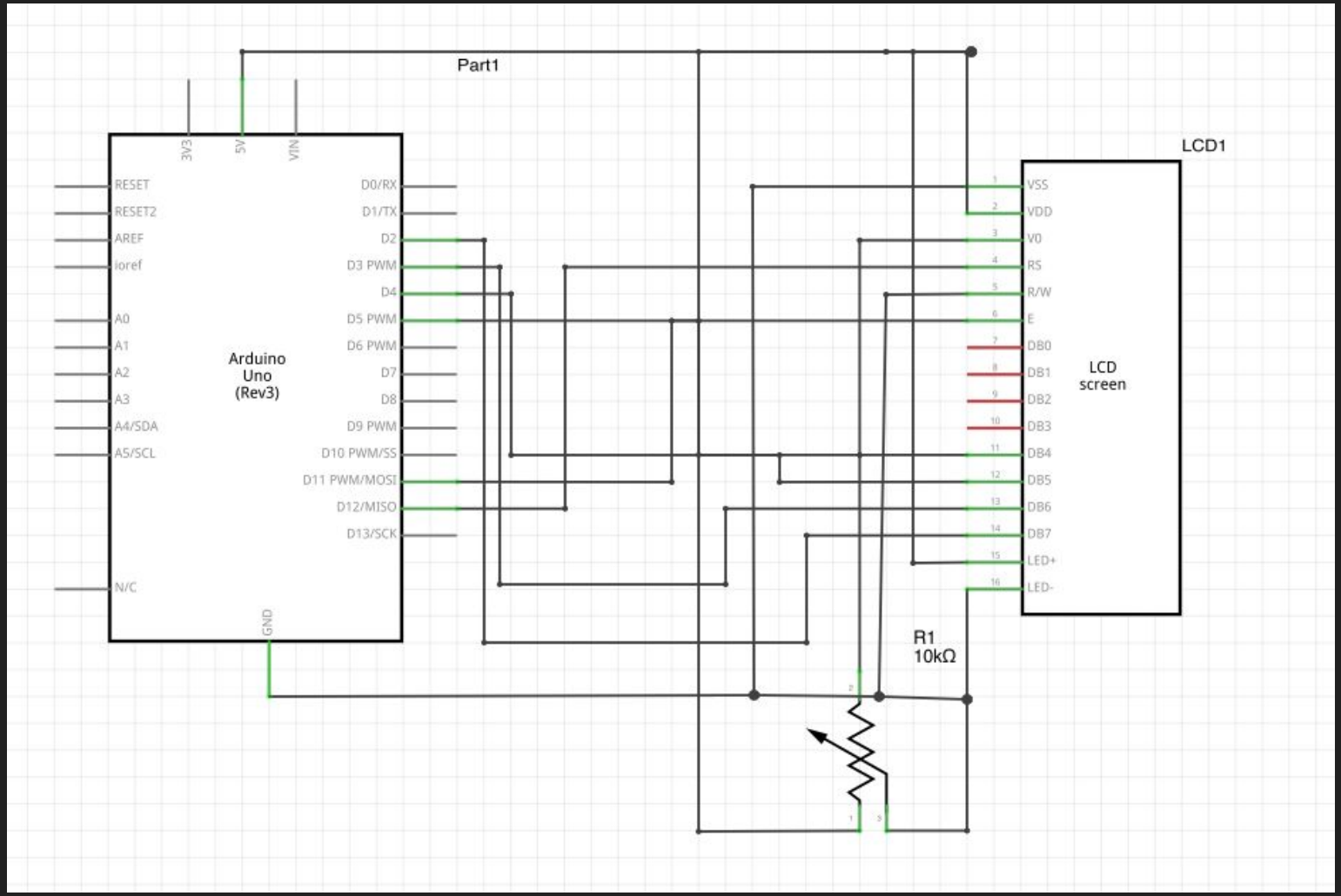
Having text characters/messages appear on the 16x2 LCD screen and adjust the screen/brightness resistance with the potentiometer.

List of Components: LCD (x1), potentiometer (x1), wires (x16)

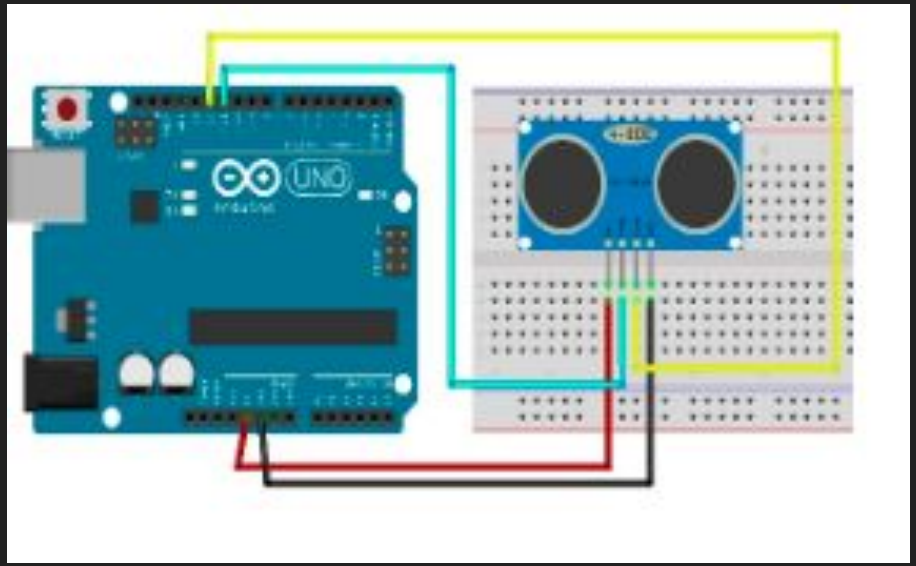
Had it say “Hello World” on the top line and a count (1,2,3,4,etc) in seconds in the bottom line as time passed.

Observations

I observed how the LCD had “Hello World” on the top line and had a count that was counting the seconds that had passed on the second line. There was also a potentiometer which changed the resistance of the LCD which was an important factor for the screen adjustment as without it the character or text characters wouldn’t appear for which a potentiometer let’s us adjust the right resistance for the characters to show without having the white boxes show but also show the characters show (perfect resistance contrast). The LCD has a chip in it which has all the individual dots that make up the display to make the characters given by the arduino. The Chip knowns the dot patterns that make up all text patterns. The LCD also uses the LiquidCrystal Library which is essentially a library implemented in the code to save work and time as the library does most of the hard work.


Part1
3V3
5V
VIN
RESET
RESET2
AREF
ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
GND
Arduino
Uno
(Rev3)
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
LCD1
VSS
VDD
V0
RS
R/W
E
DB0
DB1
DB2
DB3
DB4
DB5
DB6
DB7
LED+
LED-
LCD
screen
R1
10kΩ

Lab 16 UltraSonic Sensor



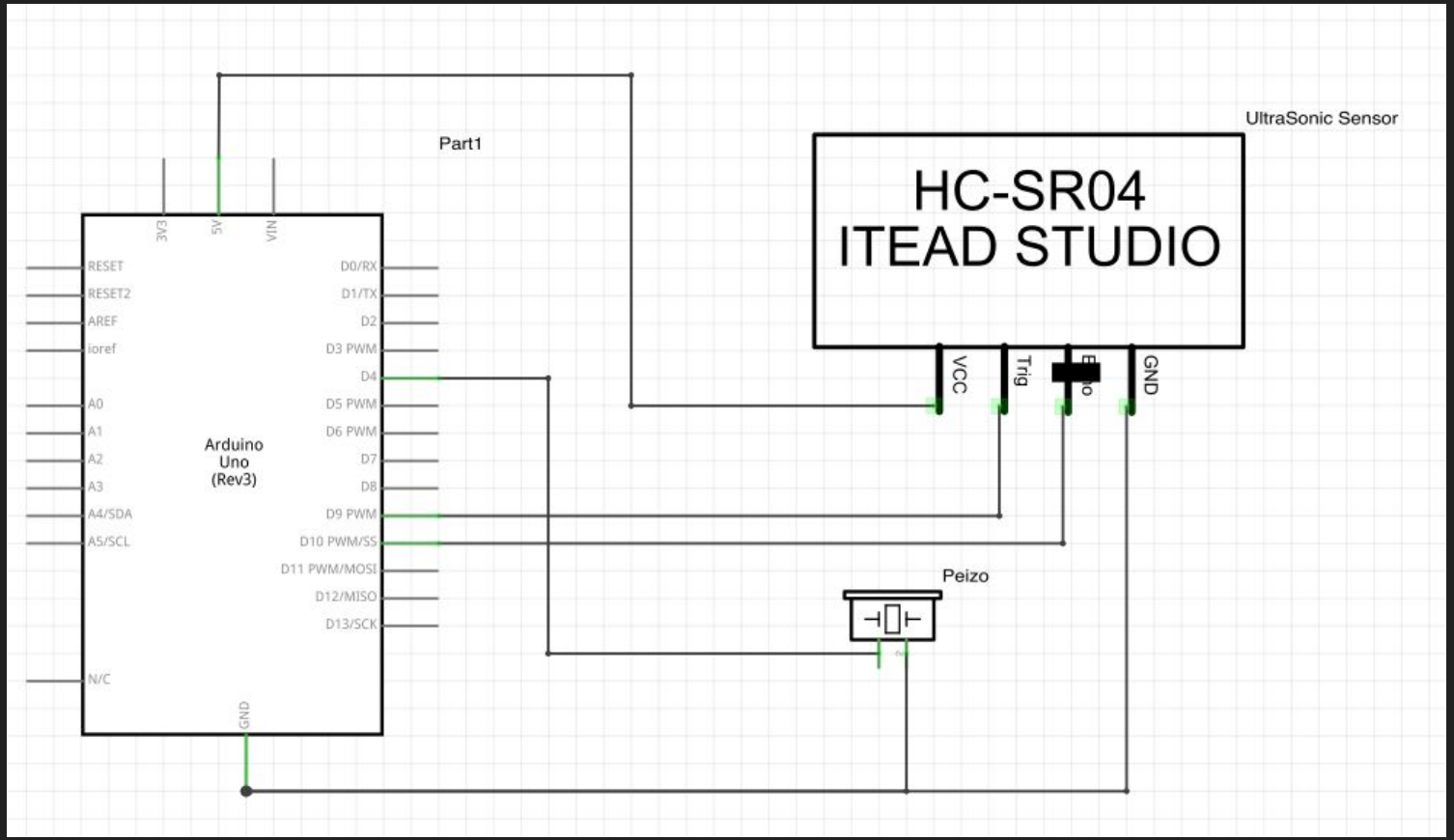
Using the Ultrasonic sensor to measure distance detected from the sensor.

List of Components : Ultrasonic sensor (x1), piezo buzzer (x1), wires (x6)

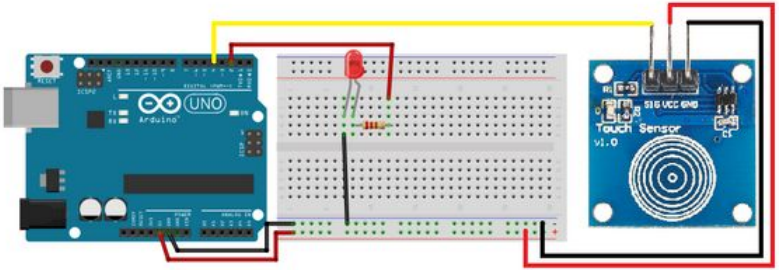
Used the sensor along with a piezo so it makes a high pitched sound close to the sensor and a lower pitch away from the sensor.

Observations

The Ultrasonic sensor detected the things close by as when I placed my hand near the sensor it gave values around 3 cm and when I placed my hand about the space of a ruler, it gave around 30 cm as the measurements were accurate. Also when the measurements were small (cm was small) the more high pitch the peizo became as the sound became higher in pitch as my hand was closer to the sensor and went it was further away the sound was lower in pitch. The Ultrasonic sensor is essentially using the trigger pin to send sound waves and then uses the echo pin to receive those sound waves when they return after bouncing off on an object. The time it takes for the waves to return is a sense of measure used to help determine distance as the sensor calculates it. This is calculated through the speed of the sound and then divided by 2 as the sound waves had to go there and had to travel twice the distance.


Part1
UltraSonic Sensor
HC-SR04
ITEAD STUDIO
VCC
Trig
Echo
GND
Arduino
Uno
(Rev3)
3V3
5V
VIN
RESET
RESET2
AREF
ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
GND
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
Peizo

Lab 17 Touch Sensor Lab



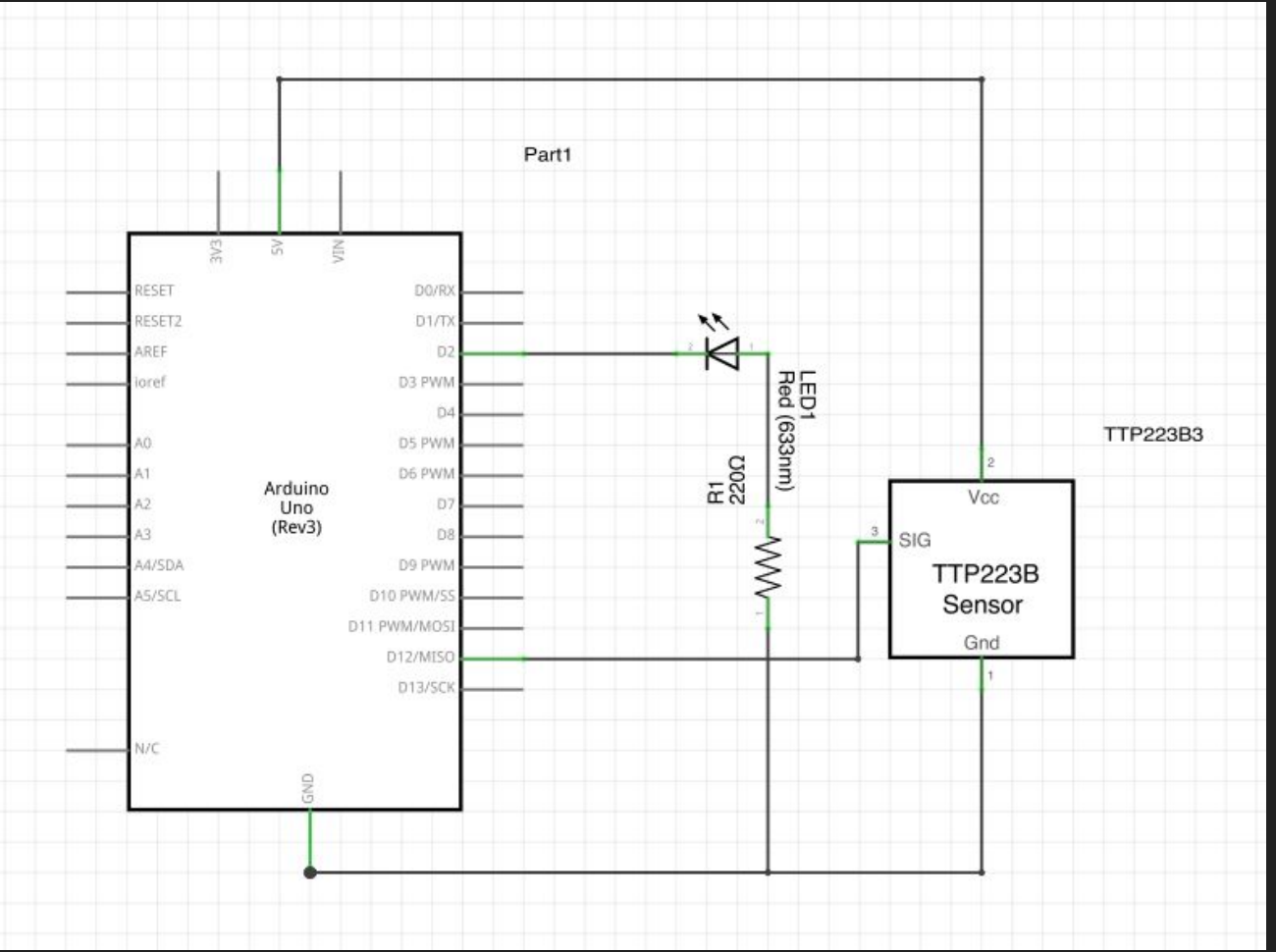
The LED turns on when the touch sensor senses a touch and stays off if no touch is detected.

List of Components : Touch sensor (x1), LED (x1), Wires (x6)

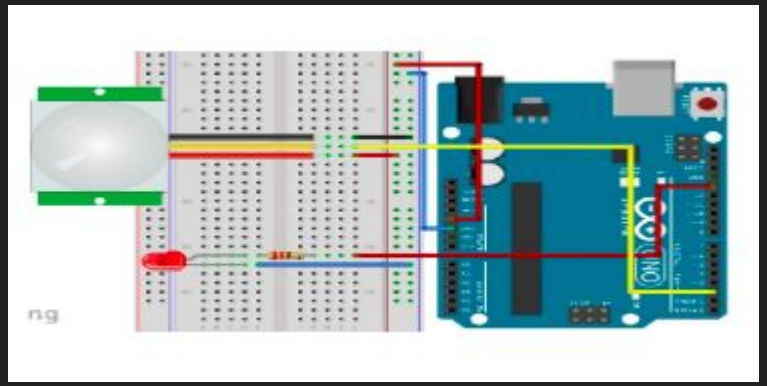
The LED was used to verify if a touch was being detected addition to the serial monitor as if touch was detected it turns on and if there isn’t it stays off.

Observations

I observed how when there was pressure or any sense of pressure detected by the touch sensor the sensor would detect it and the serial monitor would write “TOUCH” and the LED would turn on as the LED was coded to turn on when a touch was detected by the touch sensor. When there was no touch detected by the touch sensor, the LED remained off and the serial monitor displayed not touched signifying there was no sense of pressure meaning no touch. When there’s a touch or any pressure, it gets activated acts as a closed circuit which means current flows and the LED turns on and when it isn’t touch the path is broken and current doesn’t flow so the LED turns off.


Part1
3V3
5V
VIN
RESET
RESET2
AREF
Ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
GND
Arduino
Uno
(Rev3)
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
LED1
Red (633nm)
R1
220Ω
TTP223B3
Vcc
SIG
TTP223B
Sensor
Gnd

Lab 18 PIR sensor



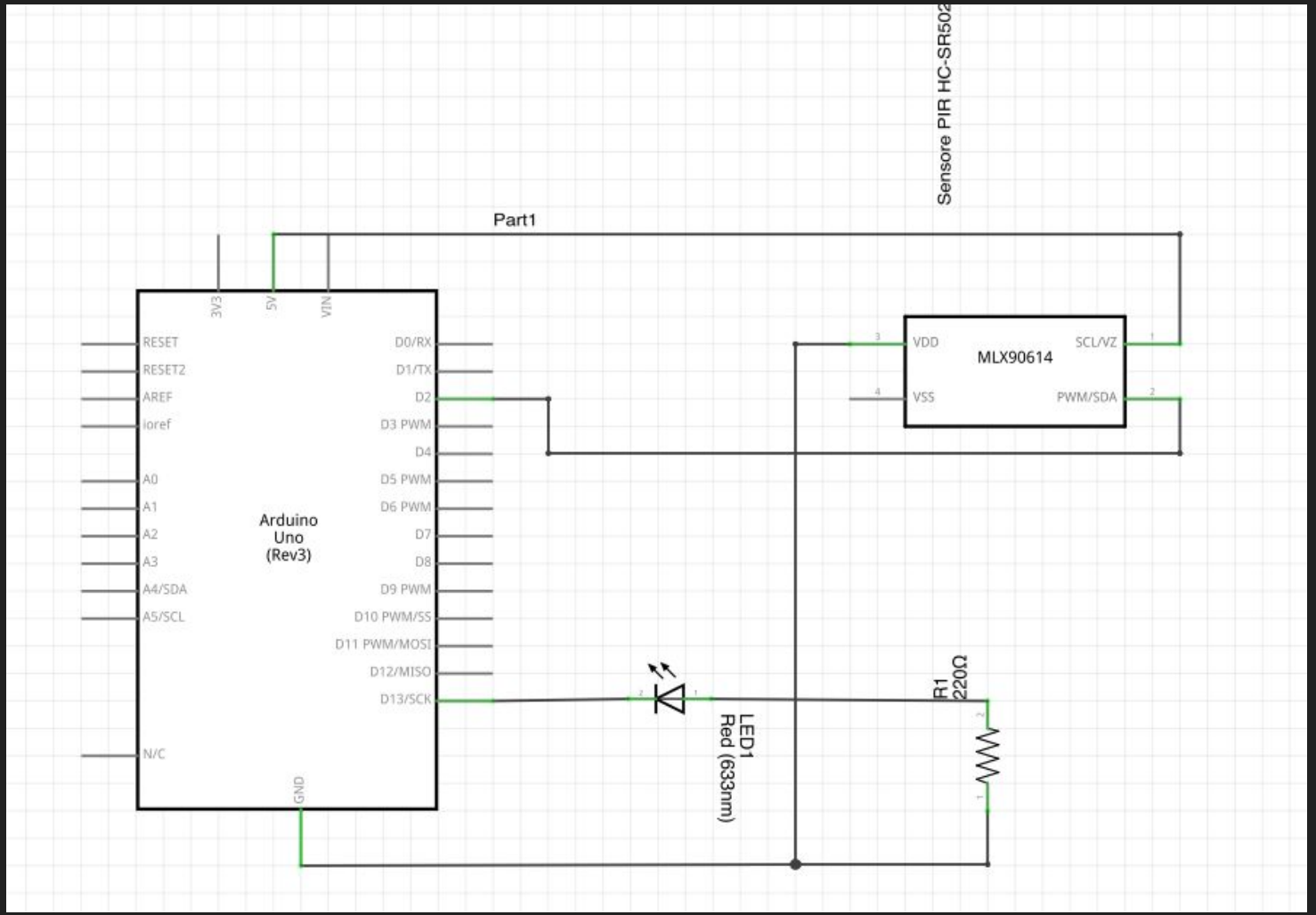
The PIR sensor detects the motion through the measure of infrared light as when it detects motion, the LED turns on and when it doesn’t, it turns off.

List of Components : PIR Motion Sensor (x1), LED (x1), 330 ohms resistor (x1), wires (x7 wires)

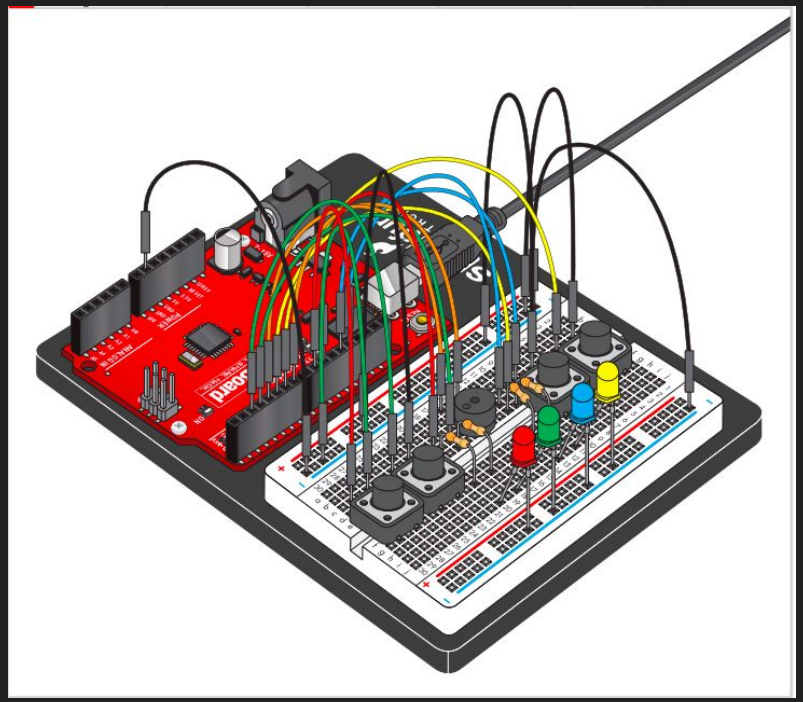
The LED and serial monitor are used to very if motion is detected or not from the PIR sensor

Observations

I observed how when I waved my hand near the sensor it took a couple seconds (2 seconds) but it detected that there was motion and displayed on the serial monitor that motion was detected and the LED turns on because motion was detected and when I move my hand show no motion to the sensor, the serial monitor displays “no motion detected” and the LED turns off because no motion was detected. This works through the PIR detect infrared light emitted or reflected by objects (people in this case).


Sensore PIR HC-SR502
Part1
3V3
5V
VIN
RESET
RESET2
AREF
Ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
GND
Arduino
Uno
(Rev3)
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
MLX90614
VDD
VSS
SCL/VZ
PWM/SDA
LED1
Red (633nm)
R1
220Ω

Lab 19 Simon Says



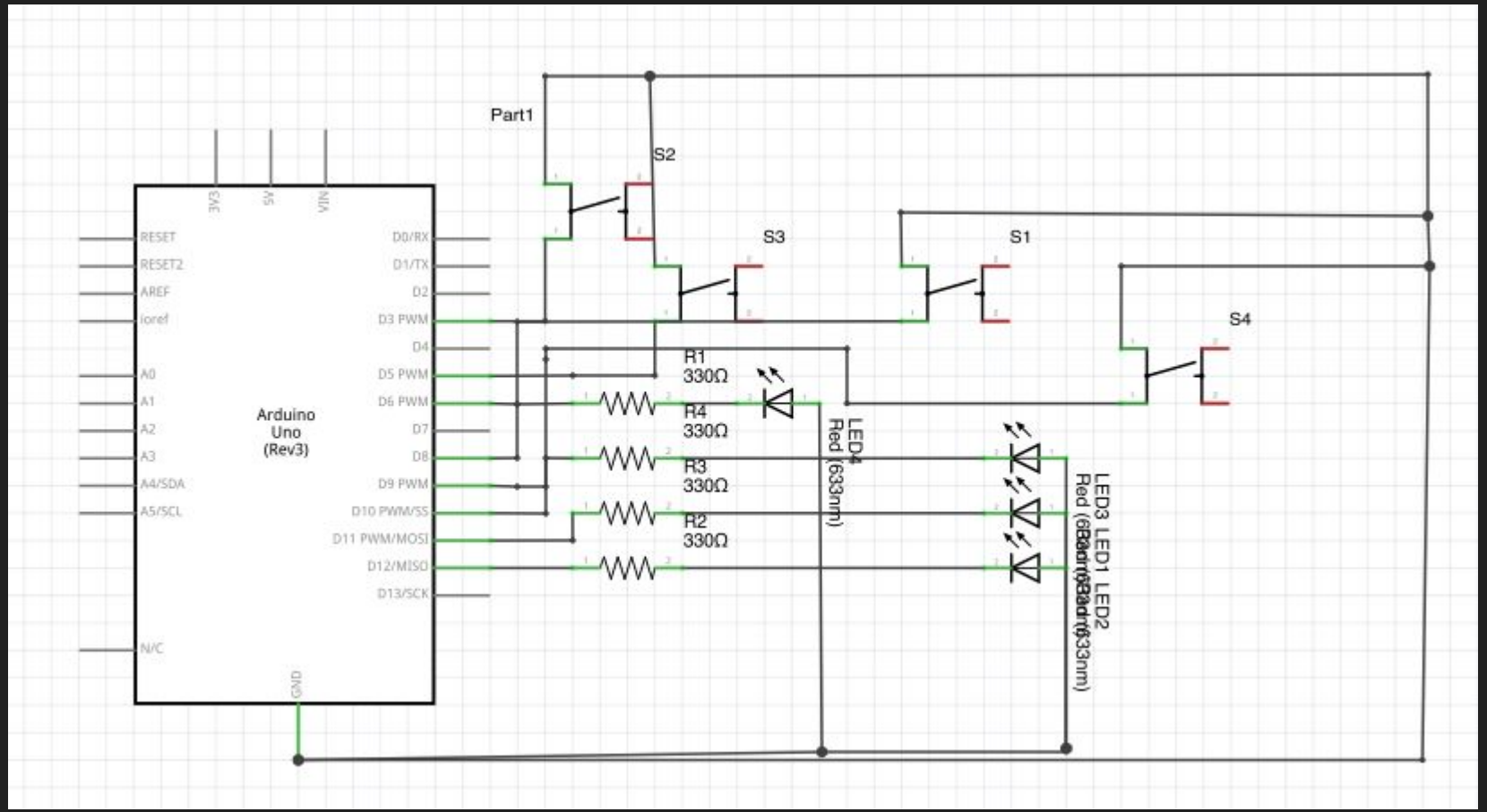
A simon says game with the use of 4 push buttons, 4 resistors and 4 led as it was coded in the concept of a simon says game.

List of Components : LEDs (x4), 330 ohms resistor (x4), push buttons (x4), wires (16x), piezo (x1)

The Piezo sounds acts as a way of telling if the answer was wrong or right as when its right it makes a small sound to move to next round but when wrong it makes a loud kind of noise and resets.

Observations

I observed how in the beginning how all four LEDS start blinking essentially tell us that the game is like the beginning stage or like waiting to be played. Once the button is pressed, the game begins and a random LED out of the four blinks. Then as the game progresses, each round it increases by one more random blink and gets longer and longer. You press the button associated with that colour in order to ensure you get it right. When the player gets it wrong the game reset and goes back to the random LED blinking stage which is basically like the game menu or the beginning before the game starts. A random number is essentially chosen between 1-4 as each number represent an LED. This number is then saved into an array each time as this ensures that the arduino remembers the order of the previous LEDs that blinked. The loop of adding a random number in the array keeps going until the player makes a mistake as that’s when it exists the loop and goes back to the random LED blinking which is basically the game menu.


Part1
S2
S3
S1
S4
R1
330Ω
R4
330Ω
R3
330Ω
R2
330Ω
LED4
Red (633nm)
LED3 LED1 LED2
Red (633nm)
Arduino
Uno
(Rev3)
3V3
5V
VIN
RESET
RESET2
AREF
ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
GND
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK