Workshop Basic Arduino Class 3 – Analog Inputs Mapping & Analog Outputs (PWM)

MSc. David Velásquez Rendón

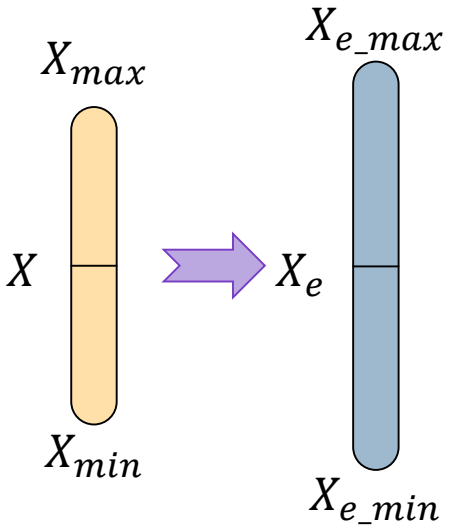
UNIVERSIDAD EAFIT®

Contents

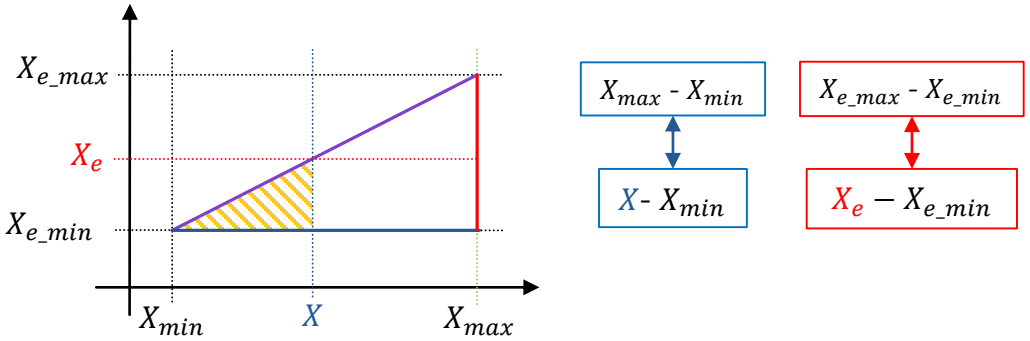
Analog variables mapping

- Allows to represent a variable with certain range into a different range.
  - Commonly used by microcontrollers to interpret voltage ADC RAW values.

In general:



To acquire X_e in terms of the X variable the following plot is made:



By geometric relations, we have:

$$\frac{X_{max} - X_{min}}{X - X_{min}} = \frac{X_{e\_max} - X_{e\_min}}{X_e - X_{e\_min}}$$

Solving for X_e we have:

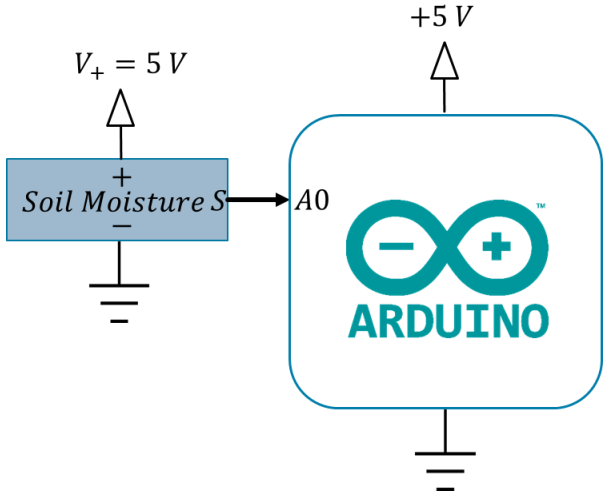
$$X_e = \frac{(X - X_{min})(X_{e\_max} - X_{e\_min})}{X_{max} - X_{min}} + X_{e\_min}$$

Analog Mapping Example – Soil Moisture Sensor

- Implement a program that calibrates from 0% to 100% a soil moisture sensor

Solution

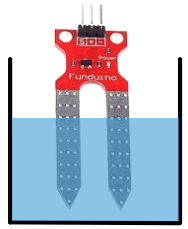
1. Connect the sensor to the Arduino following this image:



2. Upload a code that only monitors the raw analog ADC value of pin A0 and prints it on the Serial Monitor (Use potentiometer example as reference code)

3. Take note of the value shown in the Serial Monitor with the sensor in air. This value will be named ADC_{air}.

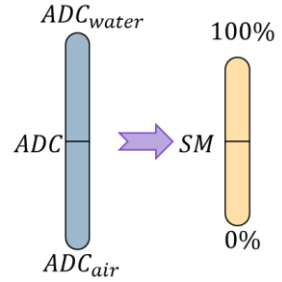
4. Submerge in the water the sensor as indicated in the image.



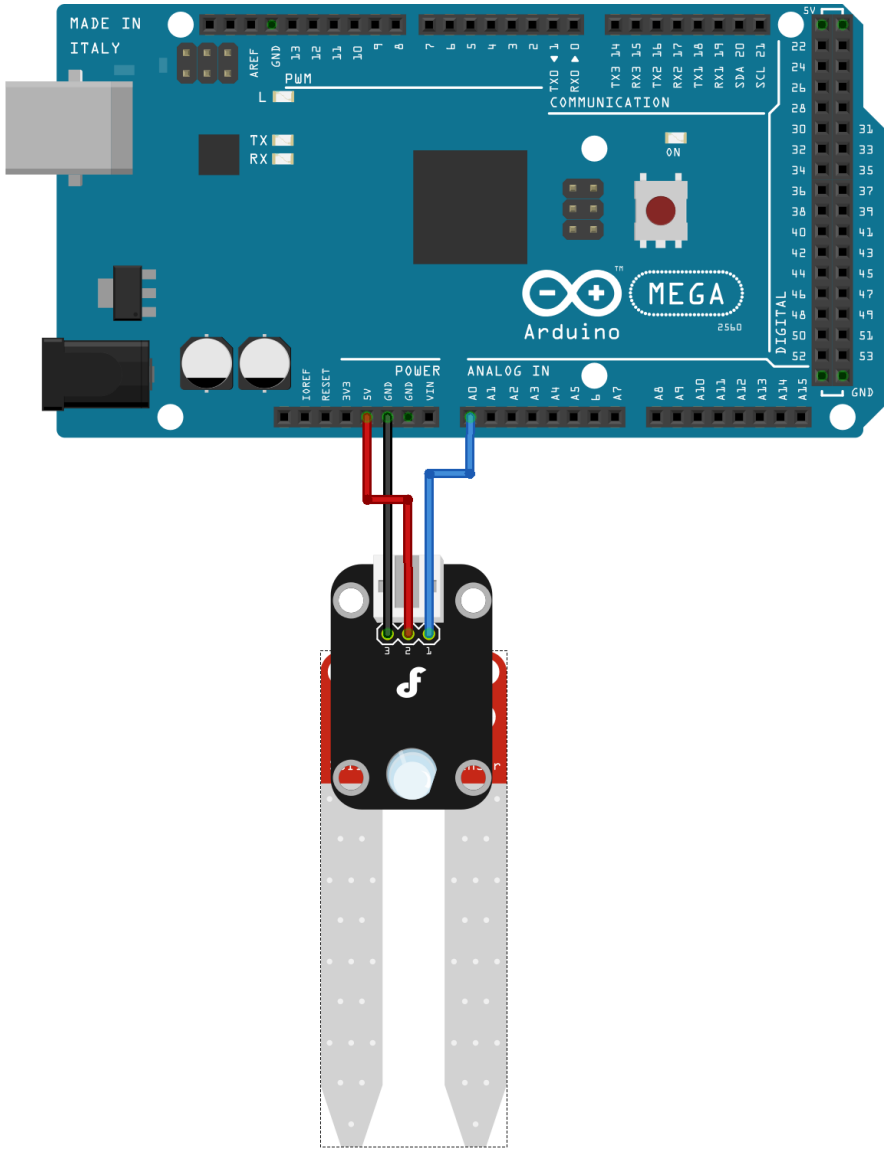
5. Take note again of the new value shown in the Serial Monitor with the sensor in the water,. This value will be named ADC_{water}

4. Apply the mapping formula to your code:

$$SM = \frac{(ADC - ADC_{air}) \cdot 100}{ADC_{water} - ADC_{air}}$$



Example 3.1 – Mapping Soil Moisture Sensor

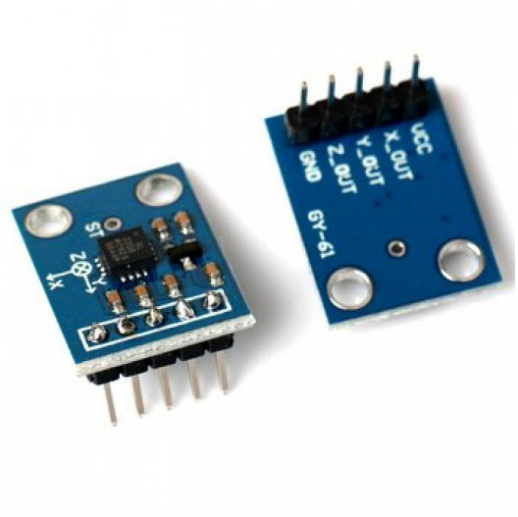


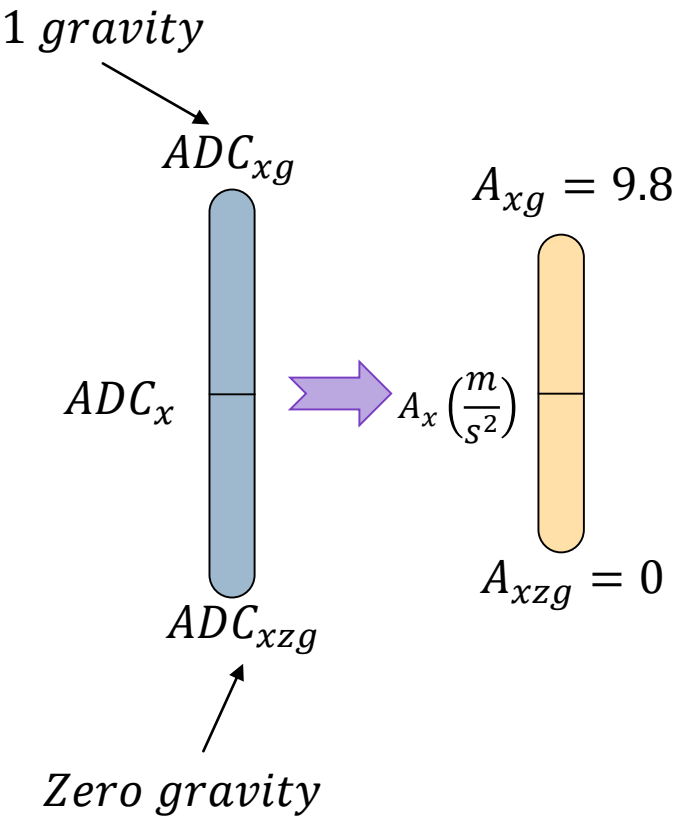
Example 3.1 – Mapping Soil Moisture Sensor


```
//I/O Pin Labeling
#define SMpin 0  //Soil Moisture sensor connected in pin A0

//Constants Declaration
const unsigned int ADCair = 200.0; //Constant to store the ADC value in the air of the
Soil Moisture sensor (200)
const unsigned int ADCwater = 800.0; //Constant to store the ADC value in the water of
the Soil Moisture sensor (800)

//Variables Declaration
unsigned int sm = 0; //Variable to store the percentage of the Soil Moisture

void setup() {
  //Communications
  Serial.begin(9600); //Begin Serial Communications with the computer by the Serial
0 port (TX0 RX0) at 9600 bauds
}

void loop() {
  sm = (analogRead(SMpin) - ADCair) * 100.0 / (ADCwater - ADCair);  //Soil Moisture
calculation using mapping formula
  Serial.print("Soil Moisture (%): ");
  Serial.println(sm);
}
```

Analog Mapping Challenge 1 – Accelerometer

- Do an Arduino program that prints on the Serial Monitor the accelerations in (m/s^2) of a Triple Axis Analog Accelerometer using Analog Mapping.

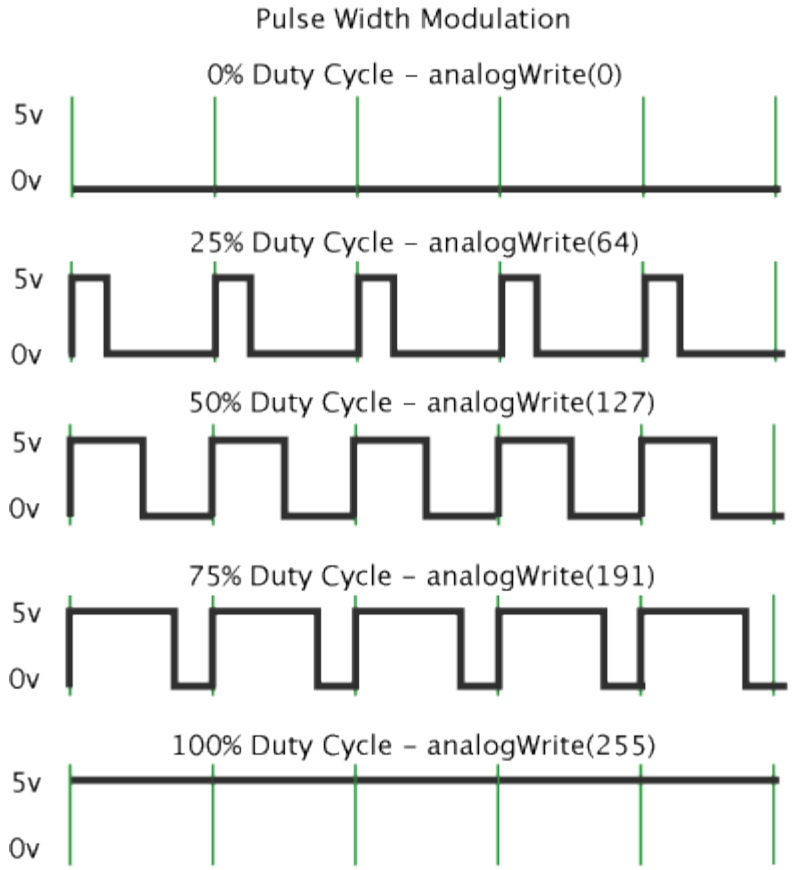


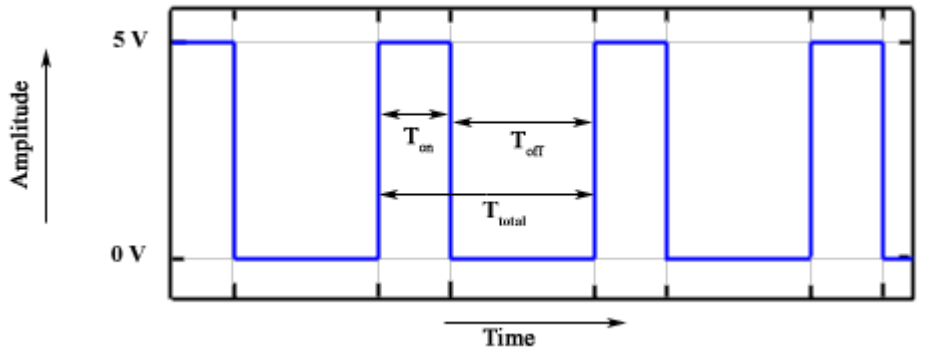


Hint: Create a function that returns a float and receives as input 5 floats (parameters) in order too reuse this function for the analog mapping of each axis of the accelerometer.

Analog Outputs - PWM

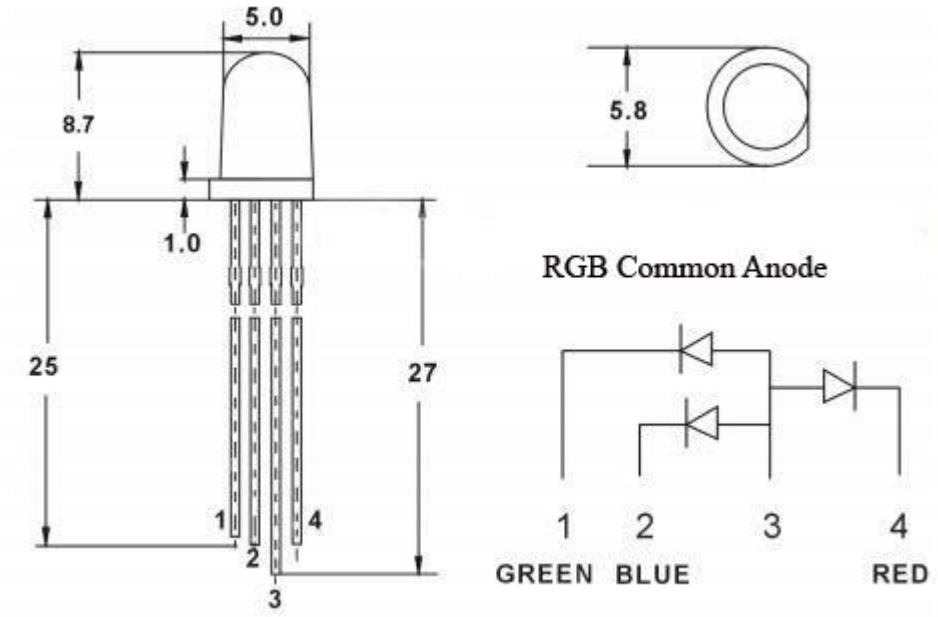
- It means Pulse Width Modulation (PWM).
- It's a **periodic** signal.
- It's **period** is always **constant** (for Arduino, the PWM frequency is **500 hz**), what changes is the **ON Time** of the digital signal.
- The **ON Time** is also called **Duty Cycle** because it represents the **quantity of energy** that is being injected to a system.
- Allows to "**vary**":
  - The **speed** of a **motor**.
  - The **temperature** of a **resistance**.
  - The **brightness intensity** of a **LED**.
- The function `analogWrite(VALUE)` is used in Arduino, where **`VALUE`** can be a number from **0 to 255**.
  - 0 means 0% of duty cycle.
  - 255 means 100% of duty cycle.



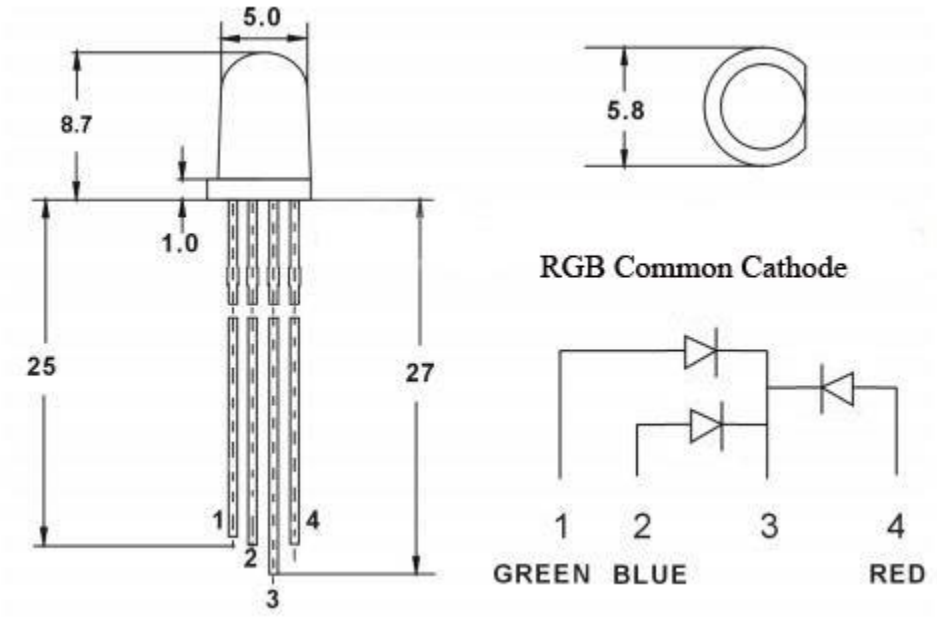


Analog Outputs – RGB LED

- They are 3 LEDs embedded in the same package. 2 types exist:
  - *Common Anode*



  - *Common Cathode*

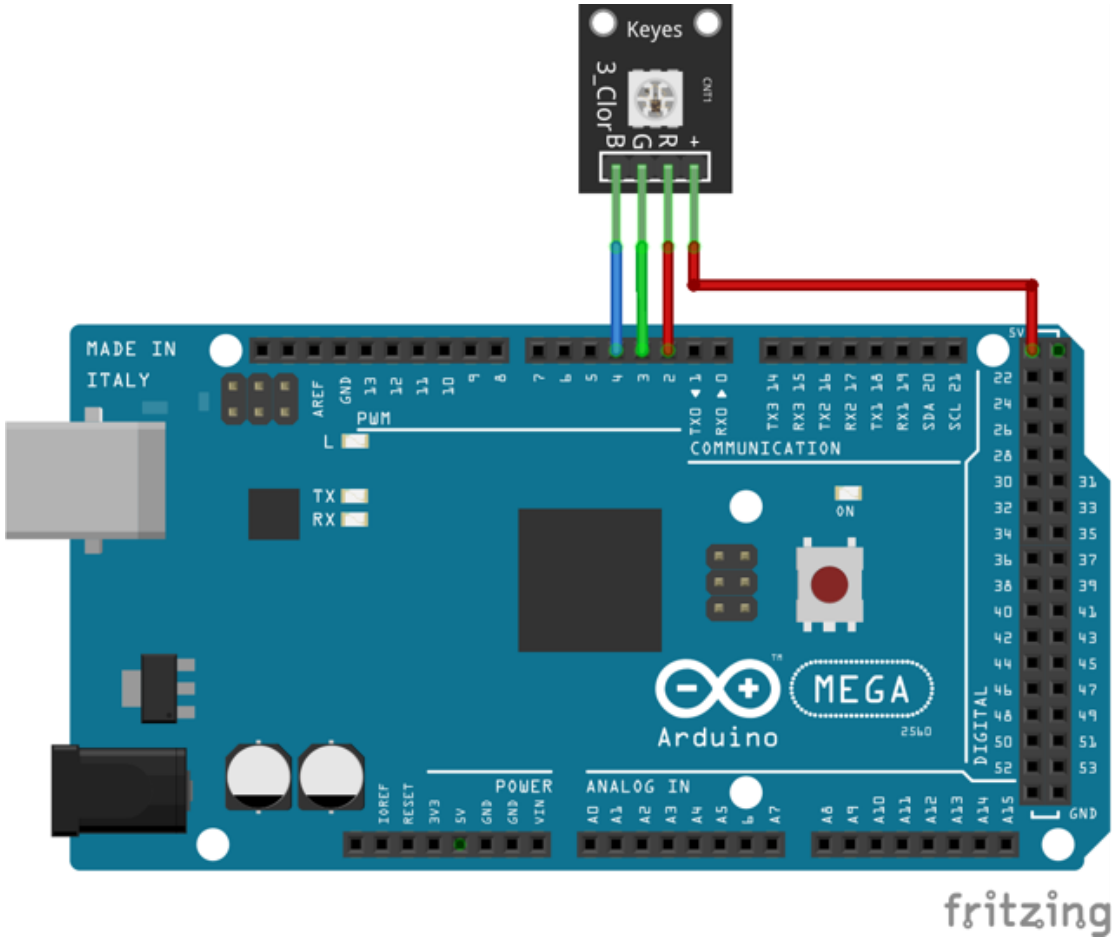


- A color combination happens depending on how much current is being injected to each color using `analogWrite(PIN, value)` function

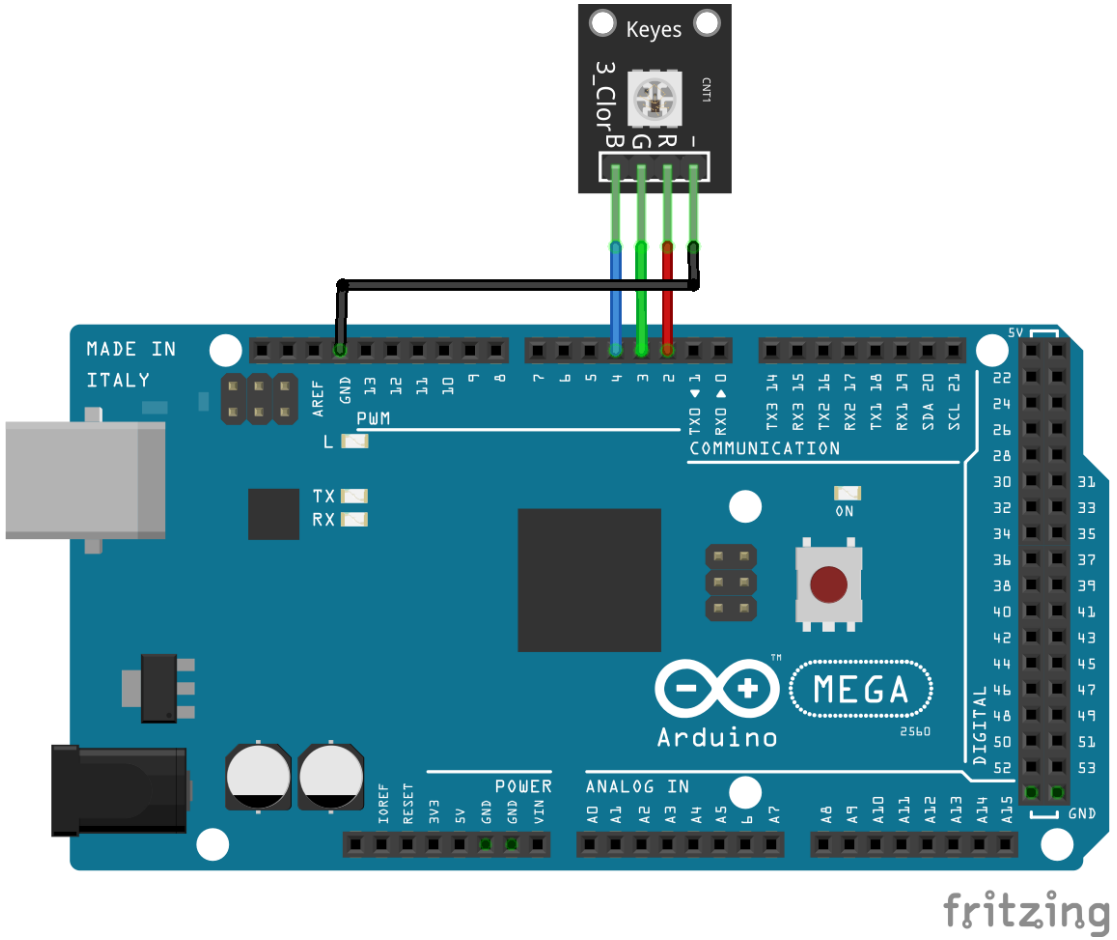
Example 3.2 – PWM RGB LED

- Do an Arduino program that varies randomly the color combination of a RGB LED

Common Anode



Common Cathode

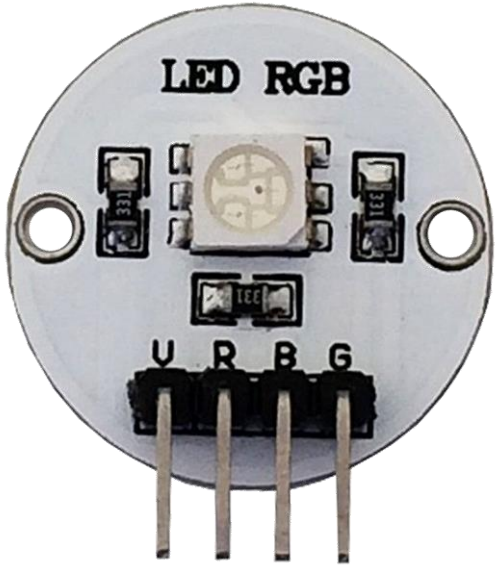


Example 3.2 – PWM RGB LED


```
//I/O Pin Labeling
#define LR 2  //Red segment of the RGB LED connected to pin 2
#define LG 3  //Green segment of the RGB LED connected to pin 3
#define LB 4  //Blue segment of the RGB LED connected to pin 4

void setup() {
 //I/O Pin Configuration
  pinMode(LR, OUTPUT);  //LR as Output
  pinMode(LG, OUTPUT);  //LG as Output
  pinMode(LB, OUTPUT);  //LB as Output

 //Physical Output Cleaning
  digitalWrite(LR, LOW);  //Turn OFF LR
  digitalWrite(LG, LOW);  //Turn OFF LG
  digitalWrite(LB, LOW);  //Turn OFF LB
}

void loop() {
  analogWrite(LR, random(256));  //Random analog writing to the red segment from 0~255
  analogWrite(LG, random(256)); //Random analog writing to the green segment from 0~255
  analogWrite(LB, random(256)); //Random analog writing to the blue segment from 0~255
  delay(200);  // 200 ms delay
}
```

Analog Outputs Challenge 2 – RGB LED + Joystick

- Do an Arduino program that varies the quantity of RED and GREEN segments of RGB LED through a Joystick (Using X rotation for RED and Y rotation for GREEN).



+

Hint: Map the joystick signals from 0 to 255 using the previous mapping function in Accelerometer Challenge

Thanks!