Práctica nº 8: LED RGB

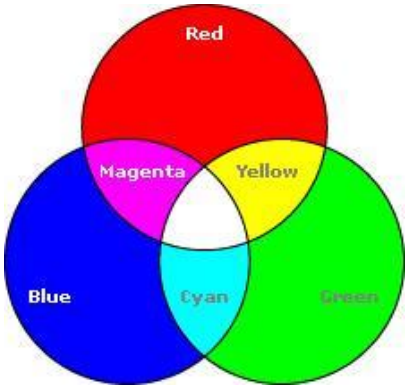
En esta práctica vamos a montar un pequeño circuito que nos permita gobernar el color que emite uno de éstos LEDs de RGB.

¿QUÉ ES UN LED RGB?

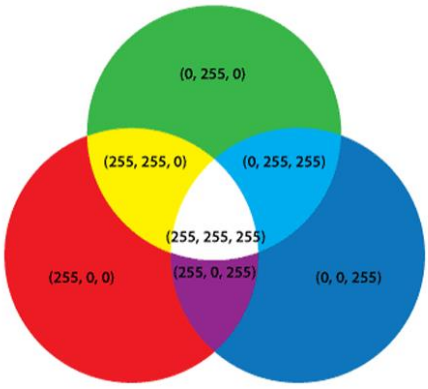
Hasta ahora hemos usado varias combinaciones de LEDS, pero siempre de un color definido. Habitualmente los rojos y amarillos son los más fáciles de conseguir, pero se pueden comprar también en tonos azules, verdes y hasta blancos. No suele haber grandes diferencias entre ellos excepto en el color.

Pero a veces es interesante disponer de una luz piloto que cambie de color según las condiciones. Por ejemplo, todos identificamos el verde como una señal de OK, mientras que el rojo indica problemas y el amarillo… bueno pues algo intermedio.

Poner varios diodos para hacer esto es engorroso y complica el diseño, así que estaría bien disponer de un diodo al que podamos indicar que color queremos que muestre. Esto es un **LED RGB**.



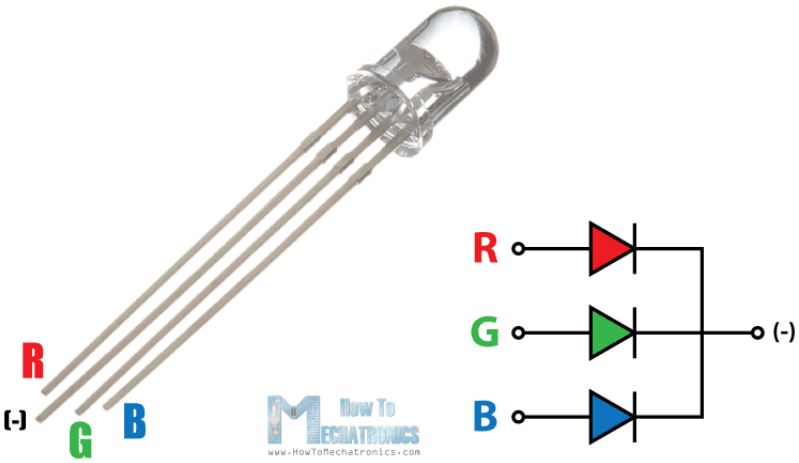
Los colores del LED RGB vienen representados con números comprendidos entre el valor 0 y el valor 255. De esta forma, para componer el color rojo pondríamos el valor máximo del rojo y el valor mínimo de los otros colores, es decir, el rojo equivale a "R=255; G=0; B=0". Y así sucesivamente con el resto de colores.



Existen dos tipos de LED RGB los de ánodo común y los de cátodo común.

En la práctica nosotros vamos a utilizar un LED RGB de cátodo común que es en realidad la unión de tres LEDs de los colores básicos (rojo, verde y azul), en un encapsulado común, compartiendo el Ground (el

negativo).Cada uno de éstos colores tiene su respectivo pin en el exterior de la cápsula, mientras que otro pin es común a todos los leds. En el caso del cátodo común, el pin que comparten todos los leds el negativo. En total el led RGB contará con 4 terminales.



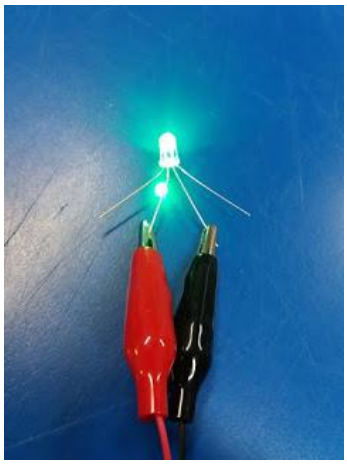
En función de la tensión que pongamos en cada pin podemos conseguir la mezcla de color que deseemos con relativa sencillez.

En función de la tensión (o ciclo de trabajo en el caso de PWM) que pongamos en cada pin podemos conseguir la mezcla de color que deseemos con relativa sencillez, por esa razón este producto es excelente para crear llamativos efectos bajo el control de tu microcontrolador.

¿CÓMO DIFERENCIAR UN RGB DE CÁTODO COMÚN Y UN RGB DE ÁNODO COMÚN?

CÁTODO COMÚN - La pata más larga es el negativo (-)

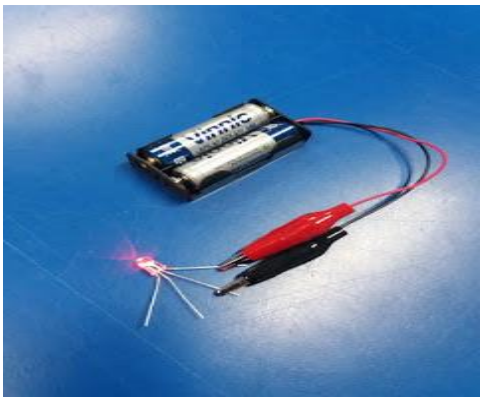
ÁNODO COMÚN - La pata más larga es el positivo (+)



Si tenemos un diodo Led RGB (red - green - blue) y no sabemos si tiene ánodo común o cátodo común, para identificar de que tipo es, haremos lo siguiente:

1. Localizar la pata más larga y seleccionar cualquiera de las otras 3 patas.

2. Utilizar un portapilas de 3V para realizar la prueba.
3. Poner la pata larga en el negativo de la pila (-) y la otra en el positivo (+), si luce, tenemos un RGB de cátodo común, si no luce, se trata de un RGB de ánodo común, en ese caso conectar la patilla más larga con el positivo (+) y cualquiera de las otras 3 en el negativo (-).



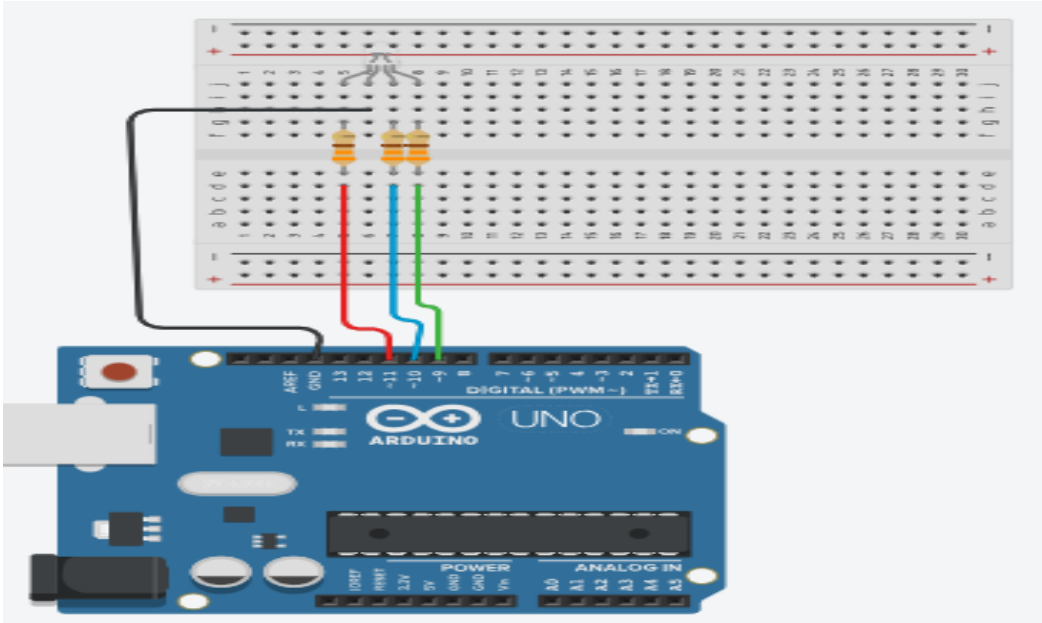
MATERIALES

- Arduino UNO.
- Cable USB tipo A-B.
- 1 placa Protoboard.
- Cables de conexión.
- Un LED RGB de cátodo común
- 3 Resistencia de 330Ω.

UTILIZANDO EL POTENCIÓMETRO CON ARDUINO

Vamos a montar un pequeño circuito que nos permita gobernar el color que emite uno de éstos LEDs de RGB.

MONTAJE



PROGRAMACIÓN.

Y el código que debes cargar en la placa es el siguiente.

LED_RGB

```
int redPin= 11;
    int greenPin = 9;
    int bluePin = 10;
    void setup() {
      pinMode(redPin, OUTPUT);
      pinMode(greenPin, OUTPUT);
      pinMode(bluePin, OUTPUT);
    }
    void loop() {
      setColor(255, 0, 0); // Red Color
      delay(1000);
      setColor(0, 255, 0); // Green Color
      delay(1000);
      setColor(0, 0, 255); // Blue Color
      delay(1000);
      setColor(255, 255, 255); // White Color
      delay(1000);
      setColor(170, 0, 255); // Purple Color
      delay(1000);
    }

// DEFINIMOS UNA FUNCIÓN LLAMADA "setColor" y le enviamos valores de rojo, verde y azul entre 0 y 255

    void setColor(int redValue, int greenValue, int blueValue) {
      analogWrite(redPin, redValue);
      analogWrite(greenPin, greenValue);
      analogWrite(bluePin, blueValue);

    }
```

Este programa debería producir una secuencia de rojo, verde, azul, blanco, violeta, y vuelta a empezar. Ahora, vamos a tratar de producir la secuencia rojo, verde, azul, pausa y vuelta a empezar utilizando la sentencia for:

LED_RGB_for §

```
void setup()
    {
        for (int i =9 ; i<12 ; i++)
             pinMode(i, OUTPUT);
    }
//Y después podríamos escribir una función como esta

void setColor(int R, int G, int B)
     {
          analogWrite(11 , R) ;    // Red - Rojo
          analogWrite(9, G) ;    // Green - Verde
          analogWrite(10, B) ;    // Blue - Azul
     }

void loop()
    {     setColor(255 ,0 ,0) ; // Red - Rojo
          delay(500);
          setColor(0,255 ,0) ;// Green - Verde
          delay(500);
          setColor(0 ,0 ,255) ;// Blue - Azul
          delay(500);
          setColor(0,0,0);
          delay(2000);
    }
```

Para que la secuencia se realice en sentid contrario azul, verde, rojo, parada, y vuelta a empezar:

LED_RGB_for2

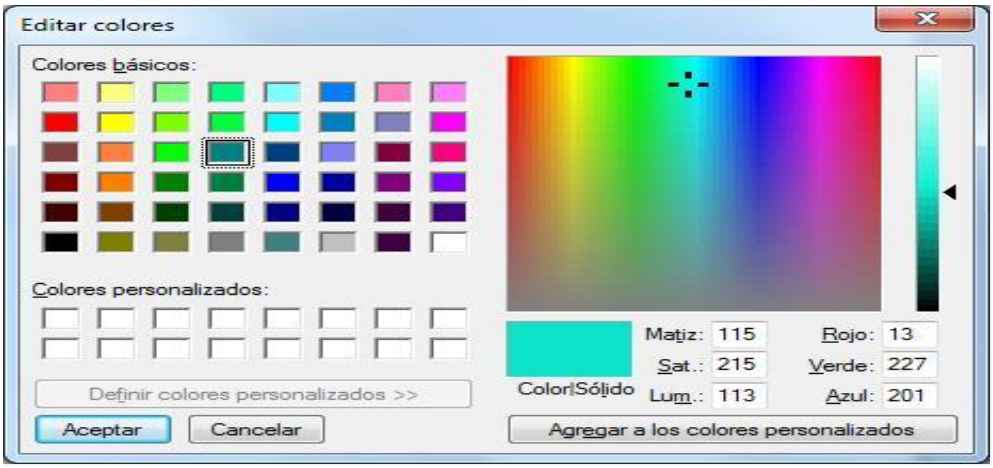
```
void setup()
    {
        for (int i =9 ; i<12 ; i++)
             pinMode(i, OUTPUT);
    }
//Y después podríamos escribir una función como esta

void setColor(int R, int G, int B)
     {
         analogWrite(11 , R) ;    // Red - Rojo
         analogWrite(9, G) ;    // Green - Verde
         analogWrite(10, B) ;    // Blue - Azul
     }

void loop()
    {     setColor(0 ,0 ,255) ; // Blue - Azul
          delay(500);
          setColor(0,255 ,0) ;// Green - Verde
          delay(500);
          setColor(255 ,0 ,0) ;// Red - Rojo
          delay(500);
          setColor(0,0,0);
          delay(2000);
    }
```

Conviene asegurarse de que hemos identificado correctamente los pines del RGB, porque de lo contrario, las mezclas posteriores de colores no serán lo que esperamos.

Vamos a ver como averiguar qué mezcla de RGB necesitamos para conseguir un color determinado utilizando el programa Paint incluido (en el menú de accesorios) de Windows.

Si arrancáis el Paint(o equivalente) suele tener un selector de colores:

Pulsándolo aparecerá algo parecido a esto:



Si vais pinchando en la zona de colores de la derecha, en la barra vertical aparecen los matices próximos al que habéis pinchado y podéis elegir el que más os guste. Debajo podéis ver la separación en RGB precisa para conseguir un tono determinado.

Así pues para conseguir ese tono de azulito de la imagen basta con que llaméis a:

```
Color(13, 227, 201) ;
```

Dado que Arduino nos permite escribir valores de 0 a 255 en los pines digitales, cuando utilizamos analogWrite(), en la práctica tendremos 255 x 255 x 255 colores diferentes o lo que es igual: 16.581.375 colores posibles.

EJERCICIOS PROPUESTOS

- Realiza cada uno de los montajes y comprueba que cada programa funciona correctamente.
- Realiza un montaje y prográmalo que muestre la siguiente secuencia de colores:
    - Rojo, verde, naranja, azul, amarillo, blanco, rosa, marrón, violeta, y vuelta empezar.
- Realiza una secuencia de al menos 5 colores utilizando la instrucción for.
- Ahora, utiliza la función random(N) que devuelve un valor al azar, comprendido entre 0 y N y en este caso, se presta especialmente bien para generar colores aleatorios en nuestro LED RGB. Probad esto:

LED_RGB_Radom

```
void setup()
      {
         for (int i =9 ; i<12 ; i++)
              pinMode(i, OUTPUT);
      }

    void loop()
    {
          setColor(random(255), random(255), random(255)) ;
          delay(500);
    }

void setColor(int R, int G, int B)
    {
          analogWrite(11 , R) ;    // Rojo
          analogWrite(9, G) ;    // Green - Verde
          analogWrite(10, B) ;    // Blue - Azul
    }
```

- Y describe lo que realiza este programa línea a línea.