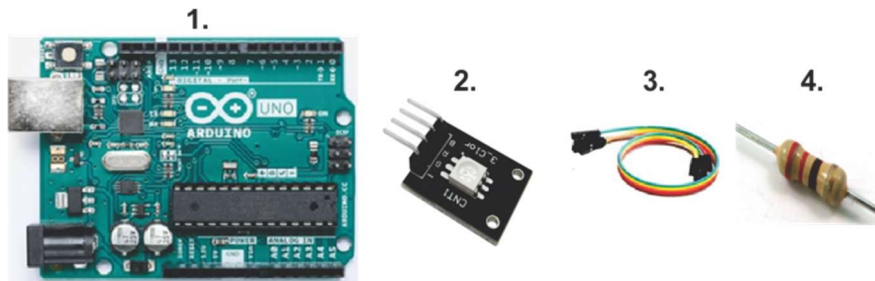


## Práctica 11: Semáforo con LED RGB

Para esta práctica se necesita:

1. Placa Arduino UNO
2. Módulo LED RGB
3. Cables para realizar las conexiones
4. Resistencia de 220 ohmios ( $220\Omega$ )



### Introducción

En esta práctica utilizaremos el módulo LED que contiene tres diodos LED con los tres colores primarios rojo, verde y azul (R=Red, G=Green, B=Blue). El módulo tiene tres entradas (una para cada LED de color) y un punto común denominado cátodo.

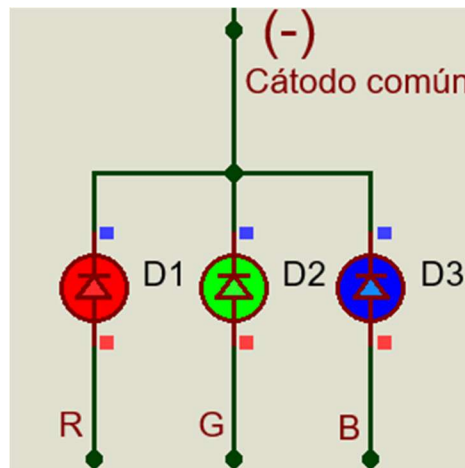


Fig. 1 LED RGB cátodo común

Podemos obtener otros colores mezclando estos tres colores primarios y variando la intensidad luminosa de cada uno. La intensidad luminosa se modifica variando la tensión de entrada en cada uno de los pines R, G y B. Controlando este módulo con la placa Arduino se pueden obtener efectos de iluminación interesantes.

Para esta practica vamos a realizar un proyecto sencillo tanto a nivel de programación como de componentes electrónicos. El proyecto consiste en un semáforo sencillo, el cual le programaremos un tiempo de espera de cinco segundos para el cambio de color de forma automática.

## Montaje

Las partes principales del módulo se puede ver en la figura 2.

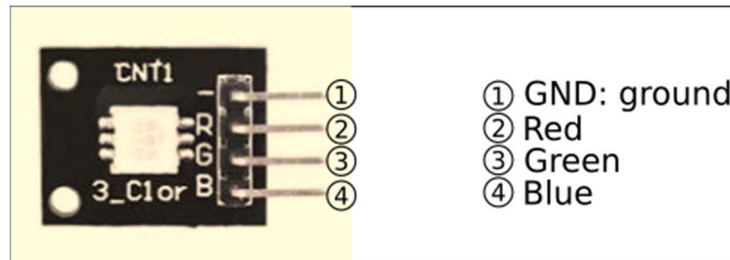
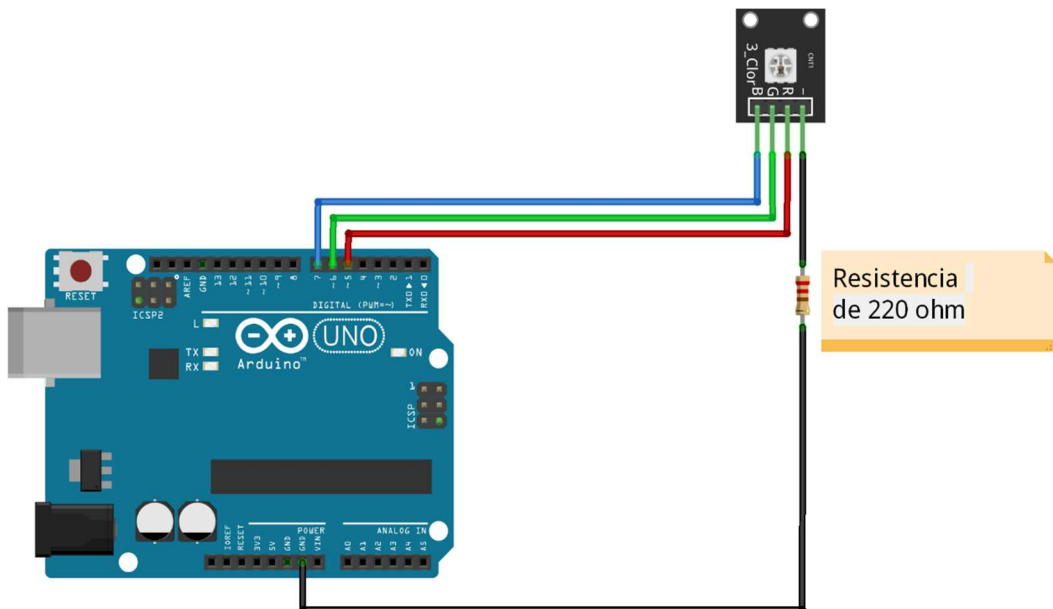


Fig. 2 Modulo LED RGB

Realizar la conexión del módulo cableando el terminal (–) a un extremo de la resistencia y el otro extremo de la resistencia al pin **GND**, el terminal “R” al pin **5**, el terminal “G” al pin **6** y el terminal “B” al pin **7** de la placa Arduino.

LED RGB	Componente	Pin Arduino
-	Resistencia	GND
R		PIN 5
G		PIN 6
B		PIN 7



## Programación

Vamos a plantear primero la lógica de funcionamiento que debe tener nuestro semáforo. Como vamos a tener tres colores (R, G y B) necesitamos tres salidas digitales de la placa Arduino. En este caso hemos seleccionado las salidas 5,6 y 7.

El siguiente paso es el control de cada salida. Para el semáforo sabemos que solo debe estar en encendido (ON) un color. Es decir, mientras un color está en encendido (ON) los otros dos deben estar apagados (OFF). Con esta claridad podemos ver que estaremos manejando los estados (ON/OFF) de los pines de salida en varias partes del código.

Para la secuencia de encendido nos faltaría el tiempo que dura un color en encendido para pasar al siguiente. Esto lo realizamos con un retardo que para esta practica la hemos definido en cinco segundos, para no tener que esperar mucho tiempo en ver las transiciones. Además, utilizaremos una variable para almacenar el valor del retardo. Esto con el fin de poder modificar el valor del tiempo una única vez y no tener que realizarlo en las diferentes partes del código donde se utilice.

Por último, debemos definir cómo será el inicio, es decir, al momento de conectar nuestra placa Arduino como deben estar las salidas. Aquí podemos definir si se inicia con un color en encendido (ON), los tres en encendido (ON) o los tres en apagado (OFF). Para este caso hemos decidido iniciar los tres colores apagados, esperar un segundo e iniciar la secuencia Rojo-Azul-Verde (R-B-G).

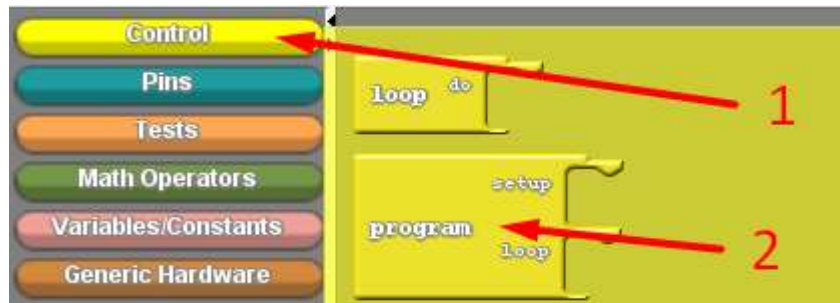
La secuencia del programa nos quedaría de la siguiente manera:

1. Definir variable para el tiempo de encendido (Retardo=5000)
2. Colocar las salidas en apagado (OFF) al inicio
3. Esperar un segundo para iniciar la secuencia
4. Colocar la salida **Red** en **ON** y las salidas **Green** y **Blue** en **OFF**
5. Esperar cinco segundos
6. Colocar la salida **Blue** en **ON** y las salidas **Green** y **Red** en **OFF**
7. Esperar cinco segundos
8. Colocar la salida **Green** en **ON** y las salidas **Red** y **Blue** en **OFF**
9. Esperar cinco segundos

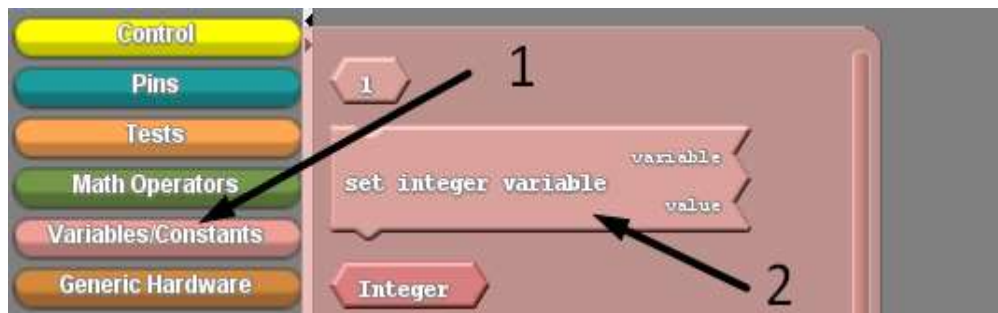
Con la secuencia del proceso pasamos a construir el código en Ardublock.

1. Quitamos el bloque **“loop”** de nuestro espacio de trabajo arrastrando y soltando en la parte inferior izquierda.

- Colocamos el bloque para tener la opción de la configuración inicial (setup). Ir al botón **"Control"** del panel izquierdo y seleccionar el bloque **program**.



- Colocar el bloque de variable entera para el retardo. Asignar el nombre Retardo y el valor en 5000.

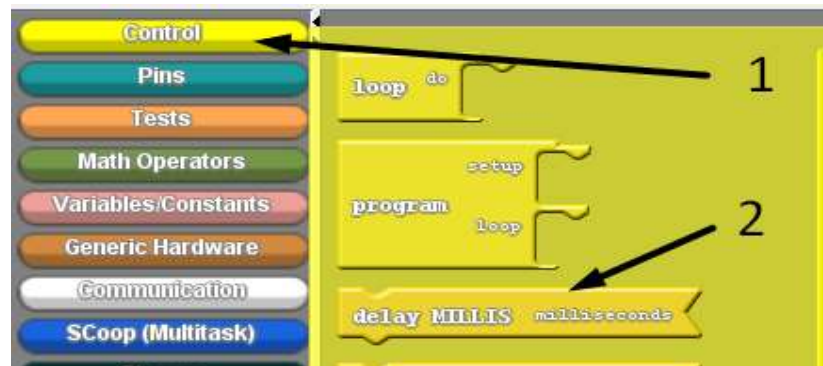


- Colocar tres bloques de salida digital, asignar los pines 5,6 y 7 sus estados en LOW. Recordemos que las conexiones las hemos realizado de la siguiente manera:

LED RGB	Pin Arduino
R (Red=rojo)	PIN 5
G (Green=verde)	PIN 6
B (Blue=azul)	PIN 7



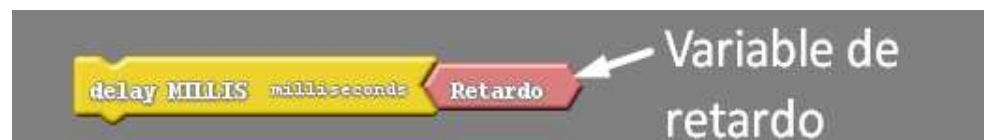
- Colocamos un bloque para un retardo de 1000 milisegundos (1 segundo).



- Colocar un bloque de salida digital y establecer la salida 5 en encendido (HIGH). En este caso se enciende el color rojo.



- Colocar el bloque de retardo y reemplazamos el bloque del número por el bloque de la variable que hemos creado en el punto 3.



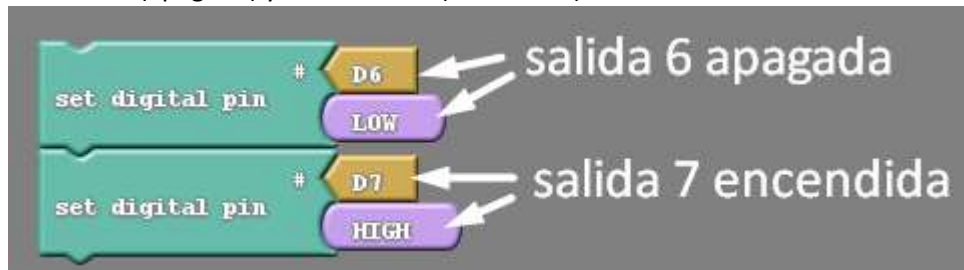
- Colocar dos bloques de salida digital, asignar los pines 5 y 6. A la salida 5 colocamos el estado en LOW (apagado) y la 6 en HIGH (encendido)



- Clonamos el bloque de retardo



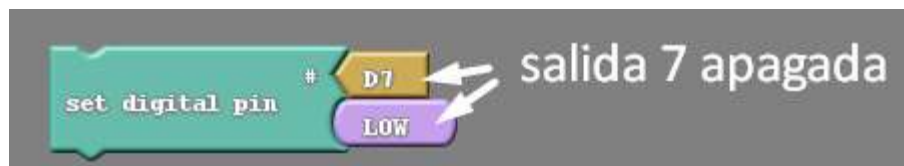
10. Colocar dos bloques de salida digital, asignar los pines 6 y 7. A la salida 6 colocamos el estado en LOW (apagado) y la 7 en HIGH (encendido)



11. Clonamos nuevamente el bloque de retardo



12. Para finalizar colocamos un bloque de salida digital asignamos el pin 7 y su estado en apagado (LOW)



Con esto hemos finalizado la inserción de bloques para nuestro semáforo. El uso de la variable **Retardo** nos facilita la modificación del tiempo que un color permanece encendido, porque solo debemos modificar el valor en el inicio del programa.



Finalmente, nuestro programa completo nos debe quedar de la siguiente manera:



## Ampliación

¿Cómo podríamos mejorar el programa para que este inicie con un botón (pulsador)?

Amplia el alcance del proyecto para tener dos semáforos, uno para peatones y el otro para vehículos, de tal manera que cuando el paso vehicular este en verde el de peatones este en rojo.