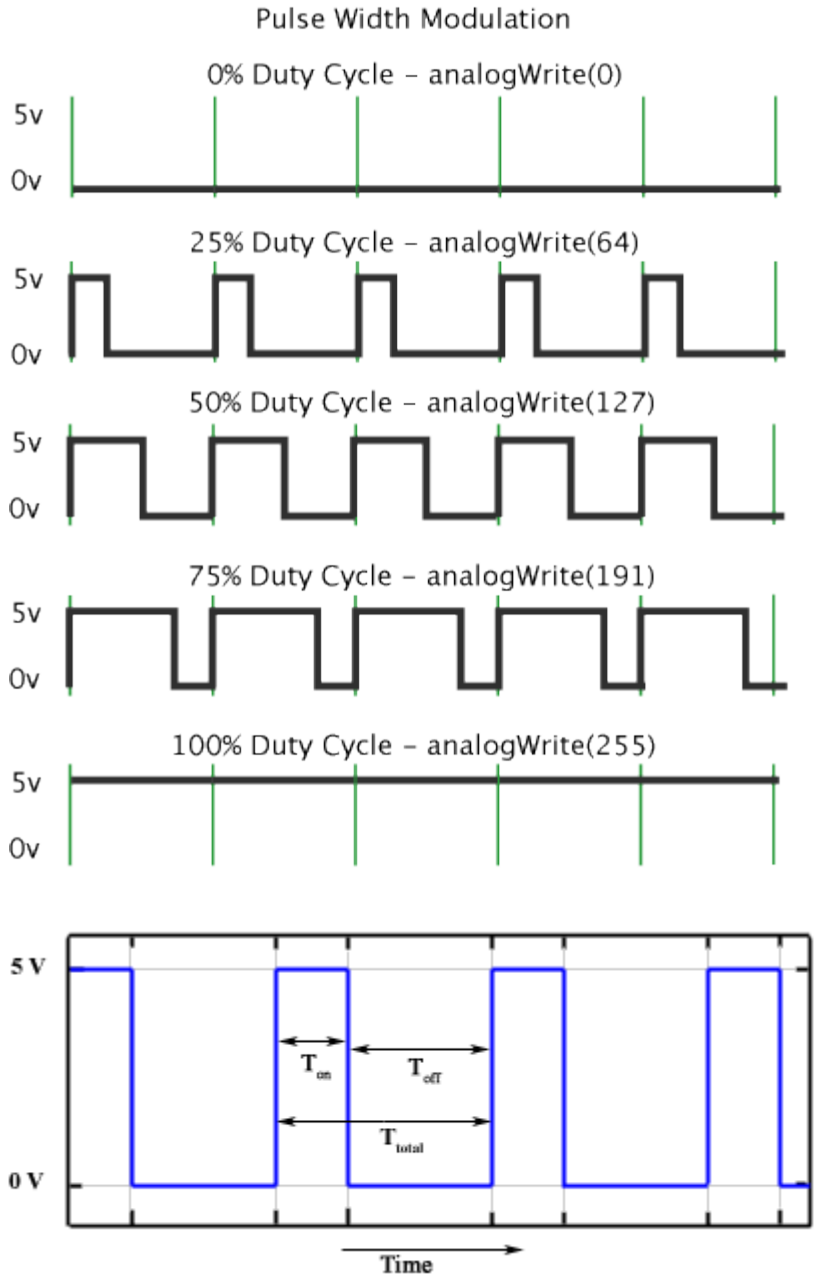
Electrónica Digital
Clase 7

CONTROL LED RGB DESDE ARDUINO (PWM)
DISPLAY DE 7 SEGMENTOS

UNIVERSIDAD EAFIT®

PWM

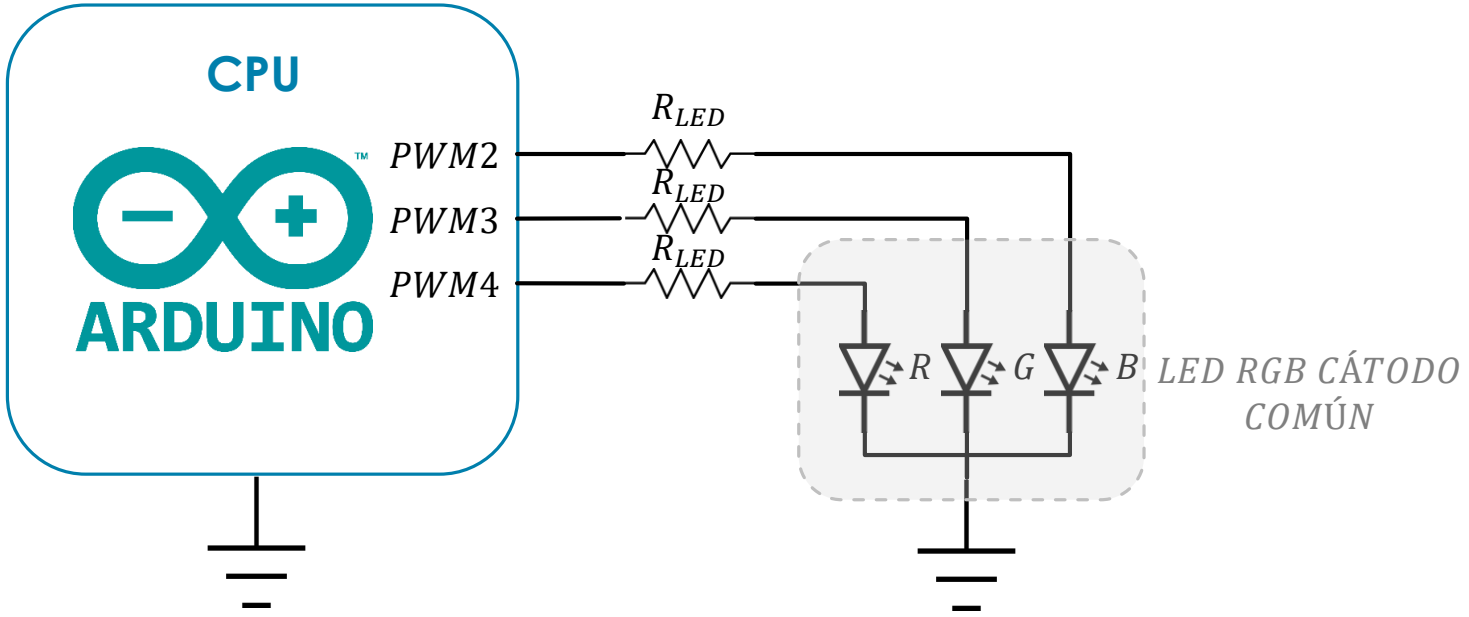
- Llamado PWM por sus siglas en inglés **Pulse Width Modulation**.
- Es una señal **periódica**.
- Su **periodo** siempre es **constante** (y por ende su **frecuencia**, en un **arduino** es de **500 hz**), lo que se **cambia** es el **tiempo de encendido** de la señal cuadrada.
- Al **tiempo de encendido** se le suele llamar **duty cycle** (ciclo de trabajo) puesto que representa la cantidad de energía que se le esta inyectando a un sistema.
- Permite "**graduar**":
  - La **velocidad** de un motor.
  - La **temperatura** de una resistencia.
  - La **intensidad de brillo** de un LED.
- Para arduino: se utiliza la función `analogWrite(PIN,VALOR)`, donde valor puede estar entre 0 y 255.
  - 0 significa 0% de duty cycle.
  - 255 significa 100% de duty cycle.



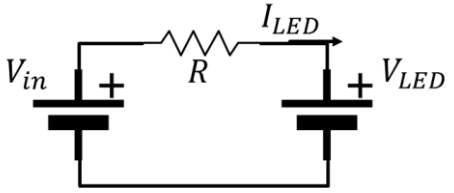
Control LED RGB

- Para **cambiar el color** de un LED RGB, se puede **graduar** la **intensidad** de **rojo**, **verde** y **azul** mediante **PWM**.
- Se debe conectar a 3 pines de PWM con su respectiva interfaz desde un microcontrolador
  - No olvidar que un **LED** siempre **requiere** una **resistencia** para no quemarse.

Circuito típico PWM – LED RGB



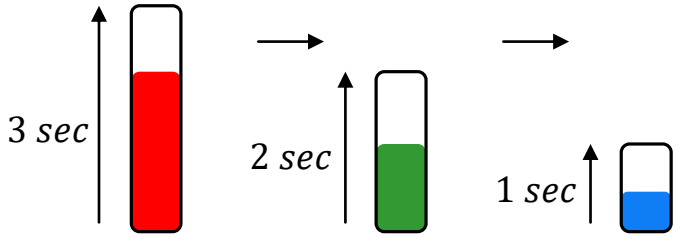
Recordar
Cálculo R para LED



$$R = \frac{V_{in} - V_{LED}}{I_{LED}}$$

Ejemplo 1 – Control LED RGB

- Realice un programa en Arduino que varíe la intensidad de cada segmento de un LED RGB de manera secuencial, desde su valor mínimo hasta su valor máximo. Haga que el tiempo total del recorrido rojo sea de 3 segundos, el de verde de 2 segundos y el azul de 1 segundo. Muestre en el monitor serial el valor de cada segmento.



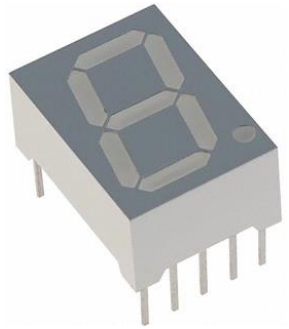

```
//Etiquetado de pines
#define LR 2 //Segmento rojo del RGB en el pin PWM2
#define LG 3 //Segmento rojo del RGB en el pin PWM3
#define LB 4 //Segmento rojo del RGB en el pin PWM4

//Definicion de constantes
const unsigned long TR = 3000; //Constante de tiempo total para el segmento rojo en 3 secs
const unsigned long TG = 2000; //Constante de tiempo total para el segmento verde en 2 secs
const unsigned long TB = 1000; //Constante de tiempo total para el segmento azul en 1 secs

//Configuracion
void setup() {
 //Decir que es entrada y que es salida
 pinMode(LR, OUTPUT); //Segmento rojo como salida
 pinMode(LG, OUTPUT); //Segmento verde como salida
 pinMode(LB, OUTPUT); //Segmento azul como salida

 //Limpieza de salidas fisicas
 digitalWrite(LR, LOW); //Apago segmento rojo
 digitalWrite(LG, LOW); //Apago segmento rojo
 digitalWrite(LB, LOW); //Apago segmento rojo

 //Comunicaciones
 Serial.begin(9600); //Comunicaciones seriales con el PC (Serial0) a 9600 bauds
}
```

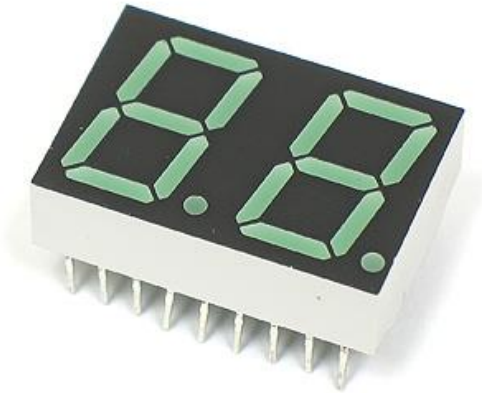
Ejemplo 1 - Continuación

```
//Ejecucion
void loop() {
 for (int i = 0; i <= 255; i++) { //Recorro todo el PWM desde 0% (0) hasta 100% (255)
 analogWrite(LR, i); //Envio el valor del contador al segmento rojo
 Serial.print("R: " + String(i) + " G: 0 B: 0"); //Imprimo en el monitor serial
 delay(TR / 255); //Retardo la señal lo que debe durar cada iteracion para un total de TR secs
 }
 for (int i = 0; i <= 255; i++) { //Recorro todo el PWM desde 0% (0) hasta 100% (255)
 analogWrite(LG, i); //Envio el valor del contador al segmento verde
 Serial.print("R: 0 G: " + String(i) + " B: 0"); //Imprimo en el monitor serial
 delay(TG / 255); //Retardo la señal lo que debe durar cada iteracion para un total de TG secs
 }
 for (int i = 0; i <= 255; i++) { //Recorro todo el PWM desde 0% (0) hasta 100% (255)
 analogWrite(LB, i); //Envio el valor del contador al segmento azul
 Serial.print("R: 0 G: 0 B: " + String(i)); //Imprimo en el monitor serial
 delay(TB / 255); //Retardo la señal lo que debe durar cada iteracion para un total de TB secs
 }
}
```

Display 7 segmentos

- Son un total de 7 LEDs organizados de tal manera que puedan formar **caracteres alfanuméricos**.
- Utilizados para **representar** comúnmente **números**.
- Permiten visualizar en su gran mayoría **contadores** o **temporizadores**.
- Hay de **ánodo común** y de **cátodo común**.
- Es común utilizar un convertidor **BCD a 7 segmentos** para **ahorrar pines digitales**.
  - Un **BCD** recibe en **4 entradas** digitales y las **convierte** en **7 salidas** digitales.





7 segmentos - Funcionamiento

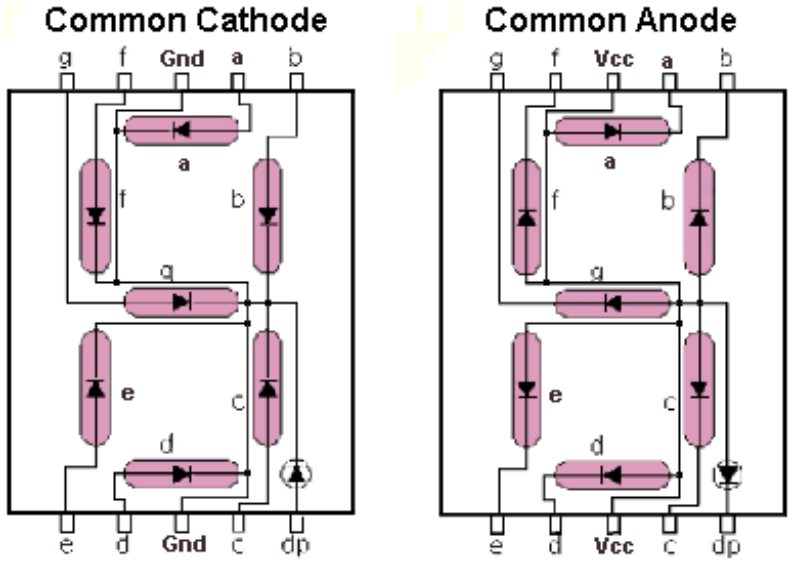
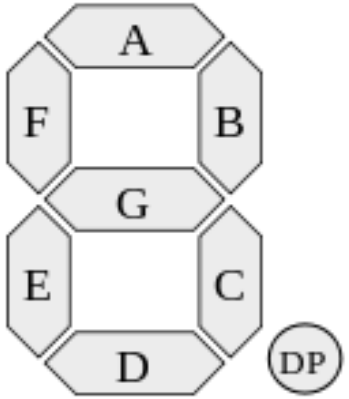


TABLA 7 SEGMENTOS

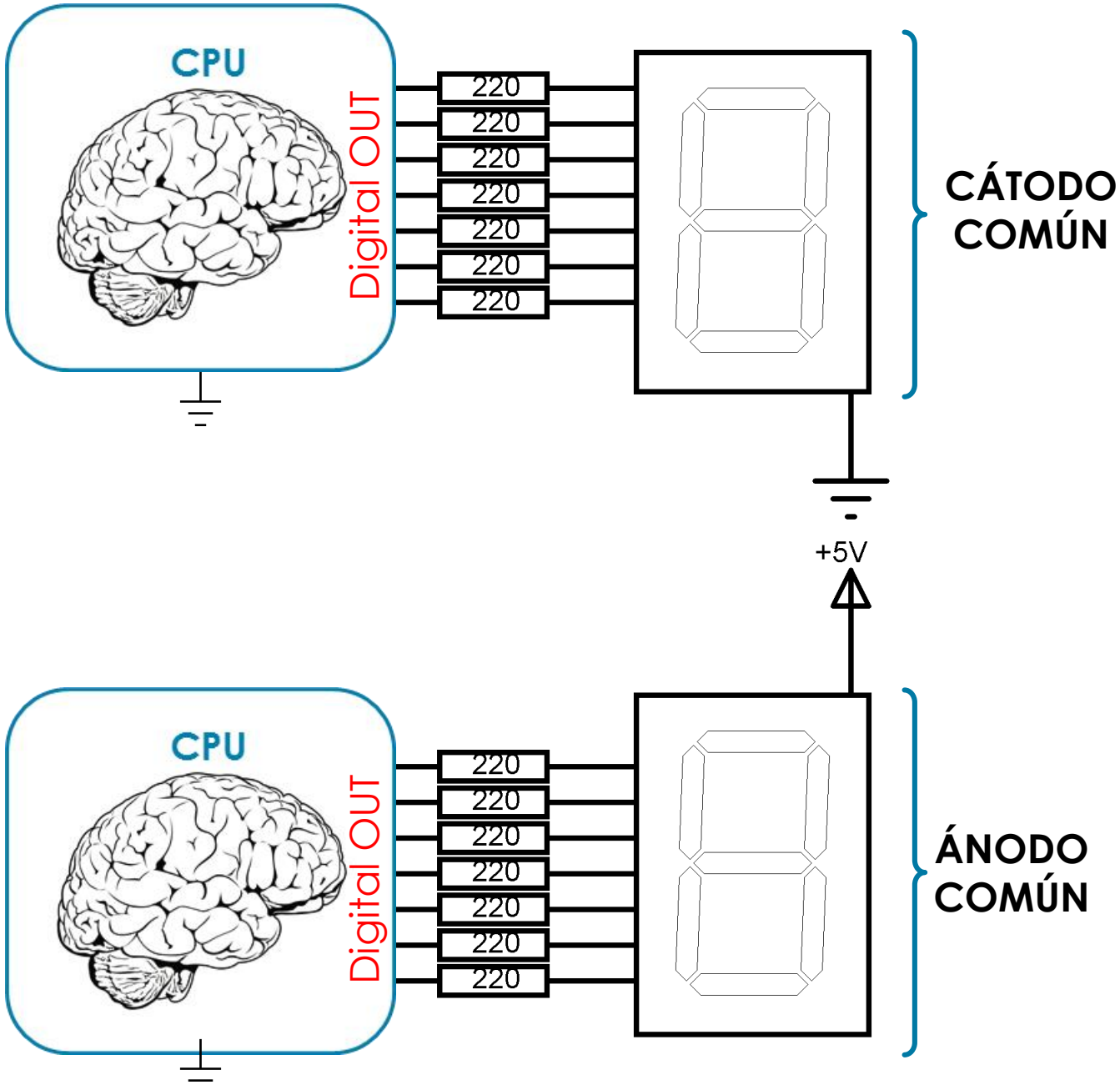
#	A	B	C	D	E	F	G
0	1	1	1	1	1	1	0
1	0	1	1	0	0	0	0
2	1	1	0	1	1	0	1
3	1	1	1	1	0	0	1
4	0	1	1	0	0	1	1
5	1	0	1	1	0	1	1
6	1	0	0	1	1	1	1
7	1	1	1	0	0	0	0
8	1	1	1	1	1	1	1
9	1	1	1	0	0	1	1



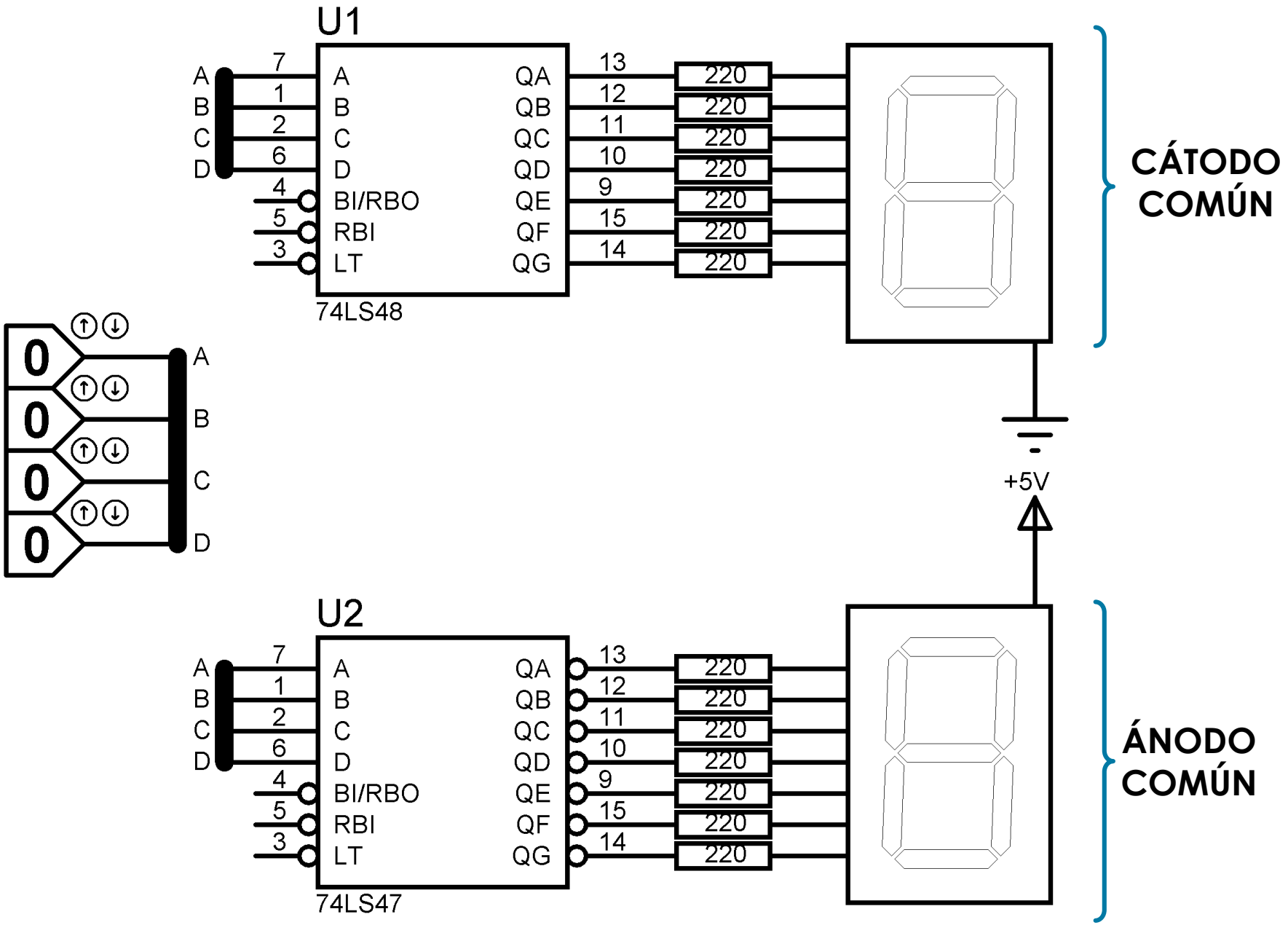
BCD A 7 SEGMENTOS

D	C	B	A	#
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

7 segmentos – Conexión directa

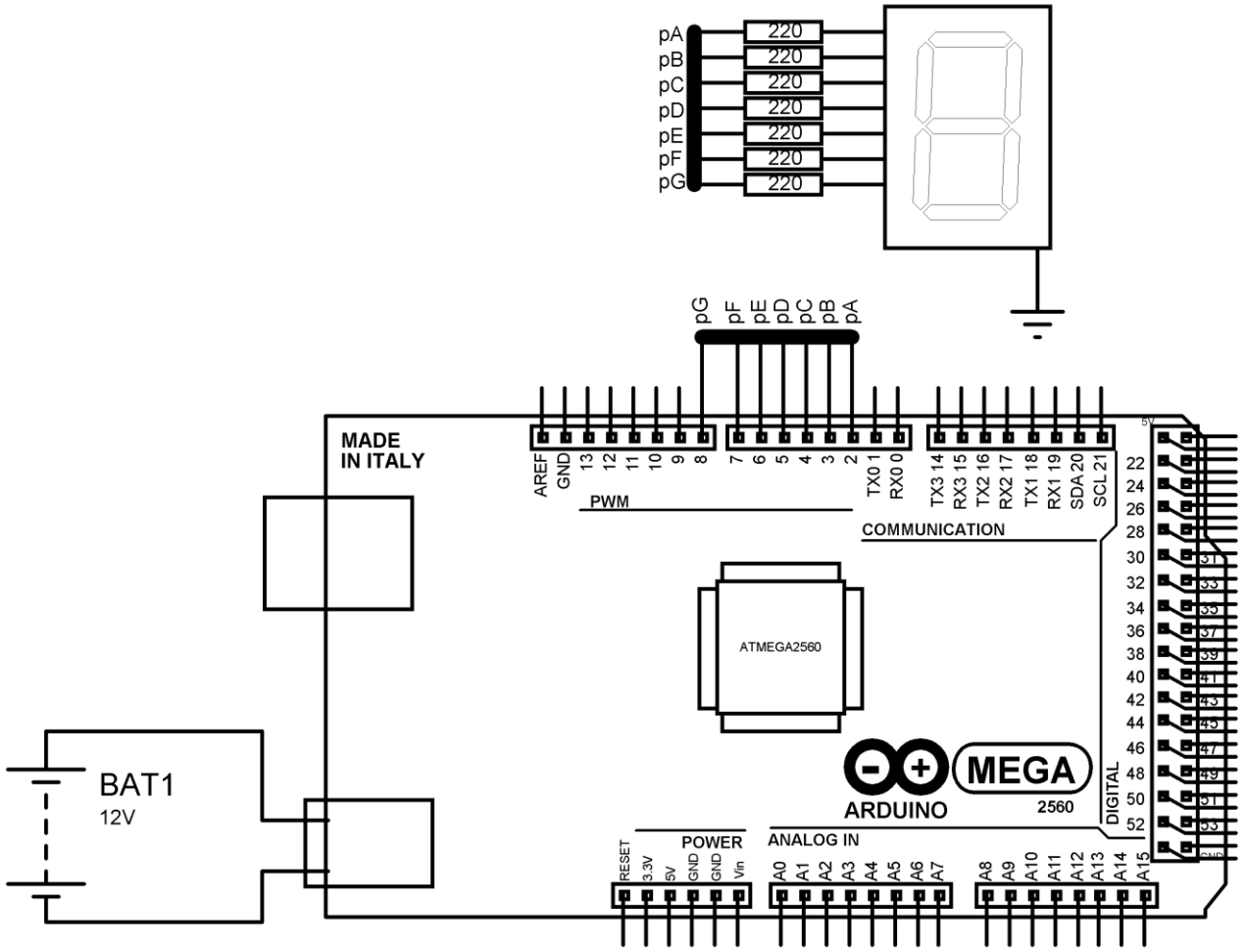


7 segmentos – Conexión con BCD



7 segmentos - Ejemplo

- Realice un programa en un Arduino que cuente de 0 a 9 segundos, los muestre en un display de 7 segmentos de cátodo común y se reinicie una vez finalice el conteo.

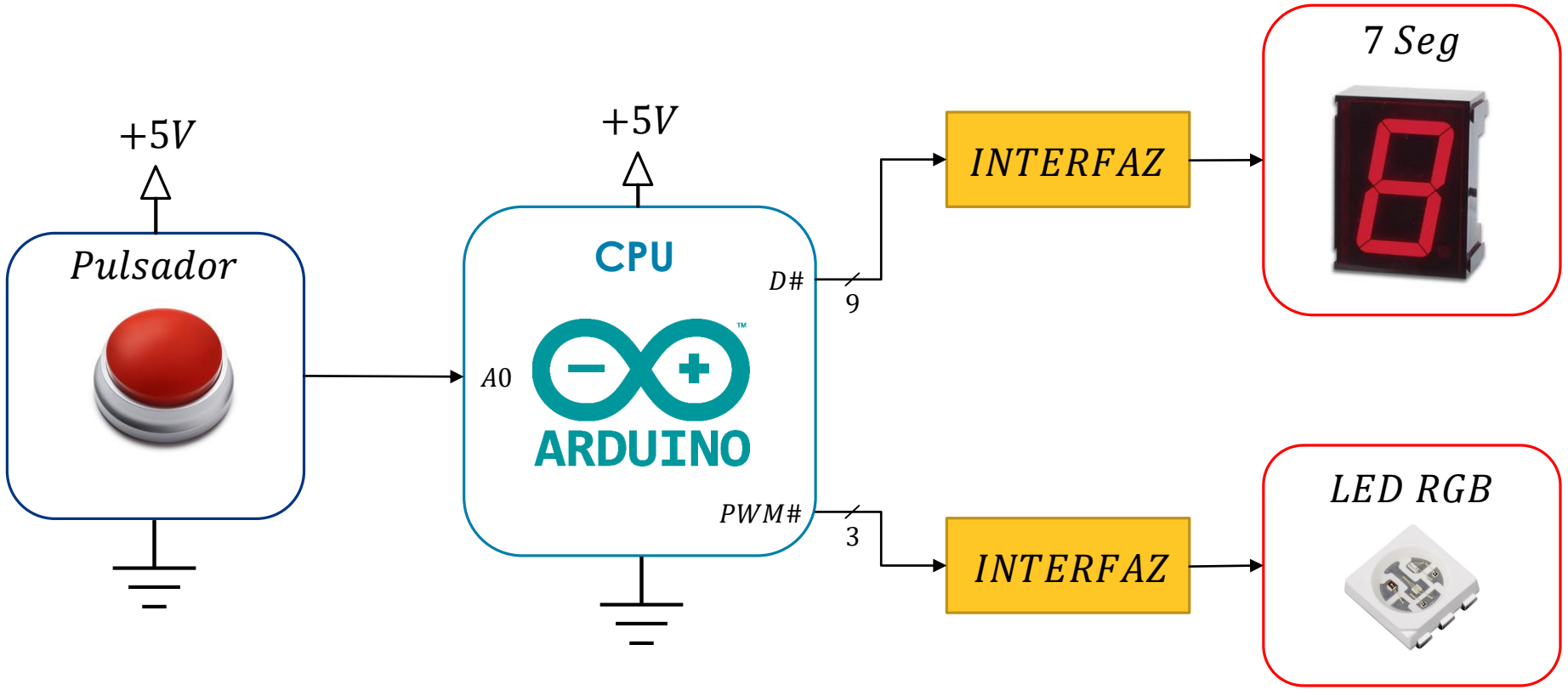


7 segmentos – Ejemplo

```
//Definicion de pines de I/O
#define pA 2
#define pB 3
#define pC 4
#define pD 5
#define pE 6
#define pF 7
#define pG 8

//Definicion de variables
unsigned int cont = 0;

void int2bcd(unsigned int num)  //Función que toma un numero entero y lo convierte a BCD
{
 switch (num)
 {
   case 0:
    digitalWrite(pA, HIGH);
    digitalWrite(pB, HIGH);
    digitalWrite(pC, HIGH);
    digitalWrite(pD, HIGH);
    digitalWrite(pE, HIGH);
    digitalWrite(pF, HIGH);
    digitalWrite(pG, LOW);
    break;

   case 1:
    digitalWrite(pA, LOW);
    digitalWrite(pB, HIGH);
    digitalWrite(pC, HIGH);
    digitalWrite(pD, LOW);
    digitalWrite(pE, LOW);
    digitalWrite(pF, LOW);
    digitalWrite(pG, LOW);
    break;

   case 2:
    digitalWrite(pA, HIGH);
    digitalWrite(pB, HIGH);
    digitalWrite(pC, LOW);
    digitalWrite(pD, HIGH);
    digitalWrite(pE, HIGH);
    digitalWrite(pF, LOW);
    digitalWrite(pG, HIGH);
    break;

   case 3:
    digitalWrite(pA, HIGH);
    digitalWrite(pB, HIGH);
    digitalWrite(pC, HIGH);
    digitalWrite(pD, HIGH);
    digitalWrite(pE, LOW);
    digitalWrite(pF, LOW);
    digitalWrite(pG, HIGH);
    break;

   case 4:
    digitalWrite(pA, LOW);
    digitalWrite(pB, HIGH);
    digitalWrite(pC, HIGH);
    digitalWrite(pD, LOW);
    digitalWrite(pE, LOW);
    digitalWrite(pF, HIGH);
    digitalWrite(pG, HIGH);
    break;

   case 5:
    digitalWrite(pA, HIGH);
    digitalWrite(pB, LOW);
    digitalWrite(pC, HIGH);
    digitalWrite(pD, HIGH);
    digitalWrite(pE, LOW);
    digitalWrite(pF, HIGH);
    digitalWrite(pG, HIGH);
    break;

   case 6:
    digitalWrite(pA, HIGH);
    digitalWrite(pB, LOW);
    digitalWrite(pC, LOW);
    digitalWrite(pD, HIGH);
    digitalWrite(pE, HIGH);
    digitalWrite(pF, HIGH);
    digitalWrite(pG, HIGH);
    break;

   case 7:
    digitalWrite(pA, HIGH);
    digitalWrite(pB, HIGH);
    digitalWrite(pC, HIGH);
    digitalWrite(pD, LOW);
    digitalWrite(pE, LOW);
    digitalWrite(pF, LOW);
    digitalWrite(pG, LOW);
    break;

   case 8:
    digitalWrite(pA, HIGH);
    digitalWrite(pB, HIGH);
    digitalWrite(pC, HIGH);
    digitalWrite(pD, HIGH);
    digitalWrite(pE, HIGH);
    digitalWrite(pF, HIGH);
    digitalWrite(pG, HIGH);
    break;

   case 9:
    digitalWrite(pA, HIGH);
    digitalWrite(pB, HIGH);
    digitalWrite(pC, HIGH);
    digitalWrite(pD, LOW);
    digitalWrite(pE, LOW);
    digitalWrite(pF, HIGH);
    digitalWrite(pG, HIGH);
    break;
 }
}

void setup()
{
  pinMode(pA, OUTPUT);
  pinMode(pB, OUTPUT);
  pinMode(pC, OUTPUT);
  pinMode(pD, OUTPUT);
  pinMode(pE, OUTPUT);
  pinMode(pF, OUTPUT);
  pinMode(pG, OUTPUT);
  digitalWrite(pA, LOW);
  digitalWrite(pB, LOW);
  digitalWrite(pC, LOW);
  digitalWrite(pD, LOW);
  digitalWrite(pE, LOW);
  digitalWrite(pF, LOW);
  digitalWrite(pG, LOW);
  Serial.begin(9600);
}

void loop()
{
  int2bcd(cont);  //Convierto el contador a BCD y lo muestro en el display de 7 segmentos
  cont = cont + 1;  //Incremento contador
  if (cont > 9) //Si el contador es mayor a 9, reinicie
  {
    cont = 0;
  }
  Serial.println(cont);
  delay(1000);  //Retardo de 1 segundo
}
```

Seguimiento

- Implemente un contador de 0 a 9 que incremente cada que suceda un pulso de un suiche (utilizar un while para evitar que cuente si no se ha desactivado el suiche y delay antirebote). Este contador se deberá visualizar en un display de 7 segmentos y dependiendo del nivel del contador, muestre diferentes colores (a su libre elección, por lo menos 4 colores diferentes).



Fin

MUCHAS GRACIAS