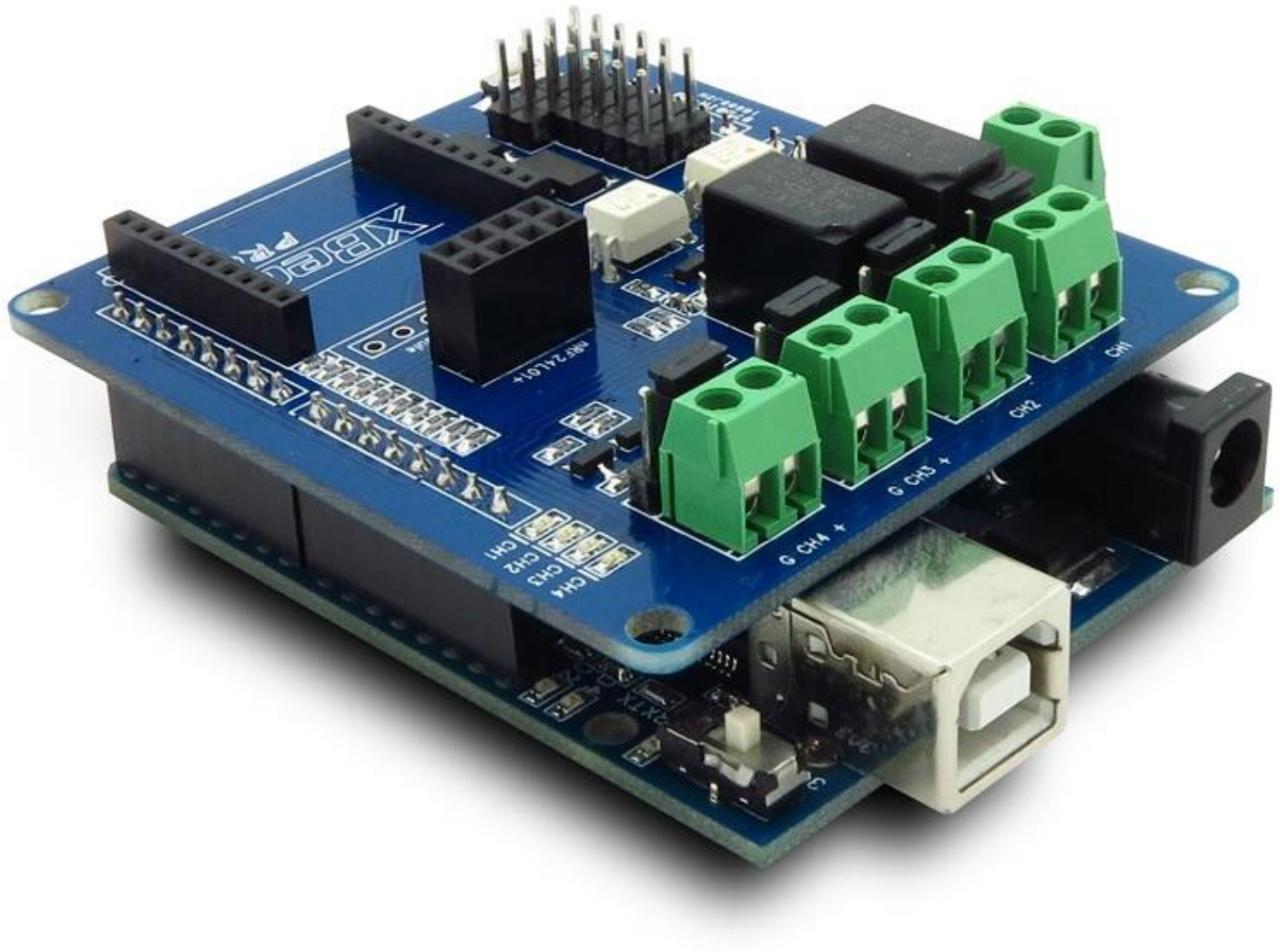
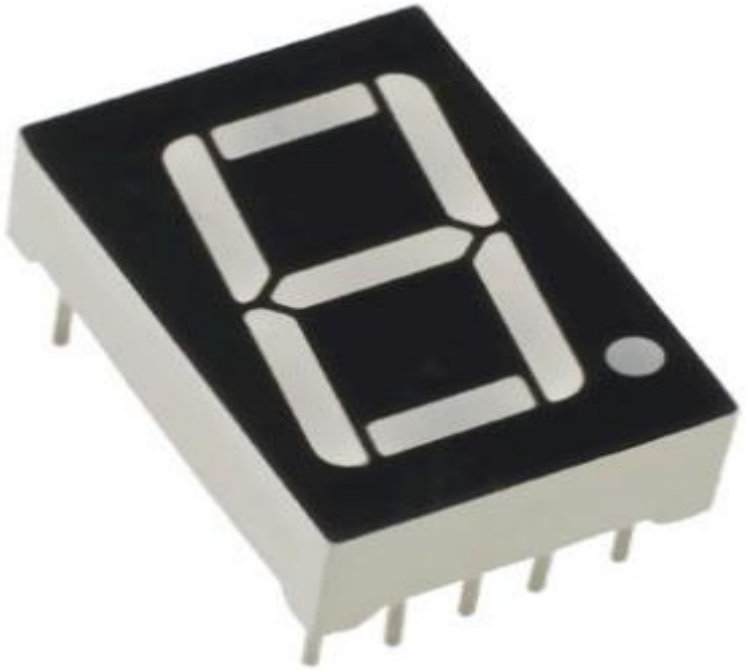


PUERTOS DIGITALES

SISTEMAS EMBEBIDOS



Display
7 segmentos

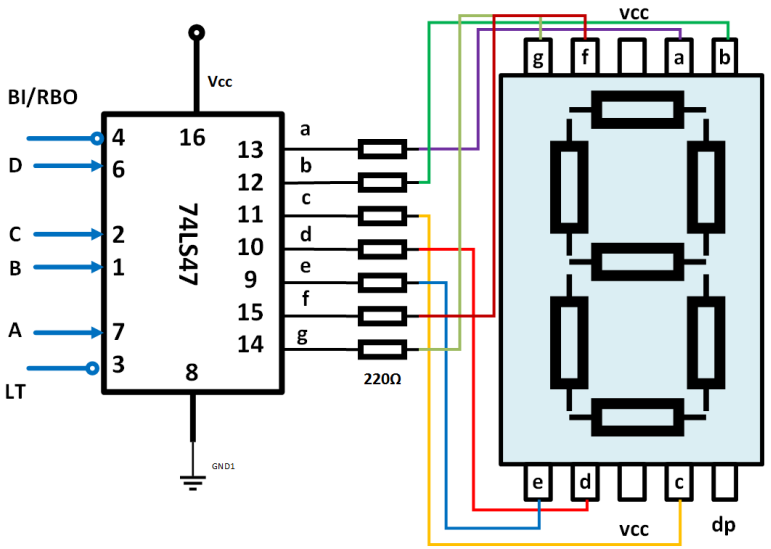
Un display es la unión de 7 leds en un solo dispositivo. Su construcción está desarrollada para la presentación de números decimales y hexadecimales

Existen dos tipos principales para los display 7 segmentos. Esta diferencia depende principalmente del arreglo como están conectados los leds que forman a cada segmento.

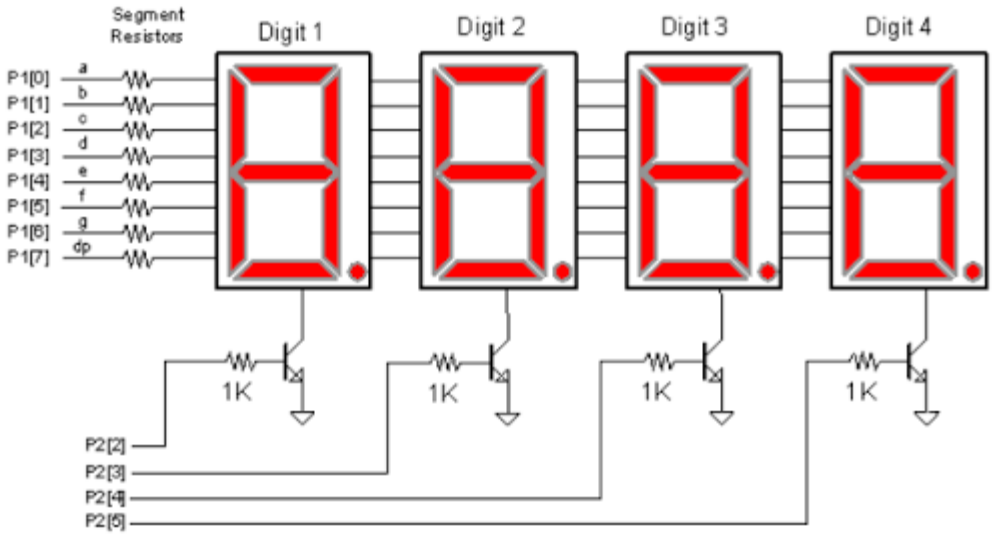
Decodificadores

Tabla de conversión

circuito



	Código 7 Segmentos Anodo Común								
Numero		g	f	e	d	c	b	a	Hex
0	0	1	0	0	0	0	0	0	40
1	0	1	1	1	1	0	0	1	79
2	0	0	1	0	0	1	0	0	24
3	0	0	1	1	0	0	0	0	30
4	0	0	0	1	1	0	0	1	19
5	0	0	0	1	0	0	1	0	12
6	0	0	0	0	0	0	1	0	02
7	0	1	1	1	1	0	0	0	78
8	0	0	0	0	0	0	0	0	00
9	0	0	0	1	1	0	0	0	18



Multiplexador 7 segmentos

El uso de display tiene un consumo elevado de pines digitales. Algunas placas de Arduino no los cuentan. Por ello, se busca multiplexar display para optimizar su uso.

Se usan transistores para mantener la corriente desde los propios piden del Arduino. De esta manera son polarizados los display solo en el momento de visualizar la información correspondiente.

pinMode()
digitalWrite()
delay()

pinMode()

Al hacer una llama a la rutina pinMode se le deja saber a Arduino si una terminal será utilizada como entrada o salida.

Sintaxis:

pinMode(pin, MODO)

MODO: INPUT o OUTPUT.

digitalWrite()

Una vez que se a especificado que una terminal de Arduino será utilizada como salida, se usa esta función para escribir un valor digital alto o bajo a dicha terminal.

Sintaxis:

digitalWrite(pin, ESTADO)

ESTADO: HIGH o LOW.

delay()

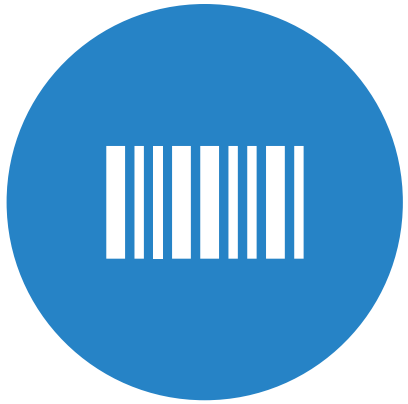
Rutina de retardo universal, detiene el flujo de ejecución del sketch actual.

Sintaxis:

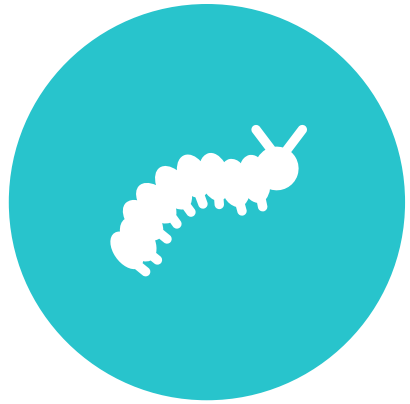
delay(ms)

ms: numero de milisegundos a esperar. (1000 ms = 1).

PROGRAMACIÓN



CÓDIGO 5: MANEJO DE DISPLAY 7 SEGMENTOS



CÓDIGO 6: MULTIPLEZADOR DE DISPLAY

Pantalla
LCD 2x16

Galileo proporciona también una biblioteca para el control de pantallas LCD: **LiquidCrystal.h**

Esta biblioteca se basa en el controlador HD44780 de Hitachi, que se encuentra en la mayoría de las pantallas LCD.

La biblioteca trabaja tanto en el modo de 4 u 8 bits de datos, además de las señales de control RS y Enable y opcionalmente la señal de control RW.

Biblioteca LiquidCrystal.h

En esta sección se utilizan las funciones de la biblioteca **LiquidCrystal.h**:

- **LiquidCrystal ( )** — Inicializa la interfaz de la pantalla LCD
- **begin (**cols, rows**)**
- **print (**data**)** — Imprime texto en la LCD
- **setCursor (**col, row**)** — Posiciona el cursor en la LCD

.

Para ver todas las funciones: http://arduino.cc/en/Reference/LiquidCrystal

LiquidCrystal ( )

LiquidCrystal ( )

Esta función crea una variable
del tipo LiquidCrystal.

.**Sintaxis:**

LiquidCristal **name**(RS, E, D4, D5, D6, D7)

LiquidCristal **name**(RS, RW, E, D4, D5, D6, D7)

LiquidCristal **name**(RS, E, D0, D1, D2, D3, D4, D5, D6, D7)

LiquidCristal **name**(RS, RW, E, D0, D1, D2, D3, D4, D5, D6, D7)

Como se puede apreciar, la LCD puede ser controlada mediante 4 u 8 líneas de datos. Para controlarla con 4 líneas, simplemente omitir escribir las señales D0, D1, D2 y D3. El uso del pin RW es opcional; en caso de no utilizarlo se puede conectar directamente a tierra.

Configuración de pines del LCD 2x16

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
GND	VCC	VEE	RS	R/W	E	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7	LED+	LED-

Código
HelloWorld.ino

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  lcd.begin(16, 2);
  lcd.print("hello, world!");
}

void loop() {
  lcd.setCursor(0, 1);
  lcd.print(millis()/1000);
}
```

Utiliza 4 bits para configurar la LCD y no se usa el bit RW.

Para el shield que usarán, la configuración es distinta:

```
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
```

Inicializa la pantalla de 16 caracteres por 2 líneas

Imprime una vez la palabra **hello, world!** en la primer línea de la pantalla LCd

Posiciona el cursor al inicio de la segunda línea

Imprime en la pantalla una cuenta en segundos

PROGRAMACIÓN

CÓDIGO 7: LCD 16X2

CÓDIGO 8: PUERTOS ENTRADA-SALIDA CON LCD 16X2