

DISPLAYS LCD

(Versión 03-10-18)


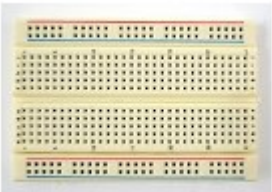



Usando displays de texto LCD 16x2 con Arduino

(Podemos encontrar infinidad de ejemplos en Internet, todos son similares, aunque puede diferir el conexionado, sin tienen en común el uso de una biblioteca que debemos incluir en nuestro Sketch, acá adoptaremos uno encontrado en la Web. Al final indicaremos las fuentes).

OBJETIVOS

- Vamos a montar un display LCD de 16x2.
- Veremos cómo enviar información
- Definiremos algunos caracteres especiales

MATERIAL REQUERIDO

	Arduino Uno o similar
	Un protoboard
	Cables para unir Arduino y protoboard
	Un Potenciómetro.
	Un display LCD, de 16x2 o 16x4.

LOS DISPLAYS LCD

Los displays LEDs de 7 segmentos, son baratos y prácticos, pero tienen el inconveniente de que no pueden mostrar mensajes de texto, sino solo números. Para lograr eso se comercializan los displays LCD. Son fáciles de encontrar en diversos formatos:

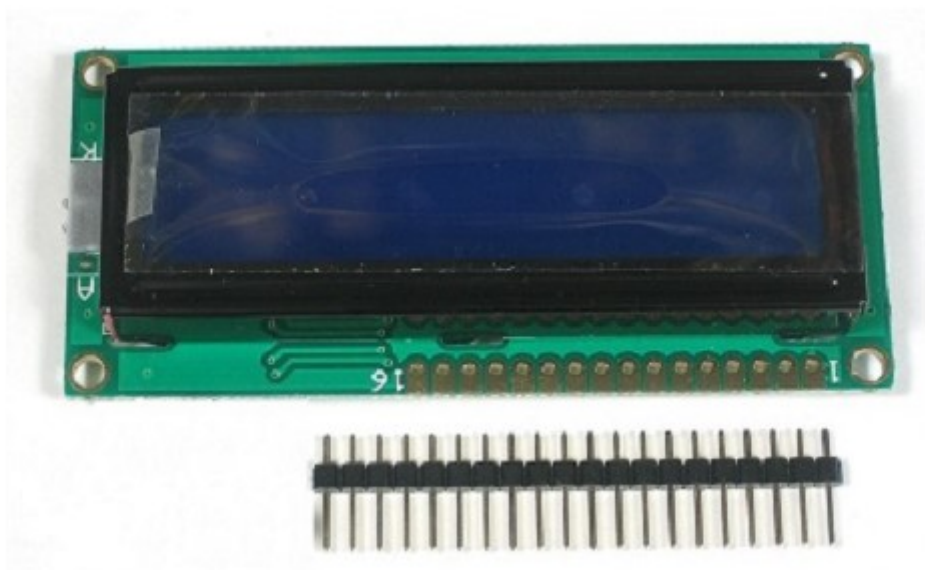
Ej:

16x2 (16 caracteres x 2 líneas) o LCD 16x4 (16 caracteres x4 líneas).

En esta sesión veremos cómo conectarlos al Arduino y cómo usarlos para mostrar mensajes.

LCD viene del inglés **Liquid Crystal Display**, o sea Pantalla de cristal líquido. Están disponibles en varios tamaños y configuraciones. Son de bajo consumo. Muy prácticos si deseamos mostrar solo texto (y algunos caracteres especiales).

Los LCD pueden venir con los pines soldados o listos para soldar, si bien podríamos conectarlo con cables soldados, lo más útil es hacerlo con pines.



De ese modo podemos incluso introducirlo en nuestro protoboard.

Detalles técnicos de los LCD (puede leerla luego)

La pantalla LCD conectada a Arduino nos permitirá mostrar texto de una manera sencilla. La plataforma Arduino cuenta con bibliotecas como "LiquidCrystal" (escrita por Limor Fried) que permite controlar displays LCDs compatibles con el driver Hitachi HD44780 (muy común en el mercado).

La pantalla LCD, suele tener unas inscripciones por la parte delantera o trasera en las que indica qué es cada pin de conexión.



Se puede apreciar que en el caso de la LCM1602C la numeración de los pines de conexión comienza desde derecha a izquierda, ¡Ojo, visto desde atrás! (16, 15,

14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1). Si seguimos observando vemos que nos indica que el pin 1 corresponde a GND, el dos a la entrada de tensión Vdd, el 3 a Vo, el 4 a RS, el 5 a RW, 6 a E, 7-14 a Dbx, 15 a BL1 y 16 a BL2.

RS (Register Select): es el pin que controla la memoria del LCD, e indica que registro de la memoria será el que se lee o escribe. Desde esta memoria se obtienen los datos para mostrar en pantalla, pero también se obtienen instrucciones que el controlador del LCD necesita para actuar.

RW (Read/Write): es el pin de lectura y escritura, que dirá si se escribe en memoria o si se lee en cada momento.

E (Enable): pin que habilita los registros.

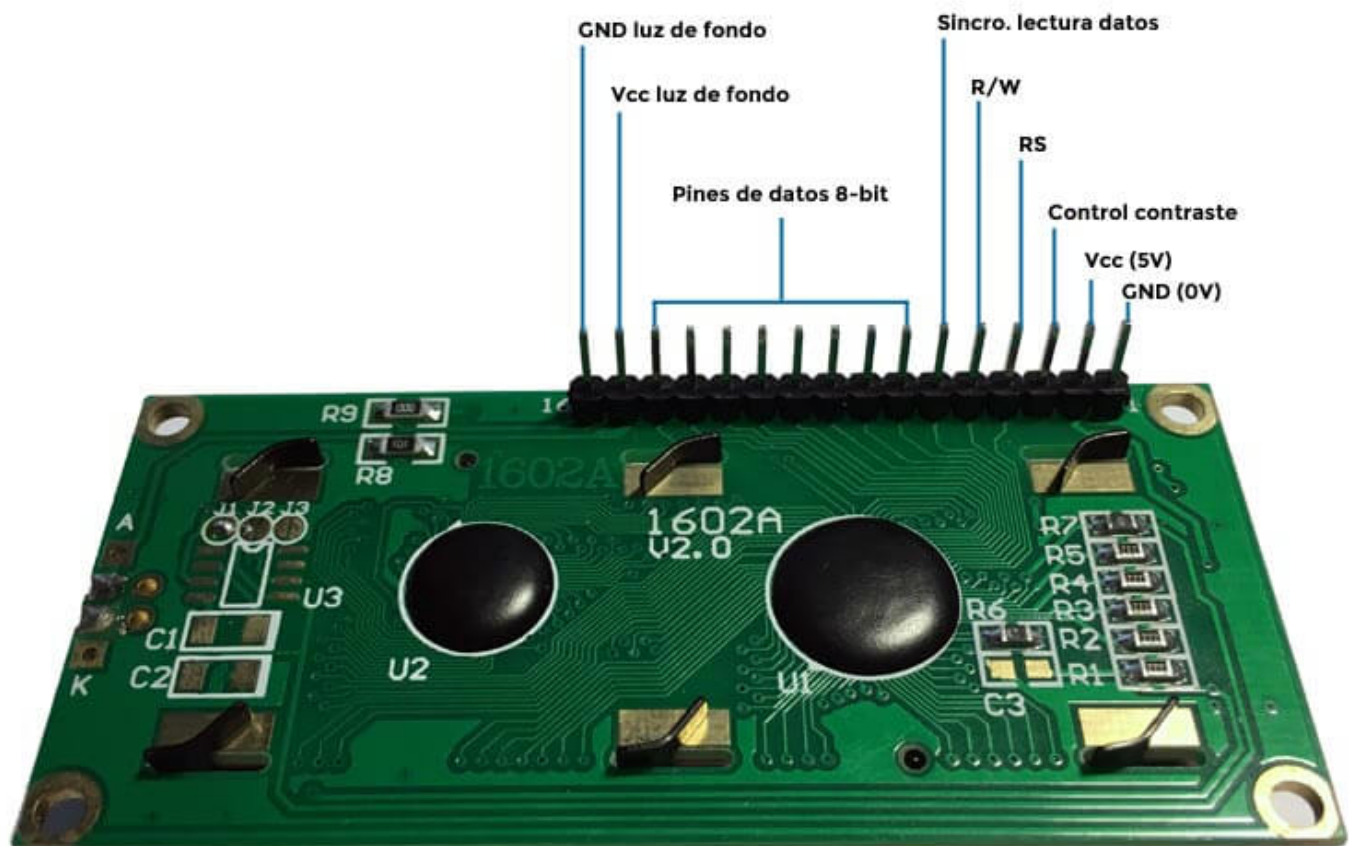
DB0-DB7: son los pines de datos, de los que se sacan los bits que llegan al registro. En el caso de la pantalla que tengo los designa como DB0-DB7 y corresponden a la numeración 7-14, pero esto puede variar en función del fabricante o modelo de la LCD. Por ejemplo, en algunas he visto que lo designan como D00-D07.

Vo: es el pin de contraste del display, con el podremos modificar el contraste de la pantalla.

Vdd: es el pin de alimentación, por el que entra la tensión que hará funcionar al LCD. Normalmente es de +5v.

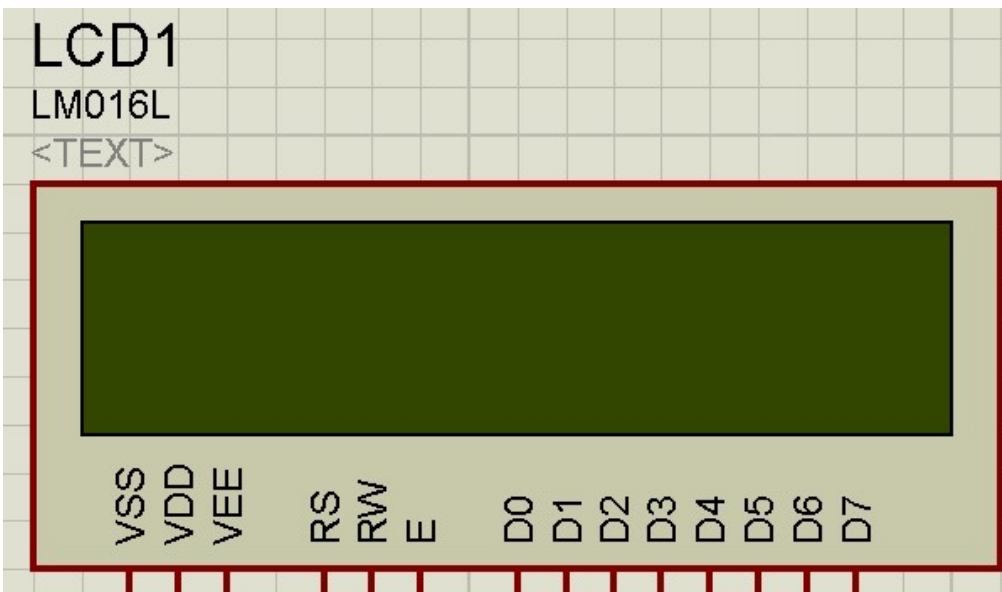
GND: complementa al anterior pin, siendo el otro pin de alimentación que se conectará a tierra.

BL1 y BL2: son los pines de retro iluminación. Controlan la luminosidad del LCD. En algunas pantallas puede aparecer con los símbolos Bklt+ y Bklt-. Las siglas provienen de "Back Light".

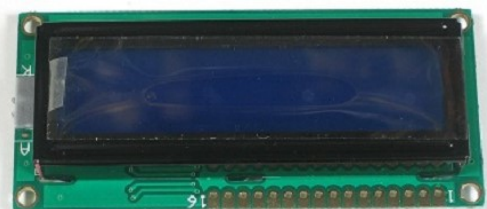




Comparemos con el de Proteus:

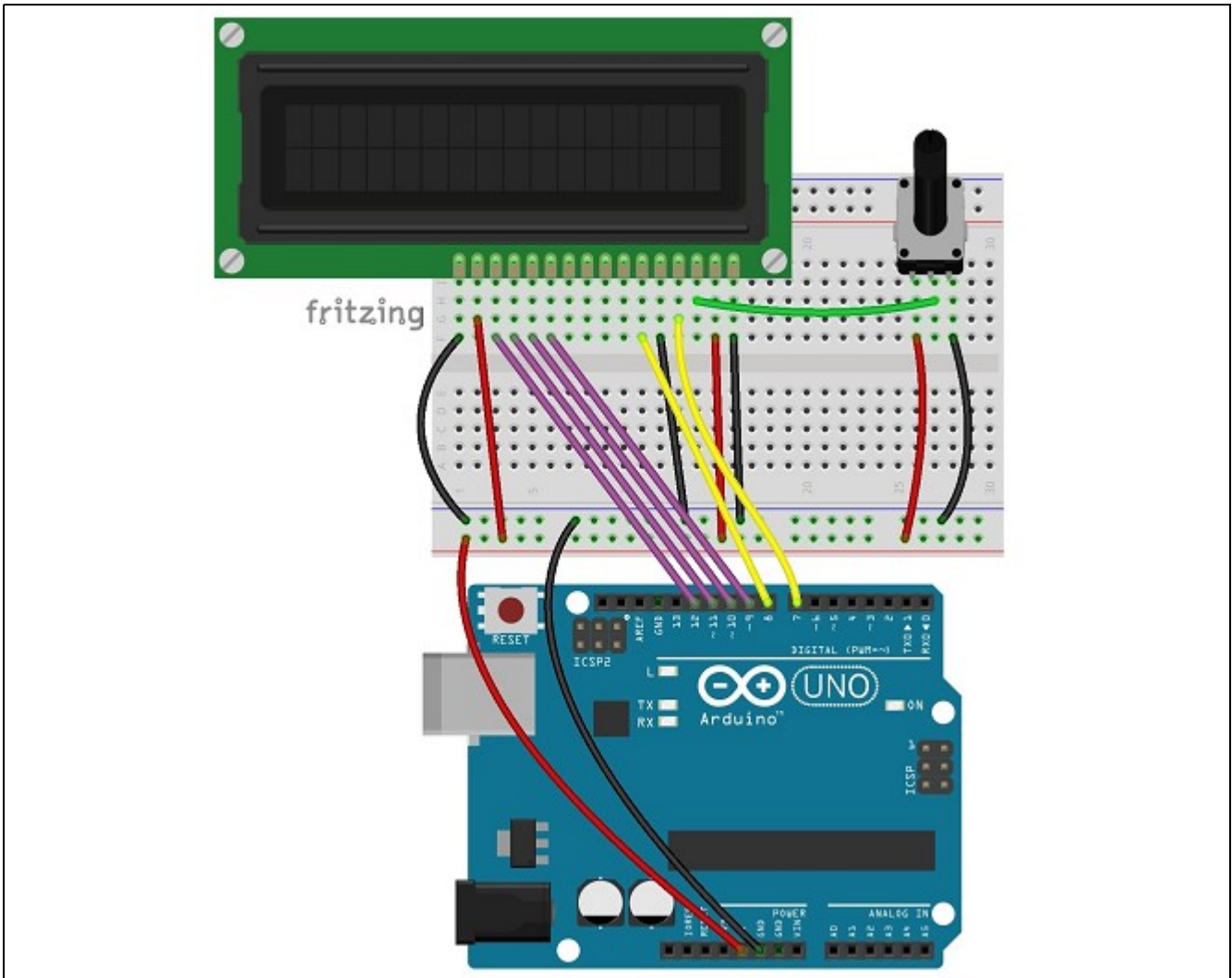
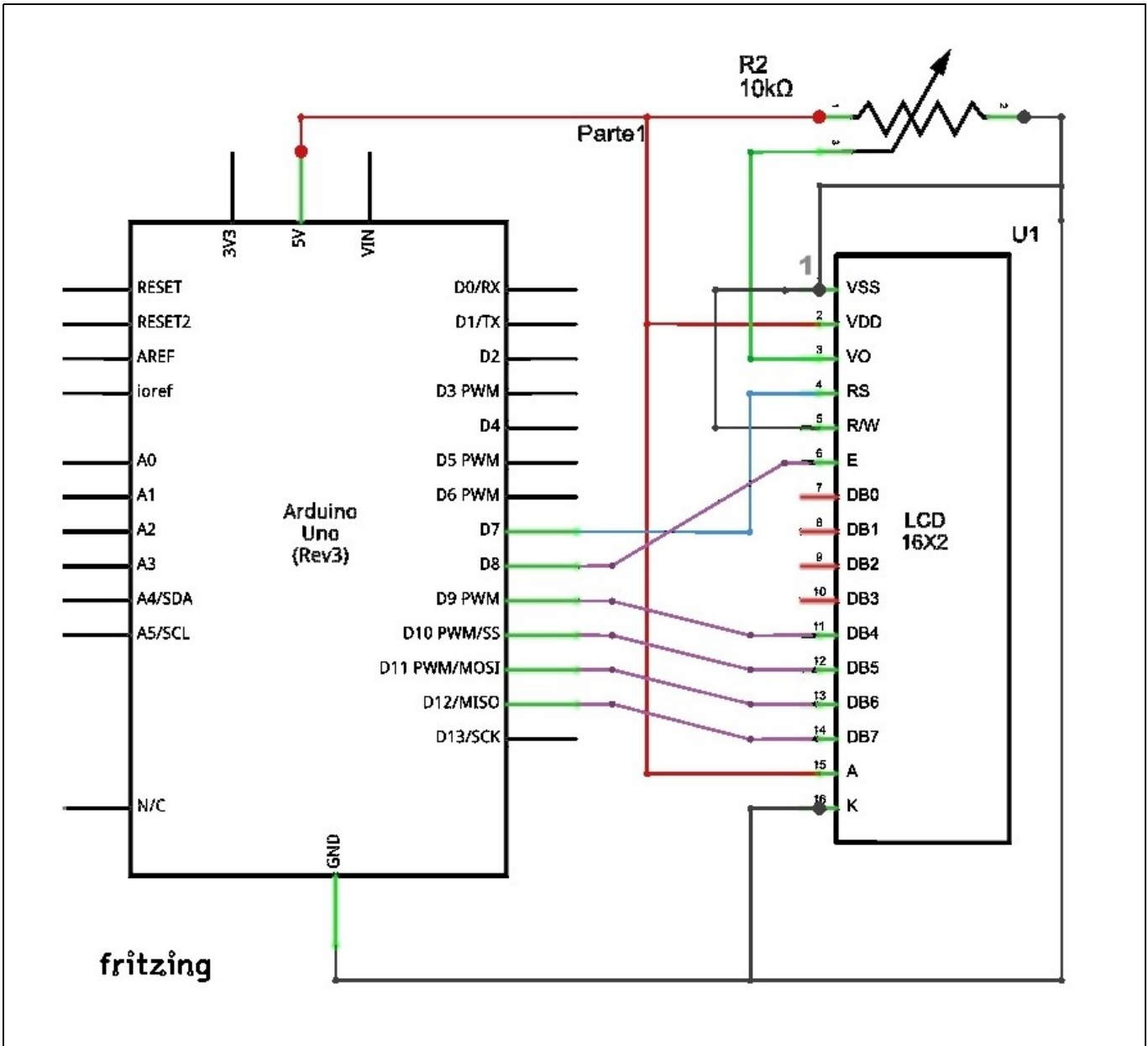


Ahora veremos como conectarlo al Arduino Uno.



Plano de conexionado LCD 16x2 con la placa Arduino

Es el conexionado que usaremos aquí.



NOTA: Es posible enviar datos al LCD por los 8 pines DB0-DB7 o enviarlos solo con 4 pines.

Vamos a conectar los pines de datos y control. Sin entrar en muchos detalles, no vamos a usar todos los pines disponibles, porque no los necesitamos.

Solo usaremos dos pines de control, **RS** y **EN** y los 4 pines de datos **D7, D6, D5, y D4** .

Vamos con las conexiones de control del display:

RW, LCD pin 5	GND
RS, LCD pin 4	Arduino pin 7
EN, LCD pin 6	Arduino pin 8

Y ya solo nos quedan los 4 cables de datos.

DB7, LCD pin 14	Arduino pin 12
DB6, LCD pin 13	Arduino pin 11
DB5, LCD pin 12	Arduino pin 10
DB4, LCD pin 11	Arduino pin 9

EL PROGRAMA DE CONTROL (Lo que va en Arduino)

Vamos a usar una librería de control del panel LCD, que viene incluida en nuestro Arduino.

Buscar en IDE:

\\Programa\Importar Libreria\LiquidCrystal

Lo primero es que al importar la librería nos ha escrito esto:

#include <LiquidCrystal.h>

Despues, hay que inicializar la librería. Creamos una instancia llamada lcd, de la clase LiquidCrystal y le pasamos como parámetros los pines que hemos usado:

LiquidCrystal lcd(7, 8, 9, 10, 11, 12); // (RS, EN, D4, D5, D6, D7)

Se debe tener cuidado porque los pines que hemos usado, no corresponden a otros ejemplos de Arduino, podemos usarlos, pero asegurarnos de cambiar la línea de definición de los pines, o no funcionara el LCD.

Ejemplo de sketch

```
//Biblioteca necesaria para LCDs
#include <LiquidCrystal.h>

//Iniciamos los pines a utilizar
LiquidCrystal lcd(7, 8, 9, 10, 11, 12); //( RS, EN, D4, D5, D6, D7)

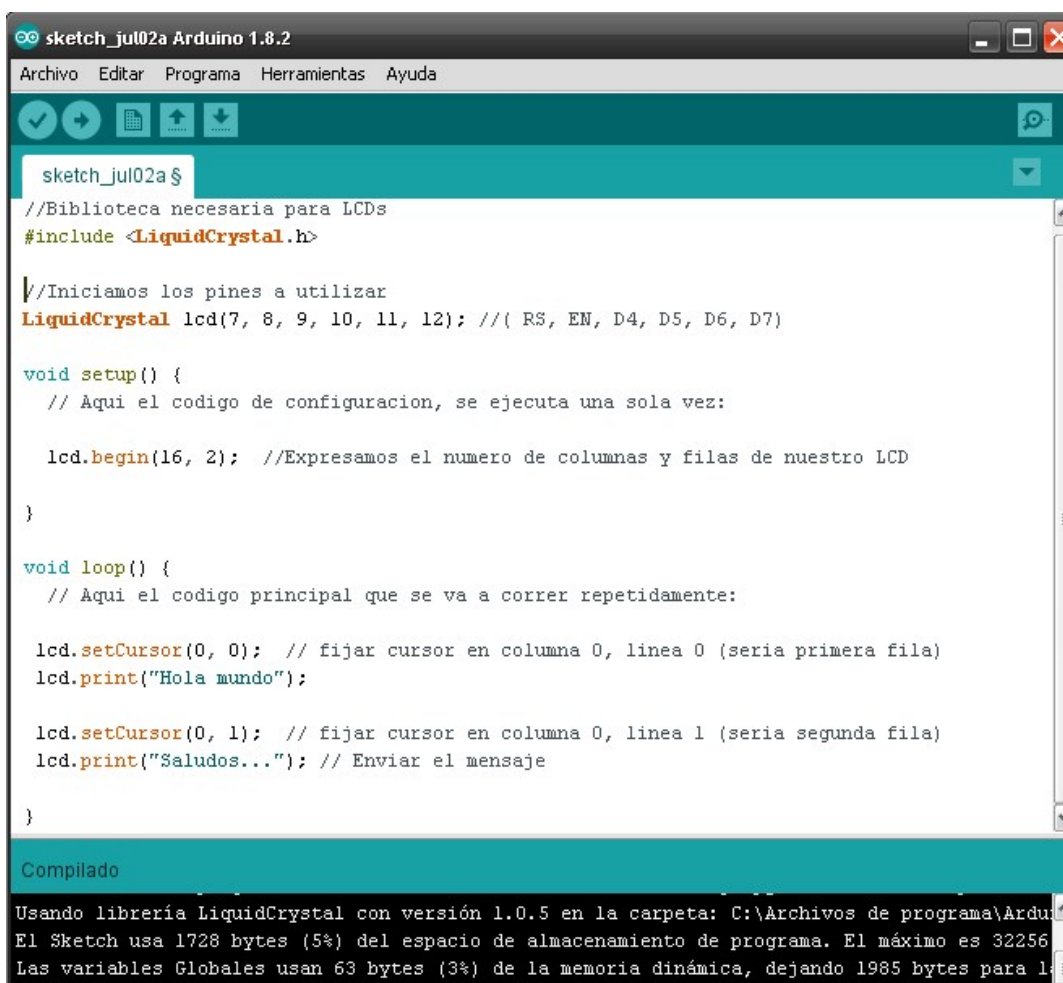
void setup() {
  // Aqui el codigo de configuracion, se ejecuta una sola vez:

  lcd.begin(16, 2); //Expresamos el numero de columnas y filas de nuestro LCD
}

void loop() {
  // Aqui el codigo principal que se va a correr repetidamente:

  lcd.setCursor(0, 0); // fijar cursor en columna 0, linea 0 (seria primera fila)
  lcd.print("Hola mundo");

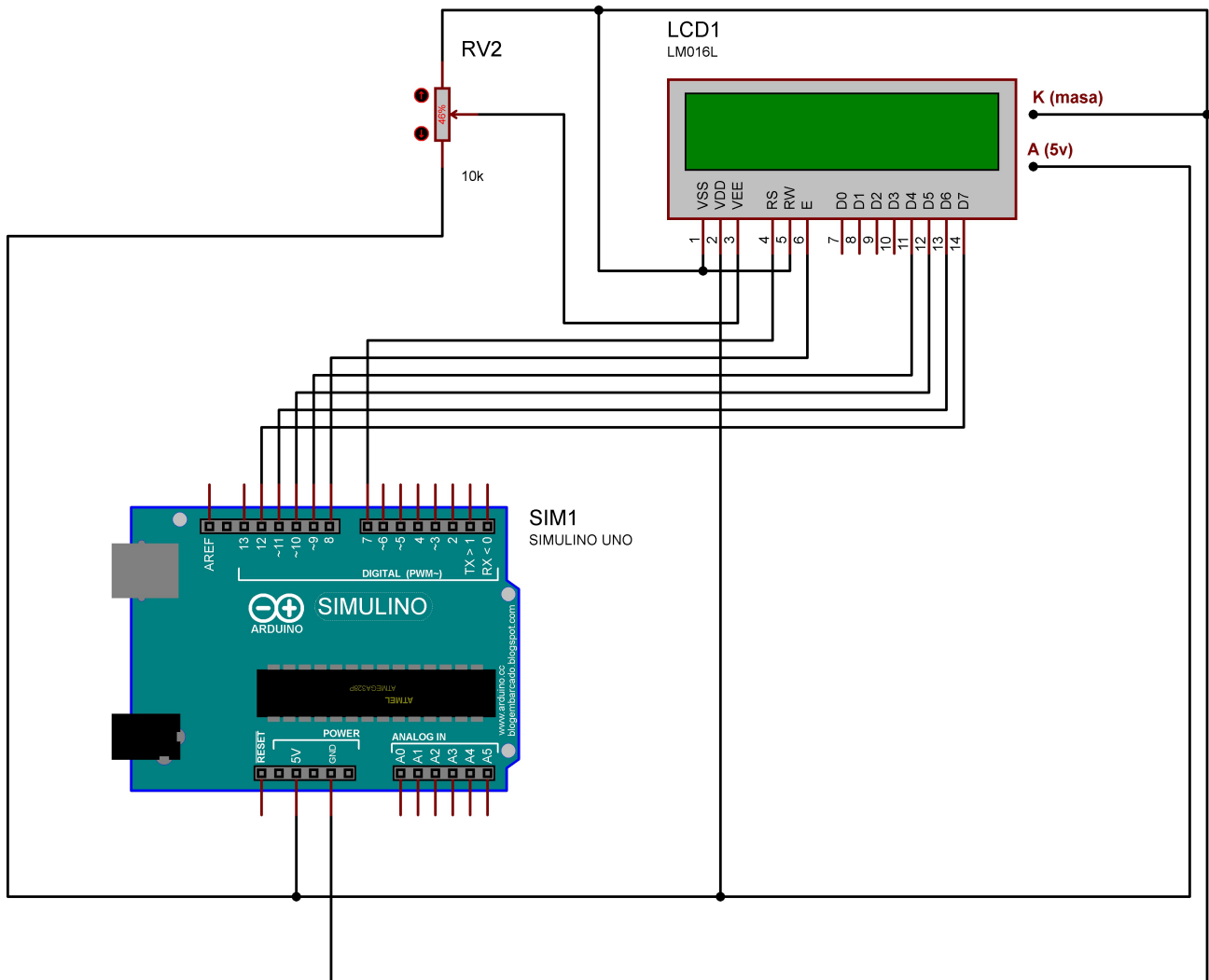
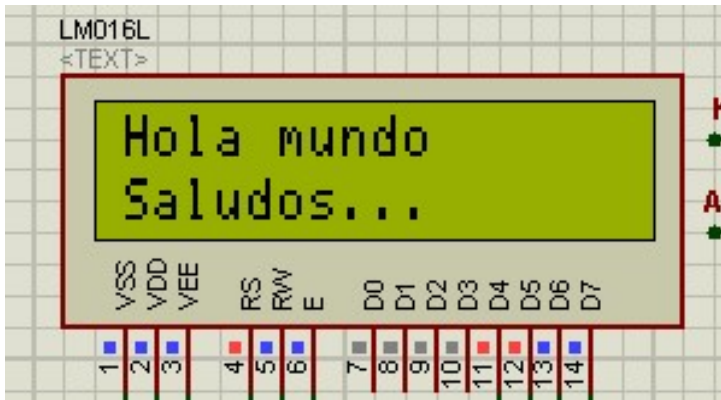
  lcd.setCursor(0, 1); // fijar cursor en columna 0, linea 1 (seria segunda fila)
  lcd.print("Saludos..."); // Enviar el mensaje
}
}
```



The screenshot shows the Arduino IDE interface. The main window displays the sketch code, which is identical to the code block above. The 'Compilado' (Compiled) window at the bottom shows the following output:

```
Usando librería LiquidCrystal con versión 1.0.5 en la carpeta: C:\Archivos de programa\Ardu...
El Sketch usa 1728 bytes (5%) del espacio de almacenamiento de programa. El máximo es 32256...
Las variables Globales usan 63 bytes (3%) de la memoria dinámica, dejando 1985 bytes para l...
```

RESULTADO EN PROTEUS



CARACTERES ESPECIALES (CONTINUARA)