PANTALLA LCD + ARDUINO

Erick Barrios Barocio.
Electrónica v.2025

En muchas aplicaciones de Arduino es común tener un sensor, el cual registra alguna magnitud física y la traduce a algún valor digital de utilidad. Dicho valor se puede mostrar en el puerto serial del IDE (ambiente de programación); sin embargo, en caso de necesitar que el módulo Arduino + sensor se puedan manejar sin necesidad de una PC, es necesaria una pantalla para mostrar los datos de medición. Las pantallas más accesibles y comunes son las LCD, por lo que es conveniente conocer su funcionamiento.

Contenido

Una pantalla de caracteres LCD (Pantalla de Cristal Líquido por sus siglas en inglés) es un dispositivo el cual puede mostrar caracteres ASCII de tamaño fijo. Cada carácter está formado por un arreglo de 5 × 8 pixeles cuadrados (Figura 1). Cada arreglo de pixeles forma el área de un carácter.

El LCD más popular es el de 16 columnas por 2 líneas (16×2) con fondo azul y caracteres blancos, es decir, cuenta con 32 áreas para caracteres. Aunque existen de otras dimensiones (16×1; 16×4; 20×4; etc.) y colores.

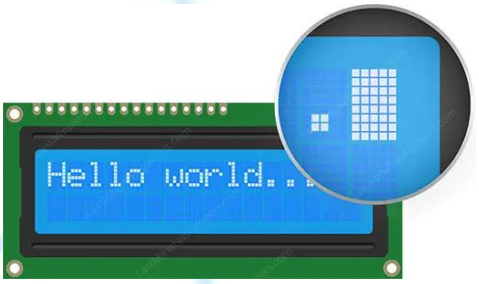

Figura 1. Arreglo de pixeles que componen un carácter en un LCD típico.

1 DISPOSICIÓN DE PINES

Estas pantallas constan de 16 pines para su control (Figura 2) [1]. El pin de la izquierda es el pin de *tierra* o referencia (GND); el segundo es el de *alimentación* de voltaje (VCC), el cual es de 5V y por lo general proviene del Arduino; el tercero es el pin de *control del contraste* (V0), al cual se puede conectar un potenciómetro o una señal de voltaje variable; el cuarto pin es el de *selección de registro* (RS), que se utiliza para seleccionar el modo de envío de comandos o envío de datos al LCD, por ejemplo, si el estado de este pin es bajo (0 Volts), es posible enviar comandos como limpiar pantalla, apagar pantalla u otros, pero si el estado es alto (5V) se envían caracteres. El siguiente pin es el de *lectura y escritura* (R/W), el modo de escritura permite enviar comandos o caracteres a la LCD, mientras que el de lectura es usado para procesos internos de la misma y que no son relevantes para nuestro caso. A continuación, se encuentra el pin de *activación* (E), el cual activa la escritura a los siguientes 8 *pines de datos* D0 a D7, a los cuales se envían conjuntos de datos de 8 bits de acuerdo con la tabla de código ASCII, por ejemplo, el carácter "A" corresponde a un código de bits de "0100 0001". Los últimos dos pines son los de *ánodo* (A) y *cátodo* (K) para la iluminación LED de fondo.

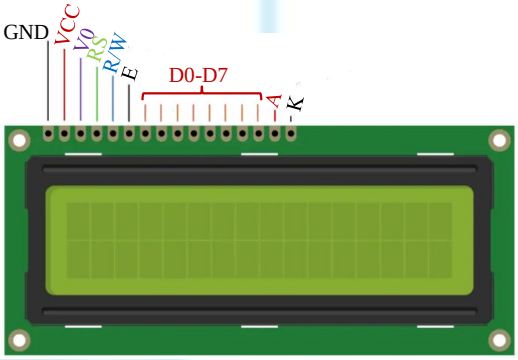

Figura 2. Diagrama de pines de una pantalla LCD.

Totos los módulos de pantalla LCD cuentan con un controlador integrado que procesa la información recibida y la proyectan. En la librería oficial de Arduino [2] se puede encontrar la librería correspondiente con las funciones y comandos necesarios para controlar el LCD, existen librerías para modos de 4 y 8 bits, las cuales difieren en el número de pines de datos utilizados, en este texto se utilizará el modo de 4 bits.

2 CONEXIÓN ARDUINO-LCD

En la Figura 3 se muestran las conexiones de pines Arduino-LCD necesarias para su funcionamiento.

Solo se utilizan 6 pines digitales del Arduino Uno y 4 de los pines de datos de la pantalla LCD (D4-D7). El pin de activación (E) del LCD se conecta al pin 2 del Arduino, mientras que el pin RS se conecta al 1 y el R/W a tierra (GND).

El pin V0 se conecta a la pata central de un potenciómetro de $1K\Omega$ el cual servirá para ajustar el contraste de la pantalla. Es de notar que los polos (patas laterales) del potenciómetro se conectan uno a 5V y otro a tierra, por lo que la pata central podrá variar entre 5V y 0V.

En caso de no contar con un potenciómetro, se puede utilizar un divisor de voltaje simple con dos resistencias, simplemente se tiene que escoger la combinación la combinación apropiada que genere un voltaje tal que el contraste sea adecuado. Por lo general la mejor combinación es $1K\Omega$ y 220Ω, obteniendo el voltaje deseado del punto medio del divisor. Sin embargo, estos valores pueden cambiar de una pantalla a otra.

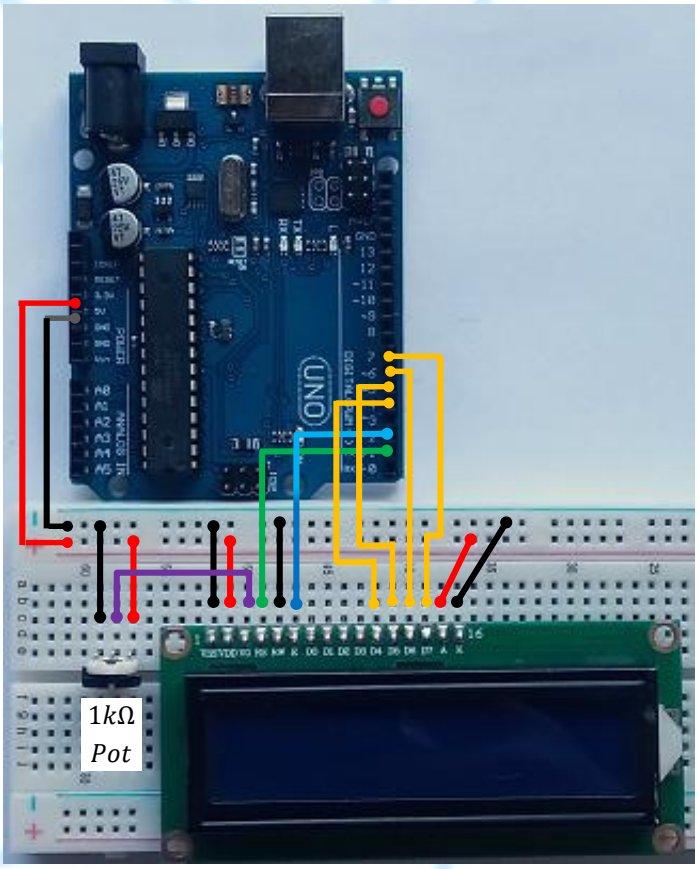

Figura 3. Diagrama de conexión Arduino-LCD.

Otra opción para sustituir el potenciómetro es introduciendo una señal PWM del Arduino al pin V0 del LCD y utilizando la siguiente función:

```
analogWrite(11,100);
```

En este caso, la función hace que se genere una señal PWM en el pin 11 del Arduino con un valor de 100 de 255, donde 255 corresponde a un voltaje promedio de 5V y 0 a 0V. En este caso un valor de 100 es aproximadamente 2V.

3 CODIGO ARDUINO TÍPICO PARA UN LCD.

El siguiente código muestra cómo se utiliza la librería de "*Liquid Crystal*" para controlar el LCD y lagunas funciones básicas.

```
1  #include <LiquidCrystal.h> // Incluye la librería LiquidCrystal
2  LiquidCrystal lcd(1, 2, 4, 5, 6, 7); // Da de alta la LCD, asignando los pines del Arduino las funciones del
3                                       LCD: (rs, enable, d4, d5, d6, d7)
4  void setup() {
5  lcd.begin(16,2); // Inicia la interface LCD, y especifica las dimensiones de (columnas y filas) de la pantalla
6  }
7
8  void loop() {
9  lcd.print("Arduino");          // Imprime "Arduino" en el LCD y desde el origen
10 delay(3000);                   // retraso de 3 segundo
11 lcd.setCursor(2,1);            // Coloca el cursor en la posición donde comenzará a escribir
12 lcd.print("Tutorial LCD");
13 delay(3000);
14 lcd.clear();                   // Limpia la pantalla
15 lcd.blink();                   // Muestra el cursor parpadeando
16 delay(4000);
17 lcd.setCursor(7,1);
18 delay(3000);
19 lcd.noBlink();                 // El cursor deja de parpadear
20 lcd.cursor();                  // Muestra el guion bajo como cursor
21 delay(4000);
22 lcd.noCursor();                // Oculta el cursor
23 lcd.clear();
24 }
```

En la línea 1, el código llama a la librería, la cual debe estar dada de alta en el conjunto de librerías del IDE del Arduino, de no ser así, el código no se ejecutará. Para incluir esta librería en el IDE, se tiene que descargar siguiendo la ruta: Programa > Incluir Librería > Administrar Bibliotecas (Figura 4a); en la ventana emergente, en la casilla de búsqueda, se escribe “LiquidCrystal” y escoger la versión más reciente (Figura 4b).

En la segunda línea, el programa reconoce que se creará (usará) una pantalla LCD conectada a través de los 6 pines indicados como parámetros.

En la sección del “setup”, en la línea 5, se inicializa la pantalla indicando sus dimensiones de caracteres, en nuestro caso 16 × 2 caracteres.

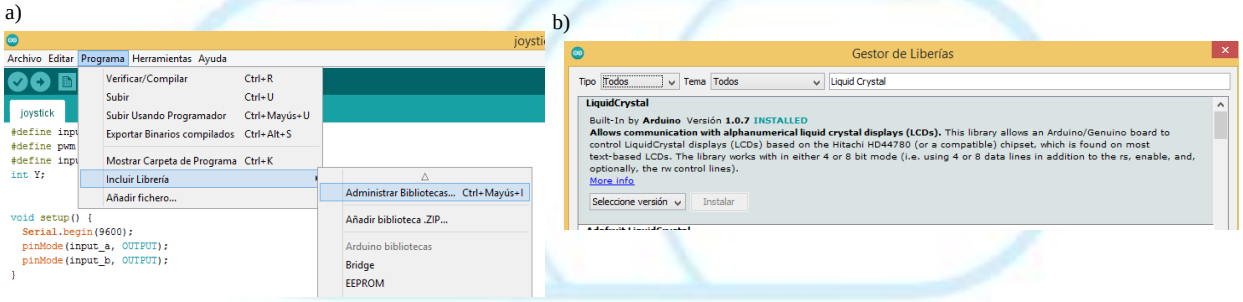

Figura 4. a) Ruta para descargar la librería para la LCD. b) Ventana emergente con resultado de la búsqueda.

Posteriormente, en la sección "loop" se escribe el código principal que manejará la pantalla.

En la línea 6 se utiliza la función "print()", cuyo argumento debe ser el texto que queramos mostrar en la LCD, el cual debe estar entre comillas dobles. Este texto se mostrará en la pantalla comenzando en el origen o posición (0,0) del LCD.

En la línea 11 se utiliza la función "setCursor(columna, fila)", cuyo argumento son dos coordenadas. Esta función cambia la posición del cursor a las coordenadas indicadas, y a partir de esta posición, el texto será impreso. En este ejemplo será en la segunda fila tercer columna. Es de señalar que la primera columna es la cero, al igual que la primera fila.

Las otras funciones básicas mostradas en el código se explican en el mismo mediante comentarios.

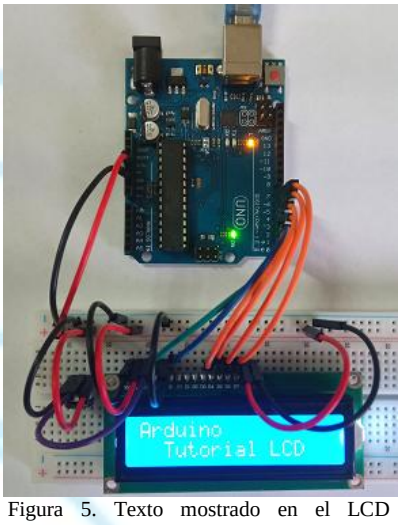

Figura 5. Texto mostrado en el LCD correspondiente al código mostrado.

Así, este código muestra la palabra "Arduino" en el origen (primer columna, primer fila), hay un retraso de 3s y luego muestra la frase "Tutorial LDC" en la segunda fila, pero desplazada 4 espacios (Figura 5); después de otros 3s, el texto se borra y aparece parpadeando el cursor en el origen por 4s, después de lo cual salta a la segunda fila, octava columna y seguirá parpadeando por otros 3s, para posteriormente dejar de parpadear y transformarse en un cursor bajo durante 4s. Finalmente el cursor regresa a su forma original (cuadrado), se borra la pantalla y se repite el programa.

3.1 EJEMPLO DE TEXTO MÓVIL

Existirán situaciones donde se tenga un texto más largo que 16 caracteres (máxima longitud de espacios de la pantalla), por lo que será de gran utilidad si podemos desplazarlo para poder leerlo completo. Esto se puede hacer con ayuda de las funciones "*scrollDisplayLeft()*" y "*scrollDisplayRight()*", las cuales mueven el texto hacia la izquierda o la derecha respectivamente y no necesitan argumentos (los paréntesis se deja vacíos). Estas funciones están incluidas en la librería "LiquidCrystal". Un ejemplo de código es el siguiente:

```
1  #include <LiquidCrystal.h>
2  LiquidCrystal lcd(1, 2, 4, 5, 6, 7);
3
4  void setup() {
5    lcd.begin(16, 2);
6    lcd.print("Ejemplo de Texto Móvil");
7  }
8
9  void loop() {
10   lcd.scrollDisplayLeft();
11   delay(500);
12 }
```

En este ejemplo, el texto se desplaza a la izquierda a una velocidad de un espacio cada medio segundo. Dicha velocidad de desplazamiento puede ser controlada con la función "delay()", la cual requiere de argumento el tiempo por paso en microsegundos. Es de notar que la pantalla comienza en blanco y el texto va apareciendo desde la derecha. En el caso de la función scrollDisplayRight(), la situación sería inversa.

3.2 GENERACIÓN DE CARACTERES PERSONALIZADOS

Además de los caracteres del código ASCII, la librería "LiquidCrystal" permite diseñar y mostrar caracteres personalizados. Para esto es necesario un arreglo matricial de 5 columnas por 8 filas lleno de 0´s y 1´s, este arreglo representa el arreglo de pixeles LED´s de cada espacio para carácter (Figura 1) y se puede construir con el comando "*byte*". Además, también es necesario utilizar las funciones "*createChar()*" y "*write()*". El siguiente código es un ejemplo:

```
1  #include <LiquidCrystal.h>
2  byte sonriza[8] = {  // Arreglo de 8 bytes. Se define como función externa al setup y loop.
3    B00000,            // B indica que se trata de un formato binario y los 5 números son los pixeles
4    B00000,            // un 0 significa pixel apagado, un 1 pixel encendido
5    B01010,
6    B00000,
7    B10001,
8    B01110,
9    B00000,
10   B00000
11   };
12
13 LiquidCrystal lcd(1, 2, 4, 5, 6, 7);
14
15 void setup() {
16   lcd.begin(16, 2);
17   lcd.createChar(0, sonriza);  // Crea el carácter personalizado
18   lcd.clear();
19   lcd.print("Carácter nuevo"); // Imprime texto en el LCD
20 }
21
22 void loop() {
23   lcd.setCursor(7, 1);
24   lcd.write(byte(0));  // Muestra el carácter personal con número 0
25  }
```

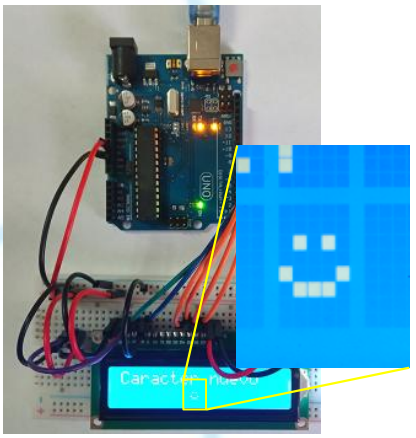

Figura 6. Impresión de un carácter personalizado.

En las líneas 2-11 se construye el carácter mediante el arreglo de 0´s y 1´s de la matriz de 5×8 que representa los pixeles. La función byte requiere de nombrar el nuevo carácter e indicar el número de bytes que contiene (8 en nuestro caso); además, al inicio de cada fila es necesario indicar que se trata de valores binarios, lo cual se hace con una "B".

En la línea 17 se indica que se creará un nuevo carácter. El primer argumento de esta función es un número entre 0 y 7, el cual es el número de carácter personalizado creado (en nuestro caso como solo es uno, lo numeramos como 0); esto implica que solo es posible crear hasta 8 caracteres personalizados en un código, el segundo argumento es el nombre del nuevo carácter.

En la línea 24, con ayuda de la función "write", se imprime el carácter nuevo en la pantalla. Esta función tiene como argumento el arreglo de byte cero.

Así, este código mostrara un carácter de una carita sonriente en la segunda línea séptima posición (Figura 6).

4 USO DE UN LCD CON UN MÓDULO I2C

Una desventaja de utilizar una pantalla LCD del modo presentado en la sección 2 (Figuras 3 y 5) es que requiere de 6 pines digitales, lo cual para el caso de un Arduino UNO es casi la mitad de los pines disponibles. Una solución a este problema es utilizar un modulo de comunicación I2C, el cual utiliza únicamente dos pines (SDA y SCL) del Arduino.

4.1 MÓDULO I2C

Este módulo tiene un chip PCF8574 expansor de pines de entrada-salida digitales (E/S) de 8 bits, convierte los datos I2C generados por un Arduino en datos necesarios para una pantalla LCD (Figura 7). El módulo incluye el potenciómetro de control de contraste de la pantalla y un puente que suministra energía a la iluminación de fondo, la cual también puede ser ajustada quitando el puente y aplicando un voltaje externo al pin del encabezado que está marcado como 'LED'.

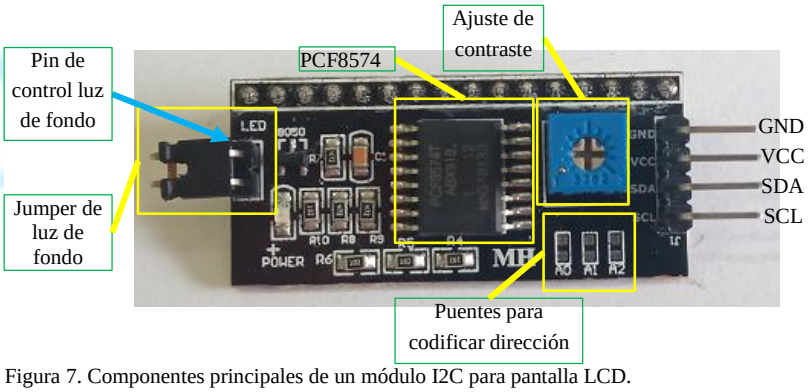

Figura 7. Componentes principales de un módulo I2C para pantalla LCD.

A) SELECCIÓN DE LA DIRECCIÓN I2C DEL MÓDULO

El módulo I2C para la pantalla permite utilizarla de forma simultánea (en paralelo) con otros dispositivos que cuenten con el protocolo de comunicación I2C en el mismo bus. Sin embargo, para esto es necesario configurar una dirección I2C diferente para dispositivo evitando que entren en conflicto.

Para ello, el adaptador dispone de tres puentes de soldadura (A0, A1 y A2), los cuales permiten codificar una dirección. Si un puente se cortocircuita con una gota de soldadura, establece un número de dirección. Algo a tener en cuenta es que, dependiendo de la empresa que produjo el PCF8574, la codificación de la dirección será distinta. Actualmente los chips más comunes son los de Texas Instruments y NXP Semiconductors.

I. Codificación de Texas Instruments.

En este caso, los tres bits de selección de dirección (A0, A1 y A2) se colocan al final del registro de dirección I2C de 7 bits de acuerdo al arreglo:

Registro de dirección						
0	1	0	0	A2	A1	A0

Bit más significativo — Bit menos significativo

Dado que hay libertad de selección en 3 entradas de dirección, que pueden tomar 2 estados (ALTO/BAJO), se pueden seleccionar 8 direcciones diferentes. De forma predeterminada, las 3 entradas de dirección están en ALTO cuando los puentes no están conectados, lo que implica una dirección de 0100111 Binario (0x27 Hexadecimal). Al cortocircuitar los puentes con soldadura, las entradas de dirección se ponen en BAJO. El rango de todas las direcciones posibles abarca desde 0x20 hasta 0x27 (Figura 8).

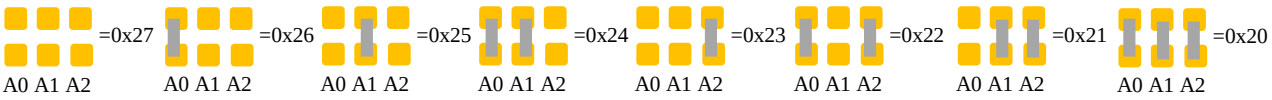

Figura 8. Direcciones generadas dependiendo de la conexión de puentes para un chip PCF8574 Texas Instruments.

II. Codificación NXP

Para este chip, los tres bits de selección de dirección (A0, A1 y A2) también se colocan al final del registro de dirección, pero la diferencia radica en los primeros bits.

Registro de dirección						
0	1	1	1	A2	A1	A0

Bit más significativo — Bit menos significativo

El número de direcciones posibles seguirá siendo 8; sin embargo, las posibles direcciones van de 0x38 a 0x3F (en hexadecimal), siendo la predeterminada (cuando los puentes no están soldados) 0x3F (0111111 Binario).

Así, el LCD probablemente tenga una dirección I2C predeterminada 0x27 Hex o 0x3F Hex, por lo que es necesario encontrarla. La forma de hacer esto se menciona más adelante.

B) PINES DEL MÓDULO I2C

Como se mencionó anteriormente, el módulo solo usa 4 pines de conexión (Figura 7):

- ***GND*** es el pin de tierra, el cual se conecta a la tierra del Arduino.
- ***VCC*** es el pin de energía del módulo y LCD, se conecta a la salida de 5V del Arduino o fuente externa.
- ***SDA*** es el pin de datos I2C. Se conecta al pin datos I2C del Arduino.
- ***SCL*** es el pin del reloj I2C. Se conecta al pin de reloj I2C del Arduino.

Por otro lado, el módulo tiene en su parte superior 16 pines, que son lo pines que controlan la pantalla LCD, por lo que estos pines se deben conectar a la pantalla de la forma mostrada en la Figura 9.

C) CONEXIÓN CON UN ARDUINO UNO

Para controlar la pantalla LCD con su modulo I2C, solo son necesarios 4 pines de un Arduino UNO, siendo dos de ellos el pin VCC (o salida de 5V) y el GND (tierra).

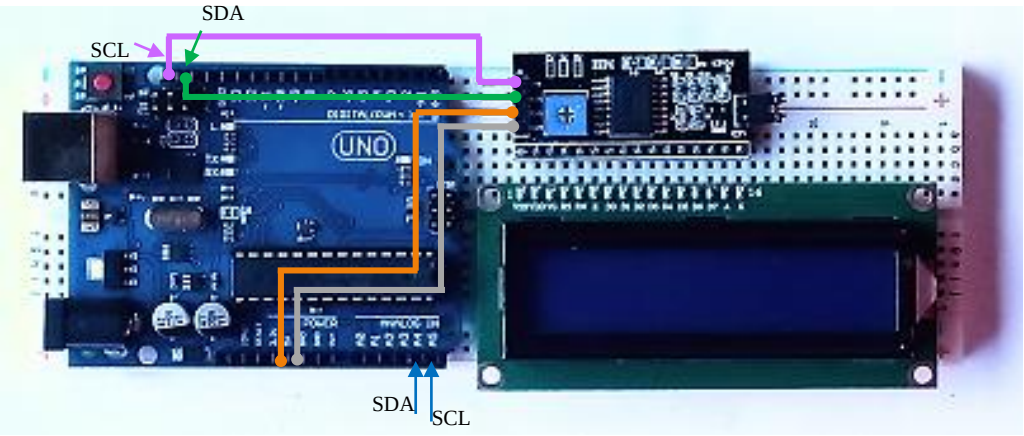

Figura 9. Diagrama de conexiones Arduino UNO - Módulo I2C/LCD – Pantalla LCD 16x2.

Los otros dos pines son los de comunicación I2C. Un Arduino UNO R3 existen cuatro de estos pines (dos de SDA y dos de SCL). Un par SDA/SCL se encuentran junto al pin AREF del Arduino cerca de la conexión USB, mientras que el otro par son los pines analógicos A4 y A5 respectivamente (Figura 9). Así, simplemente hay que conectar los pines SDA y SCL del módulo I2C/LCD a los correspondientes pines del Arduino.

D) INSTALACIÓN DE LA BIBLIOTECA Y DETERMINACIÓN DE LA DIRECCIÓN I2C

La librería adecuada para que el Arduino se comunique con el módulo I2C/LCD es la "*LiquidCrystal_I2C*", la cual se puede descargar de la página de librerías de Arduino[3] e instalándola en el folder de Librerías del Arduino IDE.

Una vez instalada la librería, lo primero es determinar la dirección del módulo (la cual depende del fabricante) y puede ser una de dos opciones (0x27 Hex o 0x3F Hex). Para eso se utiliza el siguiente código cuya autoría se atribuye a Nick Gammon [4]:

```
1   #include <Wire.h>
2   void setup() {
3     Serial.begin (9600);
4     while (!Serial) {
5               }
6     Serial.println ();
7     Serial.println ("Escaner I2C. Buscando...");
8     byte count = 0;
9
10    Wire.begin();
11      for (byte i = 8; i < 120; i++) {
12          Wire.beginTransmission (i);
13          if (Wire.endTransmission () == 0) {
14            Serial.print ("Dirección encontrada: ");
15            Serial.print (i, DEC);
16            Serial.print (" (0x");
17            Serial.print (i, HEX);
18            Serial.println (")");
19            count++;
20            delay (1);
23          }
24       }
25    Serial.println ("Completado.");
26    Serial.print ("Dispositivos encontrados: ");
27    Serial.print (count, DEC);
28  }
29
30  void loop() {}
```

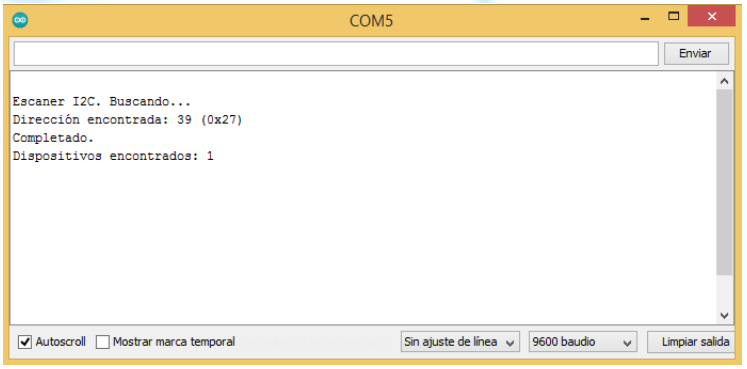

Figura 10. Respuesta en el puerto serial a la búsqueda de la dirección del módulo I2C/LCD.

Este código escanea el bus I2C y devuelve la dirección de cada dispositivo I2C que encuentra, mostrando su dirección en el puerto serial del Arduino IDE (Figura 10).

En nuestro caso, se encontró que el módulo I2C/LCD tiene la dirección 39 o 0x27 hexadecimal. Esta dirección se usará posteriormente para controlar la pantalla.

4.2 CODIGOS BÁSICOS DE EJEMPLO.

El siguiente código ejemplifica como utilizar la librería "*LiquidCrystal_I2C*" y algunos de sus comandos.

```
1  #include <LiquidCrystal_I2C.h>
2  LiquidCrystal_I2C lcd(0x27,16,2);  // Define la dirección y tamaño de la pantalla
3
4  void setup() {
5    lcd.init();
6    lcd.clear();
7    lcd.backlight();      // Enciende la luz de fondo
8
9    lcd.setCursor(2,0);   // Pone el cursor en la columna 3 de la primera fila
10   lcd.print("Hola Mundo!");
11   lcd.setCursor(2,1);   // Mueve el cursor a la tercer columna de la segunda fila
12   lcd.print("Tutorial LCD");
13 }
14
15 void loop() {
16 }
```

En la línea 1, se llama a la librería que es la que contiene las funciones necesarias para controlar el módulo.

En la línea 2, se define la existencia de una pantalla LCD en la dirección I2C "0x27", y con dimensiones de 16 columnas por 2 filas.

En la línea 5 se inicializa el módulo, mientras que en la 6 se limpia lo que tenga la pantalla y en la 7 se enciende la luz de fondo. Los comandos en las líneas 9-12, son los mismos ya vistos anteriormente.

Todas las demás funciones mencionadas en la sección 3 siguen siendo válidas e iguales. Al igual que la metodología para crear caracteres personalizado, con la única diferencia de que siempre se tiene que trabajar bajo la librería I2C.

5 REFERENCIAS

[1] Dejan, *Arduino 16×2 LCD Tutorial*, How to Mechatronix, 2024. https://howtomechatronics.com/tutorials/arduino/lcd-tutorial/

[2] Arduino Library List. 2024. https://www.arduinolibraries.info/

[3] Frank de Brabander, *Liquid Crystal I2C*, Arduino Library List. 2024. https://www.arduinolibraries.info/libraries/liquid-crystal-i2-c

[4] MCI Education. *Interfaz de un LCD I2C con Arduino*. 2024. https://cursos.mcielectronics.cl/2022/12/09/interfaz-de-un-lcd-i2c-con-arduino/