UNIDAD 6: PANTALLAS LCD

OBJETIVOS

Puedes suponer que hay una gran variedad de periféricos digitales de entrada ode salida. Hasta ahora has usado los clásicos, sencillos y económicos pulsadores/interruptores comodispositivos para introducir niveles lógicos “1” y “0”, y los diodos leds para representarlos.

Sin embargo es hora de hablar de otros periféricos digitales con mucho más renombre eimportancia. En esta Unidad vamos a hablar de la pantalla LCD como periférico de salida. Se trata de un periférico que te va a permitir visualizar cualquier tipo de información de salida como pueden ser números, letras y símbolos.

TEORÍA

- LA PANTALLA LCD
- EL JUEGO DE CARÁCTERES
- LOS CARÁCTERES GRÁFICOS
- LA LIBRERÍA LiquidCrystal.h
- LA MEMORIA EEPROM DE DATOS
-

PRÁCTICA

- CONEXIÓN DE LA PANTALLA LCD
- EJEMPLO 3-1: Hola, mundo !!
- EJEMPLO 3-2: Display
- EJEMPLO 3-3: Cursor
- EJEMPLO 3-4: Blinki
- EJEMPLO 3-5: Dirección del texto
- EJEMPLO 3-6: Scroll
- EJEMPLO 3-7: AutoScroll
- EJEMPLO 3-8: Carácteres personalizados
- EJEMPLO 3-9: Visualización

- EJEMPLO 3-10: Visualizando número enteros
- EJEMPLO 3-11: Visualizando número float
- EJEMPLO 3-12: Su turno
- EJEMPLO 3-13:Menú
- EJEMPLO 3-14: Selección de opciones
- EJEMPLO 3-15: Una última mejora

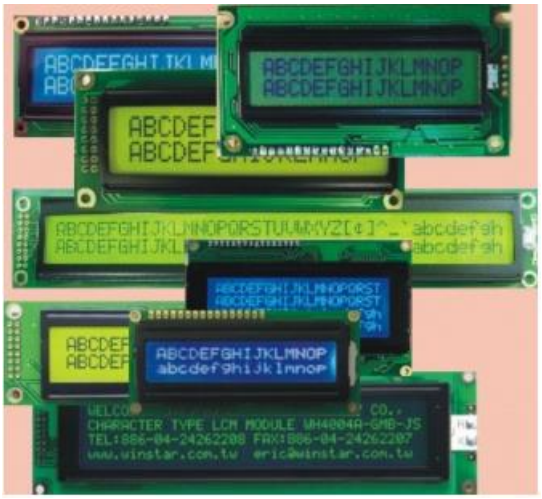
MATERIALES

- ✓ Ordenador portátil o de sobremesa.
- ✓ Entorno de trabajo Arduino IDE que se incluye en el material complementario y que se supone instalado y configurado.
- ✓ Tarjeta controladora Arduino UNO
- ✓ Pantalla LCD 2x16
- ✓ Cable USB.

TABLA DE CONTENIDOS

TEORÍA

1. LA PANTALLA LCD



Se trata de un periférico de salida que permite visualizar no sólo números, sino también todo tipo de caracteres, textos, símbolos e incluso sencillos gráficos. Seguro que las has visto en infinidad de aplicaciones. Las puedes encontrar con diferentes números de líneas y caracteres por línea. También las hay con luz de fondo y caracteres de diferentes colores y tamaños.

Todas ellas llevan su propio controlador que gestiona todas las operaciones internas. Normalmente la mayoría son compatibles con el popularHD44780 de Hitachi. Gracias a esto, es prácticamente lo mismo utilizar una pantalla de2líneaspor16 caracteres (2 x 16) que otra de 4 x 20.

Vas a utilizar una pantalla LCD de2x16.En la figura tienes una imagen de la misma, una representación simplificada y su símbolo eléctrico (**¡Error! No se encuentra el origen de la referencia.**).

Una pantalla LCD es un periférico digital. Sus señales se pueden conectar directamente con las patillas de E/S del controlador y se dividen en tres grupos:

- **Alimentación**: Por ellas se aplica la tensión de alimentación de la pantalla. Normalmente son las patillas GND y VCC de +5V. También hay una tercera patilla,

VLC, por donde se aplica una tensión variable entre 0V y +5V con la que se puede ajustar el contraste de la pantalla.

- **Control**: Son señales con las que se determina si la pantalla recibe un comando o un dato, si va a ser leída o escrita por el controlador o si la pantalla se habilita o no.
- **Datos**: Por estas señales el controlador envía a la pantalla los comandos o los datos propiamente dichos.

Lapatillanº1eslaprimeradelaizquierda.Miralasiguientetablaconla descripción de cada una de ellas (Figura 2).

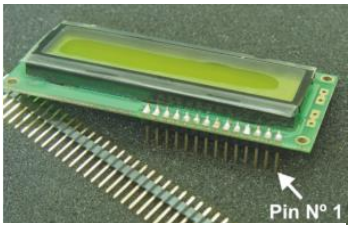


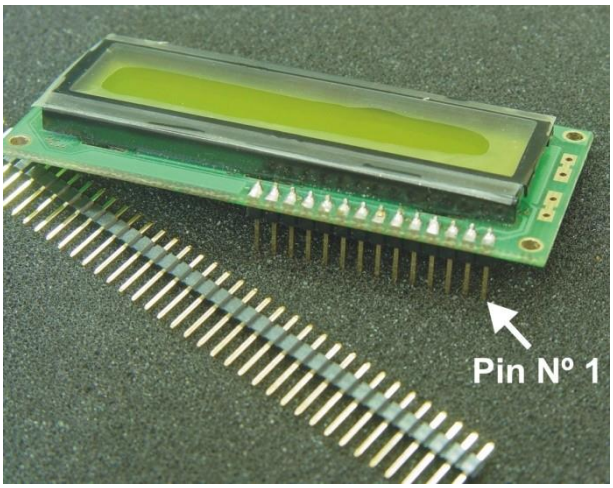
Figura 1

PATILLA Nº	NOMBRE	TIPO	DESCRIPCIÓN
1	Vss	Alimentación	**Alimentacióndetierra(0V).**
2	Vdd	Alimentación	**Alimentaciónpositivade+5Vcc.**
3	VLC	Alimentación	**Ajustedecontraste.Seleaplicaunatensiónvariableentre0y+5Vcc.**
4	RS	Entrada	**SalidadesdeelArduino.Selecciódeinstrucciones/datos: RS=0ElArduinovaatransferirinstrucciones RS=1ElArduinovaatransferirdatos(códigosASCII)**
5	R/W	Entrada	**SalidadesdeelArduino.Controlde lectura/escritura: R/W=0ElArduinorealizaunaescriturasobrelapantallaLCD R/W=1ElArduinorealizaunalecturadelapantallaLCD**
6	E	Entrada	**SalidadesdeelArduino.Habilitacióndelapantalla: E=0PantallaLCDdeshabilitada(enaltaimpedancia) E=1PantallaLCDhabilitada**
7-14	DB0:DB7	E/S	**Busdeinstrucciones/datos.ElArduinotransfierealapantalla instruccionesodatosenfunciónde laseñalRS. Conuninterfacede8bitsseempleanlaslíneasDB0:DB7 Conuninterfacede4bitsseempleanlaslíneasDB4:DB7**
15	L+	Alimentación	**Tensiónpositivaparalaluzdefondo(+5 Vcc).**
16	**L-**	**Alimentación**	**Tensiónnegativaparalaluzdefondo(0V).**

Las patillas 15 y 16 son opcionales. Pueden incluso que ni estén disponibles en la pantalla. Solo las emplean aquellas que dispongan de luz de fondo. En otras, como es nuestro caso, no se usan.

Se trata de uno de los periféricos más potentes y versátiles. Como ya decíamos es capaz de visualizar todo tipo de mensajes compuestos de texto, números y símbolos, y producir además diferentes efectos de visualización como desplazamientos a izquierda y derecha, parpadeos, scrolls, etc… Una pantalla LCD tiene su propio controlador quela gestiona.

En la figura tienes la pantalla LCD que vas a utilizar y que se incluye en el kit de materiales propuestos. Se trata de una pantalla compuesta de 2 filas y 16 caracteres por fila (2x16). También puedes apreciar cómo debes soldar una tira de 14/16 pines macho, que te facilitará su posterior inserción sobre el módulo board de prácticas. De esta forma podrás hacer rápidamente las conexiones con el controlador. Observa la posición en que se encuentra la patilla número 1.



2. EL JUEGO DE CARACTERES

Básicamente la comunicación entre Arduino y la pantallas LCD se realiza mediante las patillas de datos DB0-DB7. Se pueden usar las 8 patillas o, como en nuestro caso, únicamente 4. Se dice que vamos a trabajar con un “interface de 4 bits”.

A través de esas patillas Arduino transmite ciertos comandos o instrucciones con los que la pantalla puede realizar diferentes efectos de visualización: desplazamientos, parpadeos, borrado, colocación del cursor, etc… También transmite los códigos ASCII de los caracteres que deseas visualizar. Estos códigos son de 8 bits. Si empleas un “interface de 8 bits”, cada carácter se visualiza con una única transferencia por parte de Arduino. Con un “interface de 4 bits” hacen falta dos transferencias por cada carácter a visualizar. Es algo más lento pero también empleas menos cables de conexión. DE todas formas no te preocupes, las funciones que vas a estudiar te facilitarán enormemente la tarea.

En la figura siguiente tienes una muestra del juego de caracteres que admite la pantalla LCD, y que están establecidos por el fabricante. Una memoria ROM interna contiene la definición de cada uno de ellos, y puede variar entre diferentes modelos o versiones.

A la izquierda, en las filas, se representan los 4 bits de menos peso del carácter (B3:B0). En la parte superior de la tabla, en las columnas, se representan en binario los4 bits de más peso del carácter (B7:B4). Para codificar un carácter cualquier basta con seleccionarlo y localizar en qué columna/fila se encuentra. Por ejemplo el carácter ‘F’ se encuentra en la columna 4 (0100)

y en la fila 6 (0110). Su código binario es por tanto 0100 0110 (0x46) que se corresponde exactamente con el código ASCII del carácter 'F'.

3. LOS CARACTERES GRÁFICOS

Puedes crear un total de hasta 8 caracteres gráficos de 5x8 puntos o "pixels". Cada carácter se numera del 0 al 7 y necesita un total de 8 bytes para ser definido. Para ello, la pantalla LCD dispone de una memoria RAM interna llamada CGRAM. Una vez que están definidos, los puedes visualizar enviando su número correspondiente (entre 0 y 7). Recuerda, los caracteres gráficos se almacenan en RAM. Si desconectas la alimentación del sistema estos se perderán. Arduino tendrá que volver a definirlos.

Los caracteres gráficos se definen introduciendo en sucesivas posiciones de esa memoria CGRAM, unos bytes cuyos patrones binarios definen el carácter. La CGRAM consiste en una memoria volátil capaz de almacenar un total de 64 bytes. Un carácter de 5x8 puntos necesita 8 de esos bytes para ser definido. Puedes por tanto definir a tu gusto hasta 8 caracteres diferentes (8 x 8).

En la siguiente tabla tienes definidos, a modo ejemplo, cuatro caracteres de5x8. Se introducen en las 32 primeras posiciones de la CGRAM. El primero en las posiciones 0 a 7, el segundo en las posiciones 8 a 15 y así sucesivamente.

Cada bit de cada uno de esos bytes que valga nivel "1", implica que su correspondiente punto o pixel en el LCD se activa. El primer carácter gráfico de la CGRAM se visualiza enviando el número 0 como si de un código ASCII se tratara. El segundo carácter se visualiza enviando el número 1 y así con todos los que hubieras definido.

Para definirlos basta con que crees un array o matriz por cada carácter. Luego, mediante la función crateChar() que vas a estudiar enseguida, el contenido de cada array se copia sobre la CGRAM de la pantalla. Tienes que crear tantos arrays como caracteres desees, cuatro en este ejemplo:

1. byte corazon [8] = {10,21,17,17,17,10,4,0};
2. byte sonrisa [8] = {B0,B01010,B0,B00100,B00000,B10001,B01110,B0};
3. byte letra_ñ [8] = {31,0,24,27,17,17,17,0};
4. byte letra_ú [8] = {2,4,17,17,17,19,13,0};

Como puedes ver, los puedes definir en decimal, binario, etc…

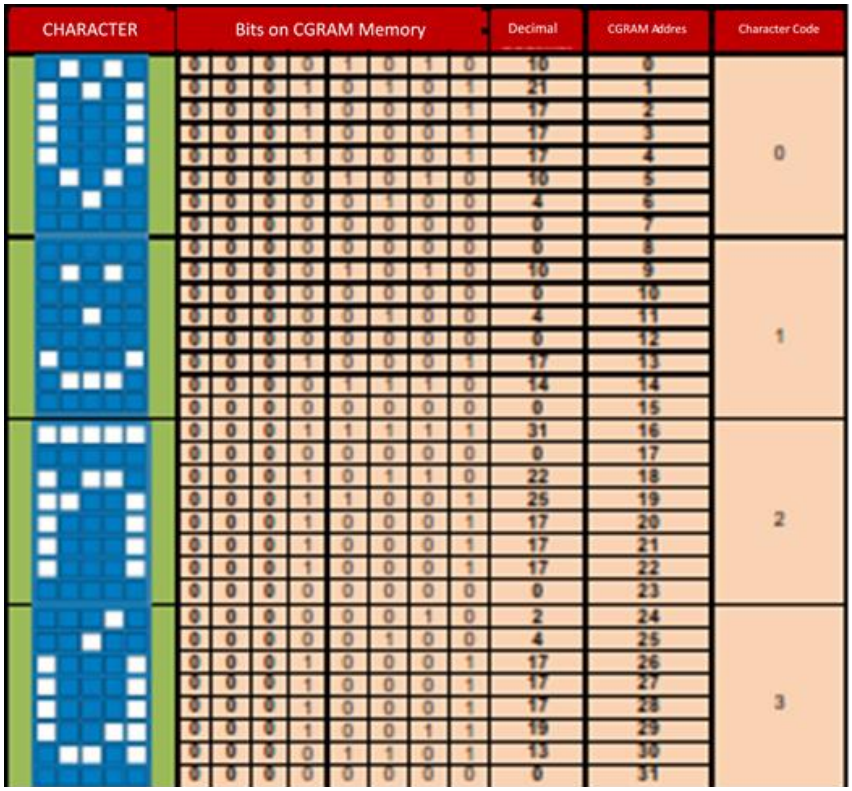


Figura 2

4. LA LIBRERÍA LIQUIDCRYSTAL.H

A estas alturas ya debes estar familiarizado con el concepto de “librería” y el manejo de las múltiples funciones que puede contener. Acuérdate de la librería “Servo.h” que usaste en el anterior curso “Arduino: La tecnología al alcance todos” para el control de servomotores.

De las muchas librerías creadas por el equipo de Arduino, vamos a hablar ahora de la librería “LiquidCrystal.h”. Contiene un buen número de funciones dedicadas al control y manejo de pantallas LCD basadas en el controlador Hitachi HD44780 o compatible.

De todas esas funciones, vamos a estudiar las más representativas e importantes. En cualquier caso, te invito a que visites el sitio www.arduino.cc donde encontrarás abundante información y ejemplos con todas ellas.

Función: LiquidCrystal()

Crea una variable tipo “LiquidCrystal” y establece las conexiones entre la pantallaLCD y el controlador Arduino. El interface entre ambos puede ser de 8 o de 4 bits, en cuyo caso las líneas D0-D3 no se emplean. Igualmente puedes determinar si se emplea o no la señal R/W de la pantalla. En caso de no usarse, como va a ser el nuestro, dicha señal debes conectarla con GND. Ya lo harás en el área de prácticas.

Sintaxis:

LiquidCrystal var(RS,E,D4,D5,D6,D7); //Para interface de 4 bits sin señal R/W LiquidCrystal

var(RS,RW,E,D4,D5,D6,D7); //Para interface de 4 bits con señal R/W LiquidCrystal

var(RS,E,D0,D1,D2,D3,D4,D5,D6,D7); //Interfacede 8bits sin señal R/W LiquidCrystal

var(RS,RW,E,D0,D1,D2,D3,D4,D5,D6,D7); //Interfacede 8bits conbseñal R/W

var: Nombre de la variable que se asigna a la pantalla LCD que vas a controlar.

RS: Patilla del Arduino que se conecta con la señal RS de la pantalla.

RW: Patilla del Arduino que se conecta con la señal R/W de la pantalla (si es que se va a usar).

E: Patilla del Arduino que se conecta con la señal E de la pantalla.

D0-D7: Patillas de Arduino que se conectan con las líneas de datos DB0-DB7 de la pantalla. Si no se indican las patillas para DB0-DB3, se supone un interface de 4 bits y únicamente se emplean las señales DB4-DB7.

Ejemplo:

LiquidCrystal lcd(7,8,9,10,11,12); // Establece la conexión de una pantalla llamada " lcd". Las patillas D7 a D12 del Arduino se conectan con las señales RS, E, DB4, DB5, DB6 y DB7 de la pantalla. La señal R/W se debe conectar con GND.

Función: Begin()

Inicia la pantalla LCD y le asigna el número de filas y el número de caracteres porfila según el modelo de que se trate. En nuestro caso vas a usar una pantalla de 16 x 2 caracteres.

Sintaxis:

var.begin(c,f);

var: Es el nombre quedefine alapantalla en cuestión (establecido enLyquidCrystal()).

c: Número de columnas.

f: Número de filas.

Ejemplo:

LiquidCrystal lcd(7,8,9,10,11,12); //Conexiones de la pantalla LCD*lcd.begin(16,2);* //La pantalla "lcd" es de 16 x 2

Función:setCursor()

Coloca el cursor de la pantalla LCD en la posición deseada. A partir de ella se iránvisualizando los posteriores caracteres.

Sintaxis:

var.setCursor(c,f);

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

c: Número de columna (contando desde 0).

f: Número de fila (contando desde 0).

Ejemplo:

LiquidCrystal lcd(7,8,9,10,11,12); //Conexiones de la pantalla LCD

lcd.setCursor(3,0); //Coloca el cursor en la posición 3 (4º carácter)de la fila 0 (la 1ª)

Función: home()

Coloca el cursor en la primera posición de la pantalla, el ángulo superior izquierdo(posición 0 de la fila 0). No se borra lo que se estuviera visualizando en ese momento.

Sintaxis:

var.home();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Función: clear()

Borra la pantalla LCD y coloca el cursor en el ángulo superior izquierdo (posición0 de la fila 0).**Sintaxis**:

var.clear();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Ejemplo:

LiquidCrystal lcd(7,8,9,10,11,12); //Conexiones de la pantalla LCD

lcd.clear(); //Borra la pantalla

Función:write()

Escribe un carácter en la posición actual del cursor.

Sintaxis:

var.write(char);

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

char: El carácter a visualizar

Ejemplo:

lcd.write('A'); //Escribe 'A'

Función: print()

Imprime sobre el LCD a partir de la posición actual del cursor.

Sintaxis:

var.print(data);

var.print(data,base);

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

data: Dato a imprimir. Puede ser char, int, long, float o string.

base: Es opcional y expresa la base de numeración deseada: BIN=Binario; DEC=Decimal(por defecto); OCT=Octal; HEX=Hexadecimal; o N=nº de decimales en el caso delos float (por defecto 2).

Ejemplos:

int A=19;

float PI=3.1416;

lcd.print(A,HEX); *//Imprime A en hexadecimal ($19_{10} = 13_{16}$)*

lcd.print("Hello"); //Imprime "Hola"

*lcd.print(PI*2,4);* //Imprime 6.2832 (cuatro decimales)

Función: cursor()

Visualiza el cursor en la pantalla LCD, en su posición actual, en forma de un guionbajo (_). A partir de aquí es donde se escribirá el siguiente carácter.

Sintaxis:

var.cursor();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Función: noCursor()

Oculta la visualización del cursor de la pantalla LCD.

Sintaxis:

var.noCursor();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Función: blink()

Visualiza el cursor sobre la pantalla, en su posición actual, como un símbolo sólido(▓) intermitente. A partir de aquí es donde se escribirá el siguiente carácter.

Sintaxis:

var.blink();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Función: noBlink()

Oculta la visualización del cursor sólido (▓) intermitente.

Sintaxis:

var.noBlink();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Función:noDisplay()

Apaga la pantalla LCD sin perder ni el contenido que hubiera en ella ni la posiciónactual del cursor.

Sintaxis:

var.noDisplay();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Función: display()

Conecta la LCD restaurando el contenido que hubiera en ella antes de ejecutarnoDisplay().**Sintaxis:**

var.display();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Función: scrollDisplayLeft()

Desplaza el contenido que visualiza la pantalla en ese momento (texto y posicióndel cursor) una posición a la izquierda.

Sintaxis:

var.scrollDisplayLeft();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Función: scrollDisplayRight()

Desplaza el contenido que visualiza la pantalla en ese momento (texto y posicióndel cursor) una posición a la derecha.

Sintaxis:

var.scrollDisplayRight();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Función: leftToRight()

Establece la dirección de escritura sobre la pantalla como de izquierda a derecha(por defecto). Esto implica que los caracteres se van escribiendo de izquierda aderecha sin afectar a los que ya están escritos.

Sintaxis:

var.LeftToRight();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Función:rightToLeft()

Establece la dirección de escritura sobre la pantalla como de derecha aizquierda. Esto implica que los caracteres se van escribiendo de derecha a izquierda sinafectar a los que ya están escritos.

Sintaxis:

var.RightToLeft();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Función: autoscroll()

Activa el scroll o desplazamiento automático de la visualización. Cada vez que semanda un carácter a la pantalla, este se visualiza y a continuación se desplaza unaposición todo el contenido que hubiera en ella. Si la dirección actual es de izquierda aderecha (leftToRight()), el contenido se desplaza hacia la izquierda. Si la direcciónactual es de derecha a izquierda (rightToLeft()), el contenido se desplaza hacia laderecha.

Sintaxis:

var.autoscroll();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Función: noAutoscroll()

Desactiva el scroll o desplazamiento automático de la visualización.

Sintaxis:

var.noAutoscroll();

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

Función: createChar()

Crea un carácter definido por el usuario. Se pueden crear un total de 8caracteres de 5 x 8 pixels y numerados del 0 al 7. La apariencia o diseño de un carácterse determina en una matriz de 8 bytes, uno por cada fila. Los 5 bits de menos peso decada byte se corresponden con los 5 pixels que forman cada fila del carácter. Una vezcreado un carácter, se puede visualizar en la pantalla indicando simplemente su número.

Sintaxis:

5. *var.createChar(n,dato);*

var: Es el nombre que define a la pantalla en cuestión (establecido enLyquidCrystal()).

n: Expresa el número del carácter a definir (del 0 al 7).

data: Es el nombre de la matriz que contiene los bytes que definen el carácterpersonalizado.

Ejemplo:

byte arrowhead [8] = {B00100, B01110, B10101, B00100, B00100, B00100, B00100, B00100};
//Crea la matriz "flecha" con 8 bytes que definen//el nuevo carácter gráfico

lcd.createChar(0, arrowhead); //Crea el nuevo carácter nº 0 con el contenido definido en el array "flecha"

…

lcd.write(byte(0)); //Visualiza el carácter gráfico nº 0 (la flecha)

6. THE EEPROM DATA MEMORY

Aunque parece que no tiene relación con lo que se está tratando en esta Unidad,y la verdad es que no la tiene, es un buen momento para hablar de la memoria EEPROM de datos del controlador Arduino UNO. Con ella vas a poder mejorar y potenciar infinidad de proyectos presentes y futuros.

Los controladores que conforman las diferentes tarjetas Arduino integran una memoria un poco especial. Se trata de la memoria EEPROM. Sobre ella se pueden leer o escribir datos de 8 bits comprendidos entre 0y 255. La particularidad es que aunque falte la tensión de alimentación, toda la información que hubieras guardado en ella, se mantiene. No se borra.

Por tanto es ideal para almacenar información que puede ser modificada pero NO volátil como: códigos de acceso, parámetros de configuración, números de teléfono, claves, etc… Como ya sabes, la tarjeta Arduino UNO incorpora un controlador modelo AT mega 328 que integra una memoria EEPROM de 1KB (1024bytes).

Su manejo, a estas alturas, te va a resultar muy fácil. Dispones de una librería, la, “EEPROM.h”, que contiene dos únicas funciones con las que puedes leer y escribir datos en esta memoria. Como de costumbre, esa librería la debes incluir en tus programas mediante **#include <EEPROM.h>.**

Función:read()

Lee el dato de 8 bits que haya en la posición de la memoria EEPROM indicada.

Sintaxis:

EEPROM.read(dir);

dir: Se trata de la dirección que vas a leer de la EEPROM. Como tiene 1024posiciones (KB), el rango posible es de 0 a 1023.

Ejemplo:

#include EEPROM.h //Incluye la librería

bytevalue; //Variable de 8 bits

value = EEPROM.read(279); //Leer el byte que hay en la posición 279

Función:write()

Escribe un valor de 8 bits (0-255) en cualquiera de las posiciones disponibles enla EEPROM.

Sintaxis:

EEPROM.write(dir, value);

dir: Se trata de la dirección de la EEPROM sobre la que vas a escribir. Como tiene1024 posiciones (KB), el rango posible es de 0 a 1023.

value: Es el valor de 8 bits (entre 0 y 255) que deseas escribir en la dirección indicada.

Ejemplo:

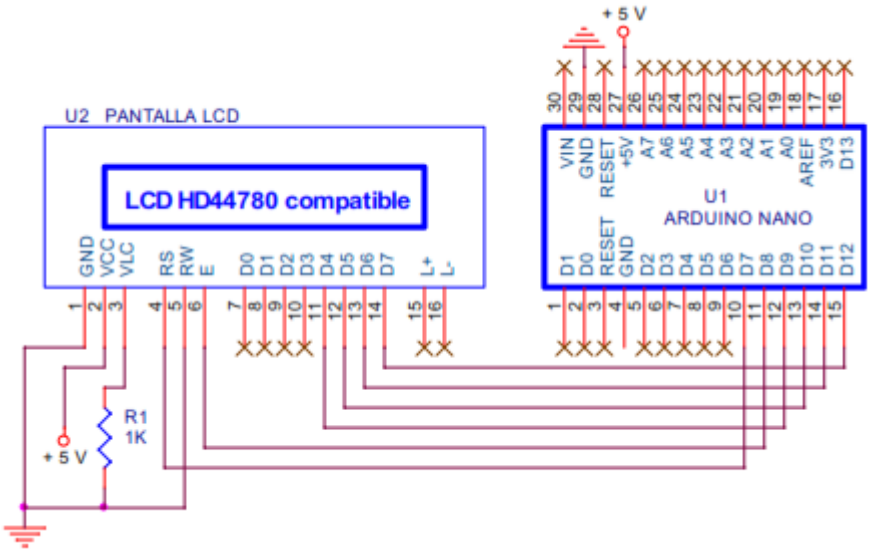
EEPROM.write(982, 33); //Escribir el valor 33 sobre la posición 982

IMPORTANTE: Se estima en unos 100000 los ciclos de escritura posibles sobre unaEEPROM. Cuídalo porque si lo sobrepasas, puede quedar inutilizada. Cada ciclo deescritura puede consumir unos 3,3 mS. Tenlo en cuenta.

PRÁCTICA

7. CONEXIÓN DE LA PANTALLA LCD

Vas a empezar el área de prácticas con la conexión de la pantalla LCD al controlador Arduino sobre el módulo board. Presta atención a las conexiones y procura realizar un buen montaje. La pantalla LCD la vas a mantener prácticamente en todos los ejemplos de las Unidades restantes de este curso. En la figura tienes el esquema eléctrico:



Las patillas D9:D12 de Arduino se conectan con las señales DB4:DB7 de la pantalla. Bajo el control de tu programa Arduino enviará las instrucciones y los caracteres a visualizar. D7 y D8 controlan las señales RS y E respectivamente. RW debe conectarse obligatoriamente con GND.

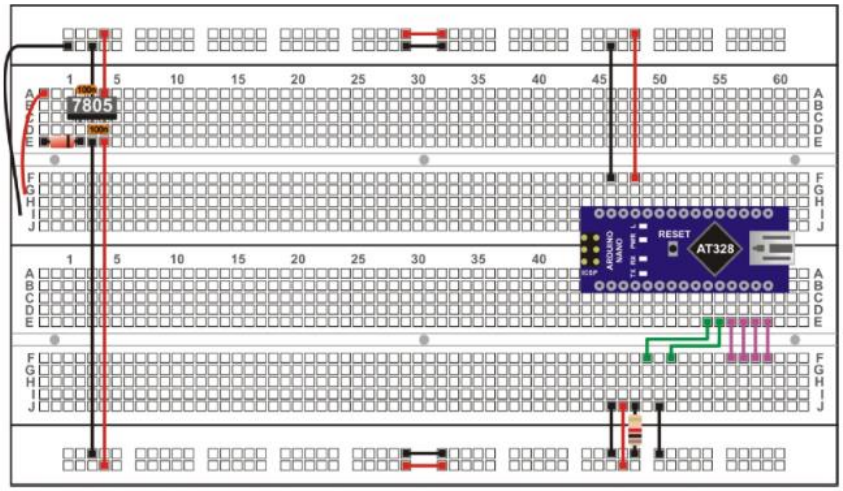
La pantalla se alimenta con +5 V mediante la patilla GND y VCC. Por la patilla VLC, la patilla 3, se debe aplicar una tensión variable entre 0 y 5 V para ajustar el contraste. Se suele conectar con GND a través de una resistencia, R1. El valor de esta resistencia puede variar en función del modelo de pantalla.

Como no se puede prever cuál va a caer en tus manos exactamente, vamos a empezar con una resistencia de 1K Ω. Si observas que los caracteres aparecen demasiado contrastados y

apenas se aprecian, cambia el valor de esta resistencia por otra superior, por ejemplo de 4K7 Ω. Entre estos dos valores seguro que das con el contraste apropiado.

Las patillas 15 y 16 de la pantalla LCD sirven para controlar la luz de fondo, pero esta sólo estará disponible según el modelo. No es nuestro caso. Es posible que tu pantalla ni tan siquiera tenga estas dos patillas y, si las tuviera, las dejarás sin conexión.

Para empezar con el montaje práctico es recomendable que pongas primero los cables de conexión tal y como se muestra en la figura y en la fotografía. La pantalla se insertará después y es posible que tape a alguno de ellos. Fíjate bien.



Ahora coloca la pantalla LCD asegurándote que su patilla 1 se inserta en la misma columna de orificios donde se encuentra el cable negro de alimentación GND, la patilla 2 en la columna del cable rojo de +5 V, la 3 en la de la resistencia, etc….

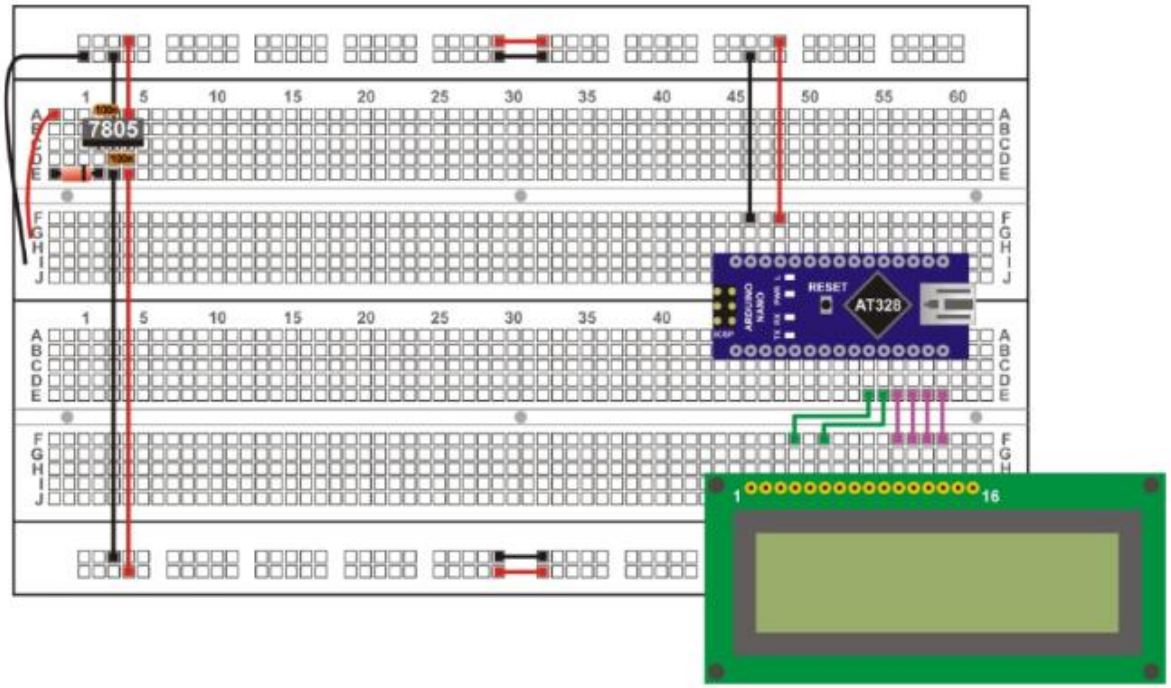


Figura 3

Seguro que se te ocurre otra disposición de los componentes diferente a la expuesta, incluso puede que mejor. Sin embargo debes de tener en cuenta que la pantalla LCD se va a quedar conectada prácticamente en todo lo que queda de curso.

8. EJEMPLO 1: ¡HOLA, MUNDO!

Aquí tienes el primer programa con el que vas a sacar el clásico mensaje sobre la pantalla LCD.

Mediante #include se incluye la librería de funciones del LCD. Con LiquidCrystal() se establece la variable lcd y se configuran las conexiones. Observa que se corresponden con las del esquema eléctrico del montaje que acabas de realizar.

En setup(), con la función lcd.begin(16,2), seleccionas el modelo de pantalla (16 x 2).

Por último, el programa principal loop() transmite el mensaje a visualizar y lo hace mediante la función lcd.print(). A continuación se introduce en un bucle sin fin,while(1);, que no hace nada útil y lo puedes considerar como el fin de la ejecución.

¡¡Qué fácil!! ¿No te lo parece? A partir de ahora vas a disponer de un potente periférico que te permitirá visualizar todo tipo de información. En la fotografía tienes el resultado de cargar y ejecutar este primer ejemplo.

9. EJEMPLO 2: DISPLAY

Con objeto de que te familiarices con algunas de las funciones incluidas en la librería “LiquidCrystal.h”, este y los siguientes ejemplos te van a proponer su empleo para que veas los diferentes efectos que producen.

En este caso vas a emplear las funciones noDisplay() y display() para deshabilitar o activar a la pantalla. Esto provoca que la pantalla quede en blanco o visualice el contenido que tuviera

10. EJEMPLO 3: CURSOR

El cursor indica el lugar de la pantalla donde se escribirá el siguiente carácterque se le envíe. Este ejemplo utiliza las funciones cursor() y noCursor() para conseguirque éste se visualice o no.

En principio el cursor se visualiza como si fuera un guion bajo (_).

11. EJEMPLO 4: BLINK

El cursor también se puede mostrar como un símbolo sólido (▓) intermitente que indica la posición donde se escribirá el siguiente carácter que se envíe a la pantalla. Para ello dispones de las funciones blink() y noBlink().

La intermitencia del cursor es espontánea. La realiza automáticamente la propia pantalla LCD. En el ejemplo, con objeto de comprobar el efecto de estas funciones, se mantiene el cursor activado (y parpadeando) durante 3 segundos, y desactivado otro tanto.

12. EJEMPLO 5: TEXT DIRECTION

Mediante las funciones leftToRight() y rightToLeft() puedes determinar cómo deseas que se vaya escribiendo y visualizando el texto sobre la pantalla. Puedes hacerlo de izquierda a derecha o de derecha a izquierda. Por defecto se escribe de izquierda a derecha, como lo hacemos nosotros habitualmente sobre el papel.

El ejemplo te muestra otra forma de visualizar un mensaje. En esta ocasión el texto se introduce en un array:

```
char Mens1[]={"ILove Arduino"};                    //Mensaje a visualizar
```

Mediante dos bucles for() cada carácter del array se va enviando secuencialmente a la pantalla utilizando la función lcd.write(), a intervalos de 0.150”.En el primer bucle for() el cursor se coloca en la posición 0 de la fila 0 (la primera delLCD) y se van escribiendo de izquierda a derecha, lo normal. En el segundo bucle for()el cursor se pone en la posición 15 de la fila 1, la última posición de la segunda, y los caracteres se van escribiendo de derecha a izquierda.

Luego se realiza una temporización de 3” y la pantalla se borra con la función lcd.clear(). Tras una nueva temporización de 1” el ciclo se vuelve a repetir.

En la siguiente fotografía tienes el resultado de ejecutar este ejemplo. ¿Lo entiendes? Piensa en los efectos de visualización que puedes conseguir.

13. EJEMPLO 6: SCROLL

Con las funciones scrollDisplayLeft() y scrollDisplayRight() también puedes conseguir unos bonitos efectos de visualización que, seguro, has visto en carteles publicitarios.

Estas funciones desplazan a izquierda o derecha todo lo que en ese momento se esté visualizando en la pantalla. En este ejemplo se visualiza un texto en las dos líneas de la misma. Luego todo el texto se va desplazando a la izquierda, posición a posición, ya intervalos de 0.3".

Cuando el último carácter desaparece por la izquierda, el texto se vuelve a desplazar hasta que desaparece por la derecha. Finalmente, desplazando a la izquierda, todo el texto se vuelve a dejar en la posición original y se realiza una secuencia de intermitencia activando/desactivando la pantalla.

14. EJEMPLO 7: AUTOSCROLL

Otro ejemplo que te permitirá crear llamativos efectos de visualización. En esta ocasión, mediante las funciones autoscroll() y noAutoscroll() se habilita o no el desplazamiento automático del contenido de la pantalla.

Cuando se activa el auto scroll, el desplazamiento se realiza automáticamente cada vez que se escribe un carácter en la pantalla. En el ejemplo anterior tenías que emplear expresamente las funciones scrollDisplayLeft() o scrollDisplayRight().

Este desplazamiento automático también puede ser a la izquierda (por defecto)o a la derecha. Todo depende de qué función leftToRight() o RightToLeft() es la que has usado por última vez.

En el ejemplo se visualizan dos mensajes que están previamente definidos en sendos arrays:

```
char Mens1[]={"ARDUINO"};          //Mensaje 1

char Mens2[]={"I'Love you"};       //Mensaje 2
```

Un primer bucle for() va escribiendo sobre la pantalla, carácter a carácter, el contenido del array “Mens1”. Por defecto y, de forma natural, estos se escriben de izquierda a derecha a intervalos de 0.3”.

El segundo bucle for() va escribiendo en la pantalla, carácter a carácter, el contenido del array “Mens2”. Sin embargo, antes de ejecutar este bucle, el cursor se coloca en la última posición de la segunda fila de la pantalla, y se ejecuta la funciónlcd.autoscroll(). Los caracteres se van escribiendo de izquierda a derecha, pero porcada uno de ellos, se produce un desplazamiento automático a la izquierda (por defecto) del contenido actual de la pantalla.

Finalmente se ejecuta lcd.noAutoscroll() y, tras borrar la pantalla mediante lcd.clear(), el ciclo se vuelve a repetir. Bonito ¿no?

15. EJEMPLO 8: CARACTERES PERSONALIZADOS

Para terminar con estos ejemplos que nos han ido mostrando el uso de las funciones más relevantes de la librería “LiquidCrystal.h” vas a crear y utilizar tus propios caracteres gráficos.

Concretamente vas a crear 5 caracteres: un corazón, un rostro alegre, otro serio, un muñeco con los brazos hacia abajo y otro con los brazos hacia arriba.Cada uno de ellos se define mediante un array. Así que tienes estos cinco: corazon[8] ; alegre[8]; serio[8]; abajo[8] y arriba[8]. El contenido de cada uno deellos describe en binario qué puntos o pixels se debe activar en cada caso.

En el setup(), mediante funciones lcd.createChar() se transfiere a la memoria CGRAM de la pantalla el contenido de los arrays. En total se transfieren 40 bytes, 8por cada carácter. Además, a cada uno de ellos se le asigna un número del 0 al 5:

```
lcd.createChar(0,corazon);          // Crea el nuevo carácter nº 0
lcd.createChar(1,alegre);           //Crea el nuevo carácter nº 1lcd.createChar(2,
serio);                             //Crea el nuevo carácter nº 2lcd.createChar(3,
```

abajo); //Crea el nuevo carácter nº 3**lcd.createChar(4,**
arriba); //Crea el nuevo carácter nº 4

Por último cada carácter gráfico se puede visualizar en la pantalla, en la posición actual del cursor, como si de un carácter ASCII normal se tratara. Basta emplear la función lcd.write(n), donde n es el número asignado a cada uno, en el ejemplo del 0 al5.

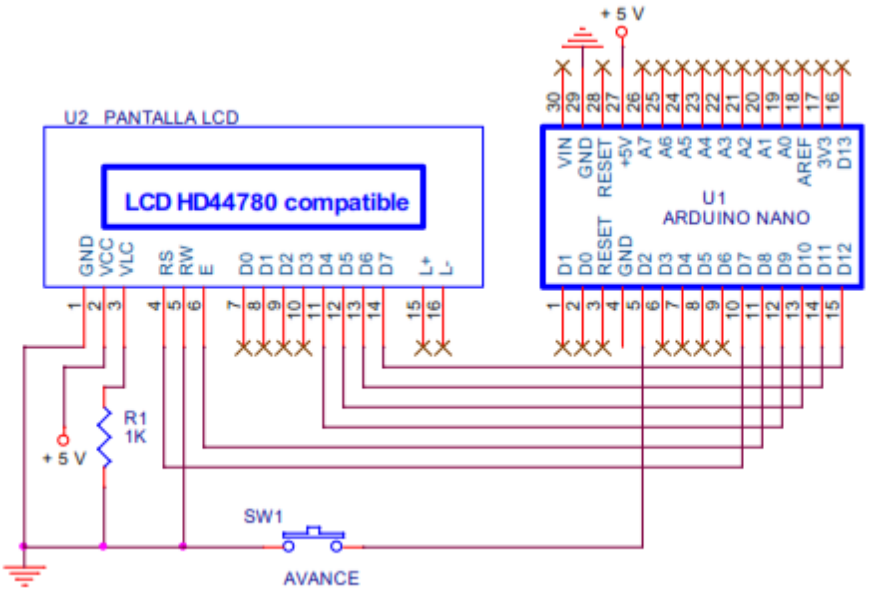
16. EJEMPLO 9: DISPLAY

Ejemplo con un indudable interés práctico. Vas a crear un nexo de unión entre el canal serie de tu ordenador PC y la pantalla LCD. En medio se encuentra, como no, el controlador Arduino.

El Arduino hace de puente. Por un lado se conecta mediante el interface USB con el PC. Por otro lado se conecta con la pantalla como se ha venido haciendo hasta ahora. Desde el PC, empleando en monitor serial, vas a transmitir vía serie un conjunto de caracteres que serán recibidos por el Arduino y visualizados en la pantalla LCD según van llegando.

17. EJEMPLO 10: VISUALIZANDO NÚMEROS ENTEROS

Ya has visto que sobre la pantalla LCD puedes visualizar todo tipo de textos y caracteres, pudiendo además realizar diferentes efectos de visualización. Por supuesto que también puedes visualizar números. Este ejemplo va a ir visualizando los números del 0 al 15 secuencialmente, cada vez que se accione un pulsador conectado en la patilla D2. Además, el número en cuestión se visualizará en decimal, hexadecimal y en binario. Aquí tienes el esquema eléctrico con la conexión entre Arduino NANO, la pantalla y el pulsador SW1.



Con tus conocimientos actuales, el programa no te resultara complejo en absoluto. Un pequeño detalle que seguro lo usarás con frecuencia. Como los números a visualizar tienen distinta longitud, una buena idea es borrar totalmente la línea de la pantalla LCD donde se van a visualizar. Esto se puede hacer completando con espacios en blanco (“ “) la línea en cuestión. Así no quedan “restos”.

Analiza el programa. Prueba a retirar las dos primeras funciones de loop() y observa lo que pasa. Seguro que ahora entiendes a qué me refiero con eso de los “restos”.

18. EJEMPLO 11: VISUALIZANDO NÚMERO FLOAT

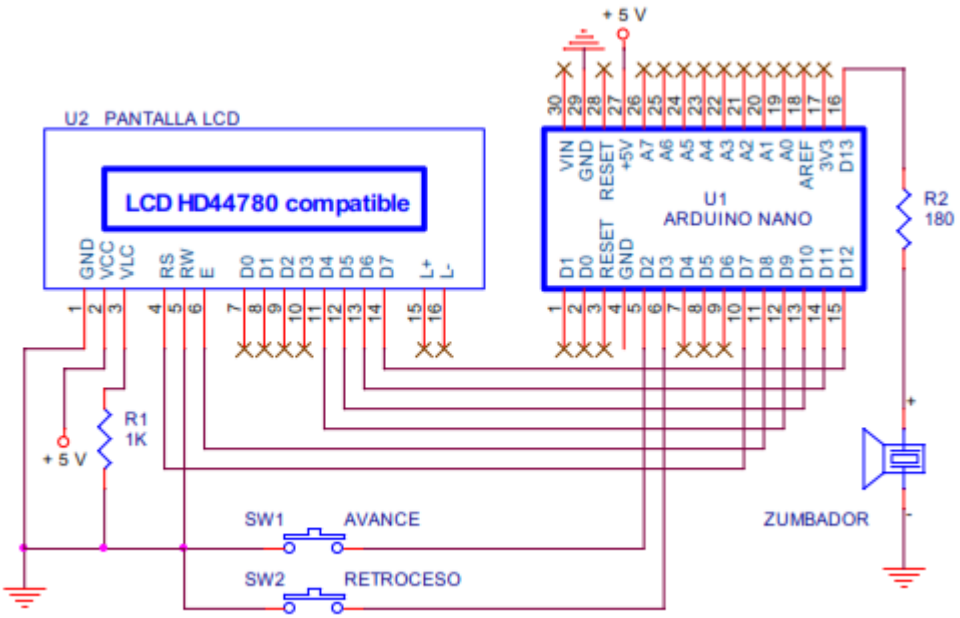
Siguiendo la línea del ejemplo anterior, ahora se trata de visualizar números float. Concretamente se va a visualizar con dos decimales el resultado de la raíz cuadrada del número N y, con cuatro decimales, el producto de N por la constante PI(3.1416). El número N va avanzando secuencialmente con cada pulsación del pulsador en D2 y evoluciona de 0 al 9.

Tanto el esquema eléctrico como el del montaje práctico es el mismo que el que empleaste en el ejemplo 3-10 anterior.

19. EJEMPLO 12: MENÚ

La pantalla LCD es un periférico ideal para hacer un intuitivo interface con el usuario. Efectivamente, puedes visualizar menús de opciones que te permitan seleccionar entre las diferentes tareas a ejecutar.

En este ejemplo se trata de realizar un menú de 6 opciones que van apareciendo secuencialmente sobre la pantalla. Para ello, mediante dos pulsadores de avance y retroceso, podrás navegar por el menú para visualizar todas las opciones disponibles. Este es el esquema eléctrico:



Las conexiones entre el Arduino y la pantalla son las mismas que las empleadas hasta ahora. En esta ocasión se emplea un segundo pulsador, SW2. Con SW1conectado con la entrada D2, se van avanzando las opciones que se visualizan sobre la pantalla. Con SW2 conectado en la patilla D3 se va retrocediendo. Ambas patillas se configuran como entradas pull-up. Te ahorras tener que poner externamente las dos resistencias correspondientes.

La pantalla realiza un efecto de “scroll vertical” para visualizar secuencialmente todas las opciones disponibles en el menú.

También se emplea un zumbador piezoeléctrico conectado con la patilla de salida D13. Cualquier pulsación tanto en SW1 como en SW2 generará una señal acústica. Aquí tienes el montaje práctico.

Ahora vas a echar un vistazo al programa. El tratar de estudiarlo y analizarlo para comprender su funcionamiento lo dejo en tus manos. Simplemente decirte que seha procurado dar una solución lo más sencilla y didáctica posible. Seguramente no será la única ni la mejor. Espero que tú, con tu iniciativa, la mejores notablemente.

Inicio:

Al principio del todo se incluye la librería “LiquidCrystal.h” y se establecen las conexiones con la pantalla. No hay nada nuevo. Se declaran las variables “Opcion” y “N_opciones”. La primera contiene la opción en curso seleccionada y la segunda el número de opciones que tiene el menú. También se declara un array llamado “flecha” que define un carácter gráfico.

Funciones:

En el proyecto se han creado dos funciones: visualiza() y beep(). La primera visualizaen la pantalla el mensaje correspondiente a la opción en curso actual según el valor dela variable “Opcion”. La función beep() no es nueva. Emite una señal acústica.

Setup():

Configura las patillas D2 y D3 como entradas con Pull-Up interna, la patilla D13 como salida, la pantalla LCD con 2 líneas de 16 caracteres y por último genera el carácter gráfico nº 0 (flecha). Tampoco hay nada especialmente relevante.

loop():

- Empieza borrando la pantalla y visualizando el carácter gráfico nº 0 (la flecha).
- Coloca el cursor en la primera fila y visualiza la opción en curso según la variable "Opcion". Hace uso de la función visualiza().
- Coloca el cursor en la segunda fila y visualiza la siguiente opción (Opcion+1).Nuevamente hace uso de la función visualiza().
- Espera a que se pulse uno de los dos pulsadores SW1 o SW2 conectados en D2y en D3.
- Si se pulsa D2 se eliminan rebotes y se espera a que se suelte. Genera un pitido e incremente la variable "Opción". Se apunta a la siguiente opción del menú.
- Si se pulsa D3 se eliminan rebotes y se espera a que se suelte. Genera un pitido y decrementa la variable "Opción". Se apunta a la opción previa del menú.
- Grábalo y comprueba su funcionamiento ¿te gusta? Pues espero que esto te sirva como idea para un buen número de futuros proyectos.

20. EJEMPLO 13: SELECCIÓN DE OPCIONES

Considéralo una prolongación del ejemplo anterior. Mira el esquema eléctrico.

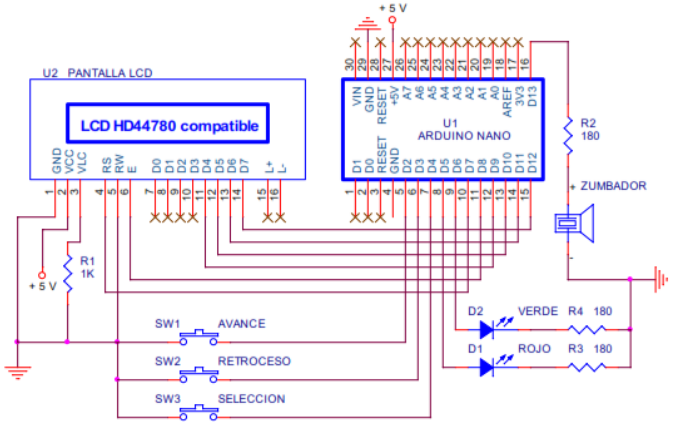


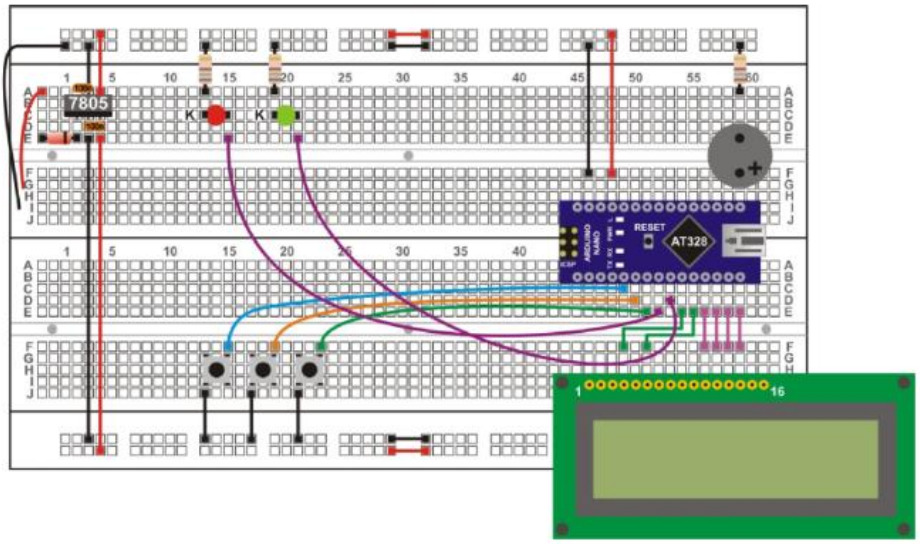
Figura 4

Se añade un pulsador más, el SW3 conectado en D4, con el que se ejecuta laopción seleccionada en ese momento. También se añaden dos leds, uno rojo y otroverde conectados en D5 y D6, que permiten ver el resultado de ejecutar esa opción.Aquí tienes el montaje práctico.

El programa es muy parecido al del ejemplo anterior. Únicamente se ha añadido la función Ejecuta() que se ejecutará cada vez que se pulse el pulsador SW3. Esta nueva función evalúa el valor de la variable “Opcion”. Según cual sea la opción actualmente seleccionada, se ejecuta la tarea apropiada y consistente en activar o desactivar los leds rojo y verde conectados con las salidas D5 y D6.

Si estudiaste y entendiste el programa del ejemplo anterior, este no te debe suponer problema alguno ¿verdad?.

21. EJEMPLO 14: UNA ÚLTIMA MEJORA



Imagina una aplicación en la que el menú no sólo tiene 6 opciones como hasta ahora. Tiene varias decenas de ellas. Cada vez que enciendes el sistema o generas un reinicio mediante

RESET, siempre empieza desde la primera opción y deberás desplazarte hasta localizar la que deseas.

Quizá podría interesarte que cada vez que se reinicie, la pantalla LCD muestre la última opción que hubieras seleccionado y no tengas que andar buscándola. ¿Qué te parece?

Nuevamente tienes la posibilidad de usar la memoria EEPROM de datos. Efectivamente, cada vez que se selecciona una nueva opción, esta se registra en la memoria EEPROM. Al fin y al cabo cada opción se representa ni más ni menos que con un número guardado en la variable “Opcion”.

Cada vez que se reinicie el sistema, esa variable se carga con el valor registrado previamente en la memoria EEPROM. En este ejemplo vas a usar la posición 1 de las1024 disponibles.

Graba el programa y comprueba su funcionamiento. Selecciona y ejecuta diferentes opciones. Luego apaga el sistema y vuelve a encenderlo. Observa que siempre aparece la última opción que seleccionaste.

Es de esperar que estos ejemplos te sugieran un buen número de aplicaciones. Ánimo y emplea tu imaginación e ingenio.

REFERENCIAS

LIBROS

[1]. **EXPLORING ARDUINO**, Jeremy Blum, Ed.Willey
[2]. Practical ARDUINO, Jonathan Oxer & Hugh Blemings, Ed.Technology in action
[3]. Programing Arduino, Next Steps, Simon Monk
[4]. Sensores y actuadores, Aplicaciones con ARDUINO, Leonel G.Corona Ramirez, Griselda S. Abarca Jiménez, Jesús Mares Carreño, Ed.Patria
[5]. ARDUINO: Curso práctico de formación, Oscar Torrente Artero, Ed.RC libros
[6]. 30 ARDUINO PROJECTS FOR THE EVIL GENIUS, Simon Monk, Ed. TAB
[7]. Beginning C for ARDUINO, Jack Purdum, Ph.D., Ed.Technology in action
[8]. ARDUINO programing notebook, Brian W.Evans

PÁGINAS WEB

[1]. https://www.arduino.cc/
[2]. https://www.prometec.net/
[3]. http://blog.bricogeek.com/
[4]. https://aprendiendoarduino.wordpress.com/
[5]. https://www.sparkfun.com