Manexo de LCD´s

Librerías útiles:

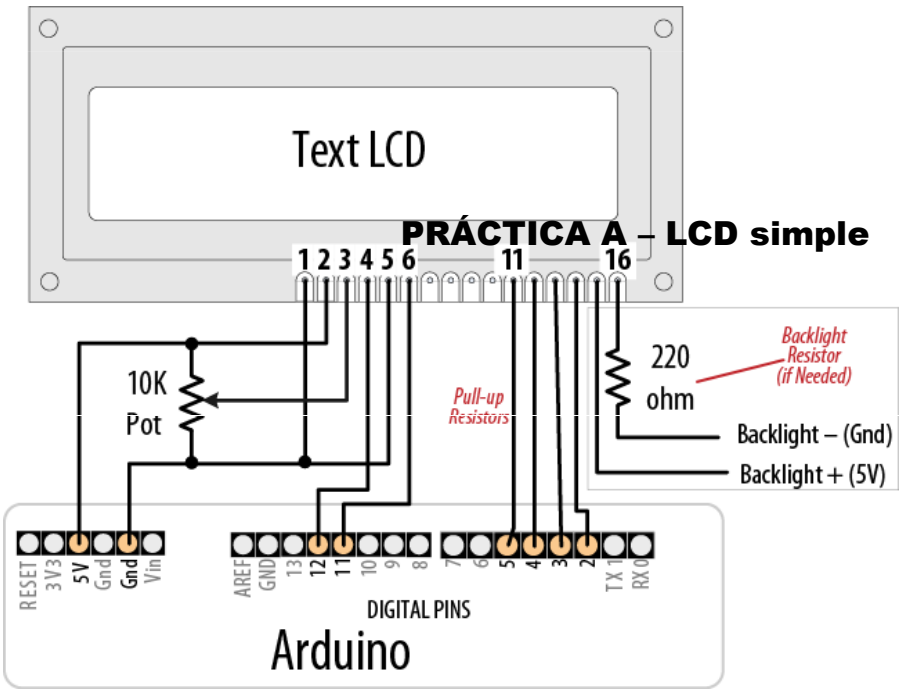
-LiquidCrystal.h

-SoftwareSerial.h

-Tamén existen pantallas gráficas con librerías non oficiais.

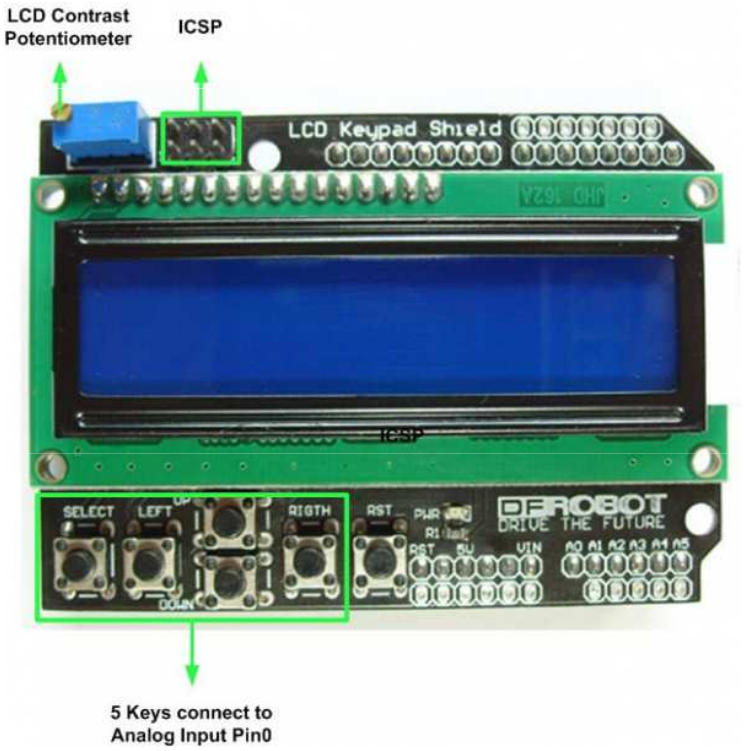
LCD con controlador Hitachi HD44780

- **Conexionado exemplo Arduino:**



LCD pin	Function	Arduino pin
1	Gnd or 0V or Vss	Gnd
2	+5V or Vdd	5V
3	Vo or contrast	
4	RS	12
5	R/W	
6	E	11
7	D0	
8	D1	
9	D2	
10	D3	
11	D4	5
12	D5	4
13	D6	3
14	D7	2
15	A or analog	
16	K or cathode	

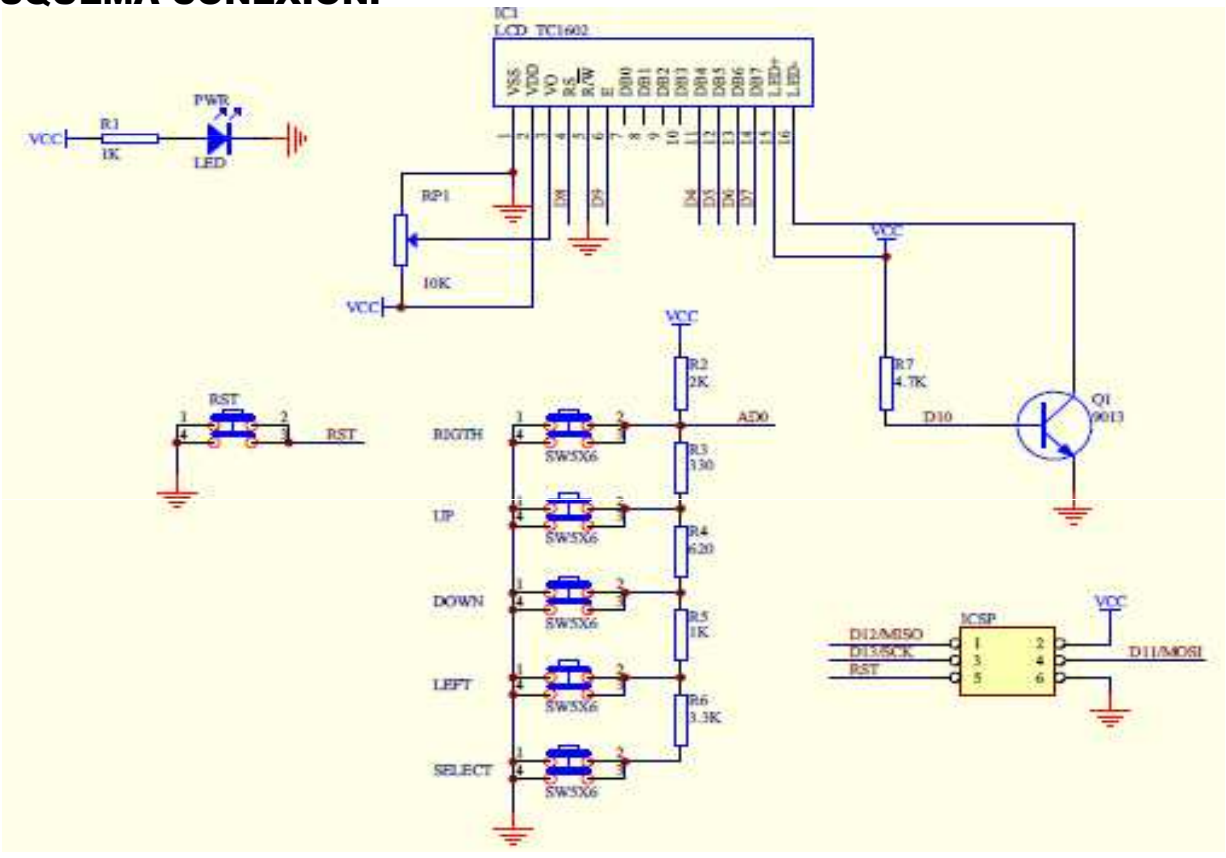
Display con botonera DF-Robot



Pin	Function
Analog 0	Button (select, up, right, down and left)
Digital 4	DB4
Digital 5	DB5
Digital 6	DB6
Digital 7	DB7
Digital 8	RS (Data or Signal Display Selection)
Digital 9	Enable
Digital 10	Backlit Control

Display con botonera DF-Robot

- ESQUEMA CONEXIÓN:



PRÁCTICA A_1 – Manexo LCD estándar

- **Exemplo: LiquidCrystal - Hello World:**

```
#include <LiquidCrystal.h>       //Libreria para o control de display LCD.

LiquidCrystal lcd(8, 9, 4, 5, 6, 7);     // Asignación de pins do display

void setup() {
    lcd.begin(16, 2);               // Indicacion do numero de filas e columnas do display.
    lcd.print("Hola Mundo!");      // Mensaxe de benvida
}

void loop() {

  lcd.setCursor(0, 1);  // Posiciona o cursor ao principio da segunda liña (column 0, line 1)

  lcd.print(millis()/1000); // Contador dos segundos que trascurriron dende o encendido.
}
```


PRÁCTICA A_2 – Lectura botonera analóxica

- **Código a engadir:**

```
int botPin=0;                 //Declaracións
int botonera;                  // Botonera conectada a AN0

void loop() {
  lcd.clear();                 //Limpa pantalla
  lcd.setCursor(0,0);   // Posiciona en (column 0, line 0)
  lcd.print("Valor botonera AN0=");
  lcd.setCursor(0, 1);  // Posiciona en (column 0, line 1)
  botonera=analogRead(botPin);
  lcd.print(botonera);  //Amosa valor entrada AN0 asociada
  delay(1000);
}
```

PRÁCTICA A_3 – Aplicación selección botonera

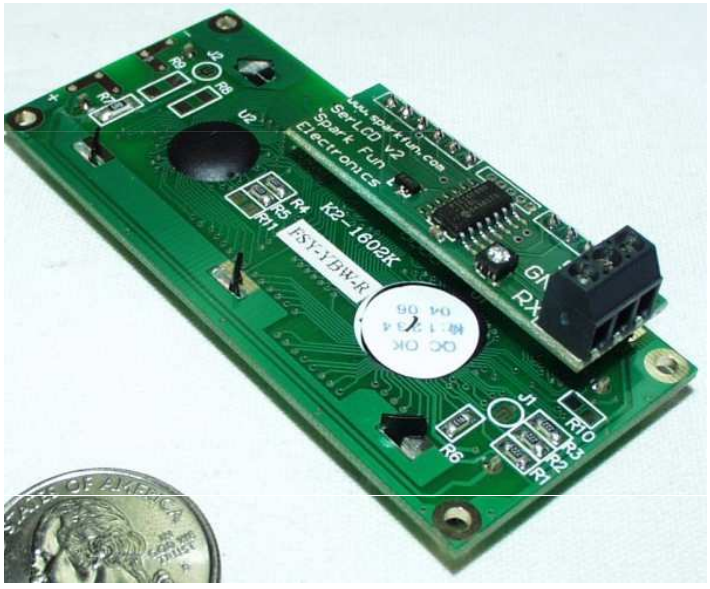
- **DESCRICIÓN:**

  Programa que amose na pantalla o nome dun ciclo formativo da familia profesional segundo o botón que pulsemos.

- **Referencias:**

  Empregar condicións asociadas ao valor analóxico da botonera.

LCD SERIAL Sparkfun



-Coas pantallas estándar empréganse 6 pins de datos para controlar a pantalla LCD.

-O modelo Sparkfun SerLCD incorpora un controlador serie que permite xestionar a pantalla con só 1 pin de datos.

Titorial: https://www.sparkfun.com/tutorials/246

LCD SERIAL Sparkfun

Nome do sinal	Especificación	color del cable (cable suministrado)
RX (recepción)	Recepción serie (entrada á pantalla). 5V (nivel TTL), 9600bps (pódese cambiar), 8 bits, 1 stop, sin paridad.	Amarelo
GND (masa)	Masa	Negro
VDD (alimentación)	5 V cun consumo de ata 60 mA, coa luz de fondo a plena potencia.	Vermello

Debe terse en conta que a entrada RX debe ser un sinal de nivel TTL 5V directamente dende un microcontrolador ou otro sistema de 5V. **NON se debe conectar a placa a voltaxes de nivel RS232 , que son + /-10V e pode danar a pantalla**

Librería “SoftwareSerial.h”

-Permite emular un canle de comunicación serie con calquera pin dixital de Arduino.

-En lugar de manexalo unha UART físicamente realízase por software, polo que pode sernos últil cando precisamos máis portos serie.

-As funcións máis salientables son:

SoftwareSerial meuSerial (pin RX,pin TX); //Declaración dun porto serie virtual asociado aos pins de transmisión/recepción dados.

meuSerial.begin (9600); //Indicación da velocidade de transmisión

meuSerial.Write(---); //Enviar a través do porto serie.

Comandos LCD Serial

- **Posicionamento:**

  Para situarse nunha posición determinada debe enviarse o carácter núm. 254 (0xFE) e a continuación o carácter indicado na seguinte táboa

posición	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Liña 1	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
Liña 2	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207

- **Envío variables numéricas:**

Só permite o envío de texto, polo que é necesaria unha conversión:

char cadea[16]; // Creación dun string para almacenar cadeas

sprintf (cadea, "--- %d --- " , variable); //Inserción do valor da variable nun string

mySerial.write(cadea); //Envío da cadea completa

PRÁCTICAS B-1 – LCD Serial "Hola Mundo"

```
#include <SoftwareSerial.h>

// Asociación aos pins RX/TX
SoftwareSerial mySerial(3,2); // pin 2 = TX, pin 3 = RX (non se usará)

void setup()
{
  mySerial.begin(9600); //  Incialización a 9600 baudios
  delay(500); // Espera para darlle tempo a arrancar ao display
}

void loop()
{
  mySerial.write(254); // move o cursor ao principio da primeira liña
  mySerial.write(128);

  mySerial.write("                "); // limpa display
  mySerial.write("                ");

  mySerial.write(254); // move o cursor ao principio da primeira liña
  mySerial.write(128);
  mySerial.write("Primeira linha!");

  mySerial.write(254); // move o cursor ao principio da segunda liña
  mySerial.write(192);
  mySerial.write("Segunda linha!");

  while(1); // Espera infinita
}
```

PRÁCTICAS B-2 – Envío valores numéricos

```
#include <SoftwareSerial.h>

// Asociación aos pins RX/TX
SoftwareSerial mySerial(3,2); // pin 2 = TX, pin 3 = RX (non se usará)
char cadea[16];   // String auxiliar para compoñer cadeas

void setup()
{
  mySerial.begin(9600); //  Incialización a 9600 baudios
  delay(500); // Espera para darlle tempo a arrancar ao display
}

void loop()
{
  mySerial.write("                "); // Borra display
  mySerial.write("                ");

  mySerial.write(254); // Move o cursor ao principio da primeira liña
  mySerial.write(128);
  mySerial.write("Quedanlle...");

  for (int s=10;s>0;s--)        // Conta atrás segundeiro
  {
    sprintf (cadea, "% 2d segundos" , s);   //Insercion do valor nunha cadea
    mySerial.write(254); // move o cursor ao principio da segunda liña
    mySerial.write(192);
    mySerial.write(cadea);
    delay (1000);
  }
```


PRÁCTICAS B-3 – Semáforo e LCD serial

- **EXERCICIO:**

Programa que manexe un semáforo que:

- **Permita o paso alternativo nos dous sentidos cada 5 segundos.**
- **Cando un peón pulse un botón, bloqueará a circulación de vehículos e daralle paso aos peóns durante 10 segundos. Deberá indicar na pantalla a secuencia:**
  - **✓Peón, pode pasar....**
  - **✓Quédanlle xx segundos**
  - **✓¡ Detéñase !**

Función millis()

- **millis():**

  Devolve un valor de tipo “unigned long” co tempo (ms) dende que se iniciou o programa. A variable desborda aproximadamente nuns 50 días.

- **Problema:**

  - As temporizacións con delay() impiden que se sigan facendo outras tarefas mentres se espera.

  - Empregando a función millis() podemos relizar temporizacións sen necesidade de bloquear o proceso.

  Tutorial: http://arduino.cc/en/Tutorial/BlinkWithoutDelay

Uso de millis() para temporizacións

```
const int ledPin =  13;
unsigned long inicio=0;   //Variable para almacenar o momento no que se inicia a temporización
boolean ledState;   //Flag auxiliar para encender/apagar o LED como báscula
long temporizacion = 1000;           // intervalo de temporización (milisegundos)

void setup() {
  // set the digital pin as output:
  pinMode(ledPin, OUTPUT);
}

void loop()
{
  ////////////////////////////////////////////////////////////////////
  // Aquí estaría o codigo que se debe executar simultameamente//
  ////////////////////////////////////////////////////////////////////

  if( millis() - inicio > temporizacion) { //Se pasou o tempo...

    inicio = millis();    // Reiniciamos a conta

    if (ledState == LOW) ledState = HIGH;
    else ledState = LOW;    // Facemos bascular ledState

    digitalWrite(ledPin, ledState);  // Aplicamos a saída
  }
}
```