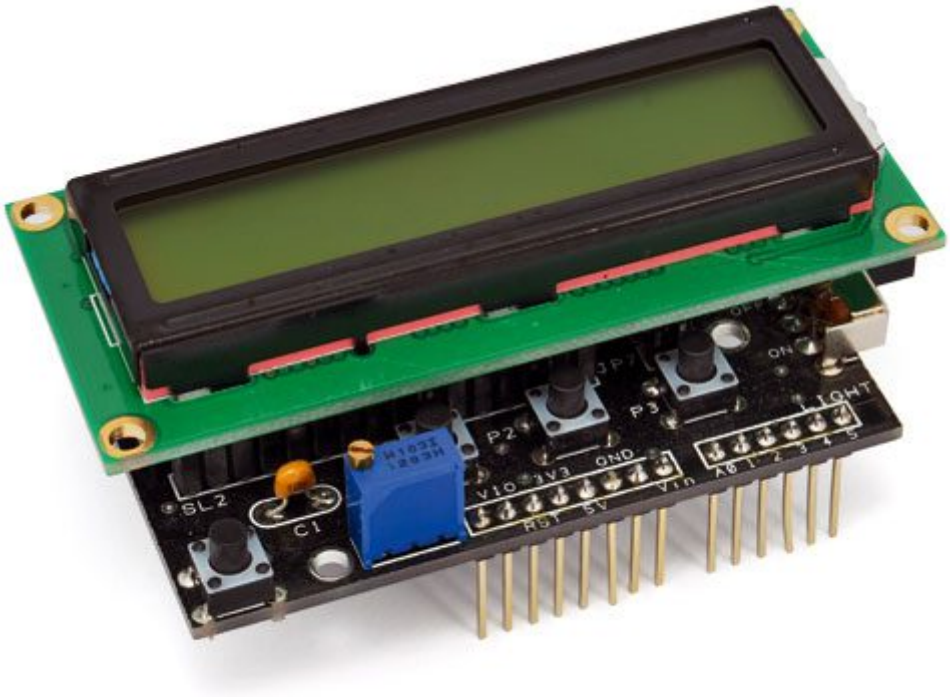
SHIELD LCD PER ARDUINO CON DISPLAY - IN KIT

Prezzo: 19.67 €

Tasse: 4.33 €

Prezzo totale (con tasse): 24.00 €



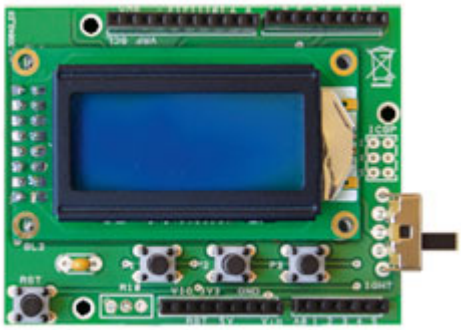
Shield per Arduino Uno Rev3 che permette di montare la quasi totalità dei display dotati del chip Hitachi HD44780 (o compatibile). Per controllare i display non serve conoscere protocolli dati o altro, dato che la gestione di tutte le linee viene demandata alla libreria disponibile nell'ambiente di sviluppo (IDE) di Arduino. La shield dispone di tre diversi tipi di connettori a seconda del display utilizzato, tre pulsanti generici, un pulsante di reset e un interruttore che vi dà la possibilità di inserire o disinserire la retroilluminazione. I contatti della scheda Arduino sono sempre disponibili dal lato di alimentazione, mentre la disponibilità dei contatti digitali (da 0 a 13) è assicurata solo per i display più piccoli. Compatibile con altre shield come la MOTORSHIELD_FE, l'SDCARDSHIELD e la WIFI_SHIELD.

N.B. La shield viene fornita con il display codice 1446-ACM1602B-FL-YBW.

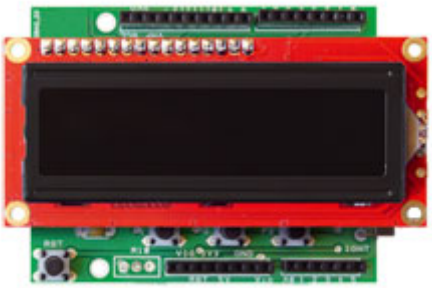
LCD SHIELD CON ALCUNI DISPLAY

Indipendentemente dal tipo di connettore adottato dal costruttore e da come sono disposti i pin, tutti i display basati su controller HD44780 o compatibile presentano la medesima interfaccia, composta da otto bit di dati, tre linee di controllo, l'alimentazione +Vcc e GND ed una linea per la regolazione del contrasto; tuttavia la disposizione delle linee cambia da modello a modello, ragion per cui prima di usare un display occorre consultarne il data-sheet. In alcuni modelli è previsto un retroilluminatore a LED, alimentato tramite due contatti del connettore.

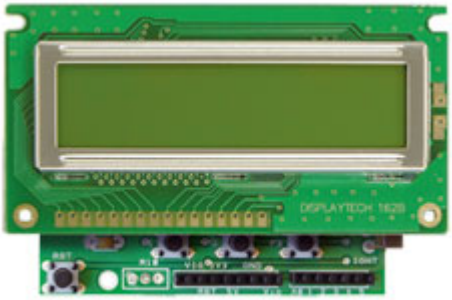
LCD shield con display LMB0820 (alfanumerico 8x2)
Codice Futura Elettronica 1446-LCD8X2BN



LCD shield con display ADM1602K (alfanumerico 16x2)
Codice Futura Elettronica 1446-LCD16X2WB
oppure 2846-LCD1602A



LCD shield con display DISPLAYTECH162B (alfanumerico 16x2)
Codice Futura Elettronica 1446-CDL4162



LCD shield con display ACM1602B (alfanumerico 16x2)
Codice Futura Elettronica 1446-ACM1602B-FL-YBW

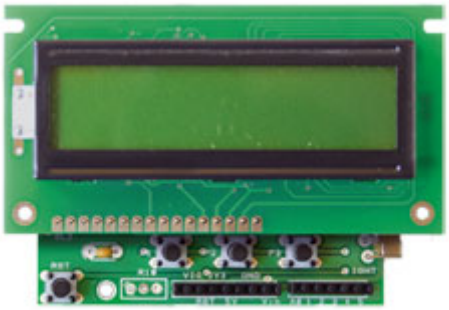


Tabella 1 - Assegnazione pin display LCD ad Arduino

PIN DISPLAY	FUNZIONE	PIN DI ARDUINO
VDD	Alimentazione +5volt	+5V
Vss	Alimentazione GND	GND
Vo	Tensione per il contrasto	Trimmer
RS	Selezione scrittura di dati o comandi	8
R/W	Abilita la lettura o la scrittura all'interno del display	GND
Enable	Linea di abilitazione	9
DB0	Linea dati 0	Non usato
DB1	Linea dati 1	Non usato
DB2	Linea dati 2	Non usato
DB3	Linea dati 3	Non usato
DB4	Linea dati 4	4
DB5	Linea dati 5	5
DB6	Linea dati 6	6
DB7	Linea dati 7	7
BL+	Retroilluminazione LED+	+5V
BL-	Retroilluminazione LED-	GND

Tabella 2 - Contatti dello shield assegnati ai pulsanti

PULSANTE	PIN DI ARDUINO (MODALITÀ DIGITALE)	PIN DI ARDUINO (MODALITÀ ANALOGICA)
P1	10	AN3
P2	11	AN3
P3	12	AN3

Tabella 3 - Modalità analogica di lettura dei pulsanti

PULSANTE PREMUTO	TENSIONE AN3	VALORE ADC
Nessuno	5v	1024
P1	0v	0
P2	1,6v	328
P3	3,3v	676

Tabella 4 - Configurazione motorshield_FE

JUMPER MOTORSHIELD_FE	PIN DI ARDUINO
PWMA	3
PWMB	11
DIRA	2
DIRB	12

Tabella 5 - Esempi disponibili nella libreria LiquidCrystal

AUTOSCROLL	DIMOSTRAZIONE DELLE FUNZIONI AUTOSCROLL() E NOAUTOSCROLL() DA APPLICARE PER LE SCRITTE SCORREVOLI
Blink	Esempio nel quale si abilita e disabilita il lampeggio del cursore
Cursor	Esempio nel quale si abilita e disabilita la visualizzazione del cursore
CustomCharacter	Esempio di utilizzo di caratteri personali
Display	Dimostrazione delle funzioni display() e noDisplay() per attivare o disabilitare la visualizzazione.
HelloWorld	Visualizza la scritta Hello World ed il valore di un contatore
Scroll	Visualizza la scritta "Hello World!" ed utilizza le funzioni scrollDisplayLeft() e scrollDisplayRight() per farla scorrere.
SerialDisplay	Visualizza i caratteri inviati da PC ad esempio con SerialMonitor
SetCursor	Come impostare la posizione del cursore sul display
TextDirection	Dimostrazione delle funzioni leftToRight() and rightToLeft() per muovere il cursore.

SKETCH DI ESEMPIO (clicca sulle immagini per ingrandire)

Per quanto riguarda la programmazione attraverso l'IDE di Arduino, non ci sono particolari difficoltà: non è necessaria una specifica libreria in quanto quella di sistema, denominata liquidCrystal, comprende già tutte le necessarie funzioni; vediamole di seguito, istruzione per istruzione.

- **LiquidCrystal lcd(rs, enable, d4, d5, d6, d7)**; inizializza il display e lo dichiara come oggetto di nome "lcd" usato successivamente per la chiamata alle funzioni della libreria. Devono essere specificati i pin utilizzati da Arduino per comandare il display, quattro linee di dati e due linee di controllo.
- **lcd.begin(cols, rows)**; imposta per l'oggetto "lcd" il numero di colonne (cols) ed il numero di righe (rows). Per il display LMB0820 bisogna specificare i valori 8 e 2, mentre per tutti gli altri, i valori 16 e 2
- **lcd.clear()**; cancella il contenuto del display.• lcd.home(); posiziona il cursore all'inizio della prima riga e prima colonna.
- **lcd.setCursor(thisRow, thisCol)**; posiziona il cursore nella riga thisRow (0 è la prima riga) e nella colonna thisCol (0 è la prima colonna). Ad esempio, per posizionare il cursore a metà della seconda riga, in un display 16x2, si scriverà lcd.setCursor(1,8);.• lcd.write(data); scrive un carattere sul display fornendone il relativo codice ASCII.• lcd.print("testo"); scrive un testo sul display a partire dalla posizione attuale del cursore.
- **lcd.print(var)**; scrive il valore della variabile "var" sul display a partire dalla posizione attuale del cursore. Se la variabile è un float, verranno scritte tutta la parte intera e due sole cifre della parte decimale.• lcd.cursor(); attiva la visualizzazione del cursore sulla posizione attuale.• lcd.noCursor(); disattiva la visualizzazione del cursore sulla posizione attuale.• lcd.blink(); attiva il lampeggio del cursore (se visualizzato).• lcd.noBlink(); disattiva il lampeggio del cursore (se questo è visualizzato).• lcd.display(); attiva la visualizzazione sul display.
- **lcd.noDisplay()**; disattiva la visualizzazione sul display.• lcd.scrollDisplayRight(); trasla il testo del display di una posizione a destra.• lcd.scrollDisplayLeft(); trasla il testo del display di una posizione a sinistra.
- **lcd.autoscroll()**; abilita lo spostamento automatico del cursore ad ogni inserimento di un nuovo carattere.• lcd.noAutoscroll(); disabilita lo spostamento automatico del cursore ad ogni inserimento di un nuovo carattere.
- **lcd.leftToRight()**; imposta la direzione nella quale sarà inserito il prossimo carattere. Per impostazione predefinita il carattere sarà aggiunto
  a destra dell'ultimo carattere inserito.
- **lcd.rightToLeft()**; imposta la direzione nella quale sarà inserito il prossimo carattere. Il carattere sarà aggiunto a sinistra dell'ultimo carattere inserito.
- **lcd.createChar()**; permette di creare un carattere speciale non incluso nel set base dei caratteri del display. È possibile creare sino ad otto caratteri speciali (da 0 a 7) con una risoluzione di 8x5 pixel. Il nuovo carattere viene specificato con un vettore di byte contenente i valori dei pixel che dovranno essere visualizzati. La funzione write sarà usata per la visualizzazione dei caratteri personali (indice da 0 a 7).Tutti gli esempi disponibili con la libreria sono compatibili con il nostro LCDshield, però è necessario specificare con attenzione i pin utilizzati quando si dichiara l'oggetto LiquidCrystal (rs,enable, d4, d5, d6, d7):
  rs>8
  enable>9
  d4>4
  d5>5
  d6>6
  d7>7
  La corretta sintassi per creare l'oggetto "lcd" è: LiquidCrystal lcd (8, 9, 4, 5, 6, 7); modificate quindi questa riga per rendere compatibili gli esempi della libreria con il nostro shield. È necessario specificare anche il tipo di display utilizzato tramite la riga lcd.begin(numCaratteri, numRighe); numCaratteri indica il numero di caratteri del display (di

Listato 1

```
/*
  LCDshieldFE_1 Esempio1
  Pulsanti connessi ai pin digitali
 */

// include la libreria:
#include <LiquidCrystal.h>
#define P1 10
#define P2 11
#define P3 12
// inizializza il display assegnando un nome
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);

void setup() {
  // Imposta gli ingressi dei pulsanti
  pinMode(P1, INPUT_PULLUP);
  pinMode(P2, INPUT_PULLUP);
  pinMode(P3, INPUT_PULLUP);
  // Imposta il numero di righe e colonne del display
  lcd.begin(16, 2);
  // Visualizza il messaggio
  lcd.print("Hello!");
}

void loop() {
  // si posiziona sulla riga sottostante
  lcd.setCursor(0, 1);
  // visualizza il pulsante premuto
  if ( digitalRead(P1)==0 ) lcd.print("P1 ");
  if ( digitalRead(P2)==0 ) lcd.print("P2 ");
  if ( digitalRead(P3)==0 ) lcd.print("P3");
}
```

Listato 2

```
/*
  LCDshieldFE_2 Esempio2
  Pulsanti connessi alla linea analogica
 */

// include la libreria:
#include <LiquidCrystal.h>
#define pushPin 3
// inizializza il display assegnando un nome
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);

void setup() {
  // Imposta il numero di righe e colonne del display
  lcd.begin(16, 2);
  // Visualizza il messaggio
  lcd.print("Hello!");
}

void loop() {
  // si posiziona sulla riga sottostante
  lcd.setCursor(0, 1);

  // visualizza il pulsante premuto
  int pushValue = analogRead(pushPin);
  if ( pushValue < 164 )
    lcd.print("P1");
  else if ( pushValue < 502 )
    lcd.print("P2");
  else if ( pushValue < 850 )
    lcd.print("P3");
  else
    lcd.print("  ");
}
```