Tutorial Display LCD e PIC

Si fa presente per coloro che sono a "digiuno" con la programmazione dei PIC di leggersi il tutorial in questo link http://www.grix.it/viewer.php?page=429

Descrizione

Questo tutorial vuol essere rivolto a tutti coloro abbiano necessità di utilizzare un dispaly LCD di tipo ALFANUMERICO con interfaccia Hitachi 44780 di tipo parallela.

Senza entrare nei dettagli ma soffermandosi alla sola metodologia su come pilotare il display, prenderò come riferimento un display 20 caratteri a quattro linee.

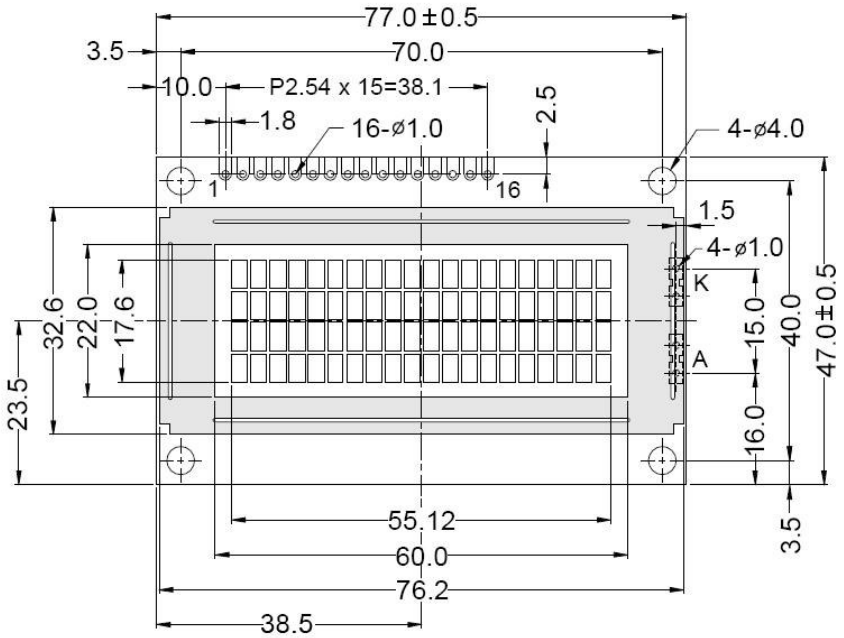


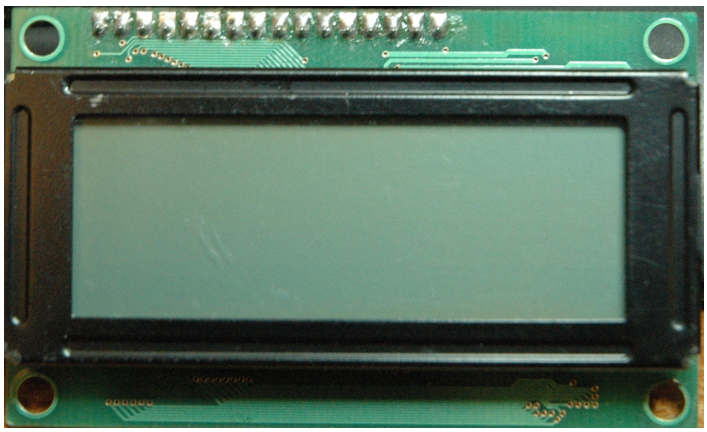
Fig.1

Come da figura 1 è riportato il disegno meccanico del display, utile per tutti coloro dovranno inserire il display in alloggiamenti con spazi prefissati.

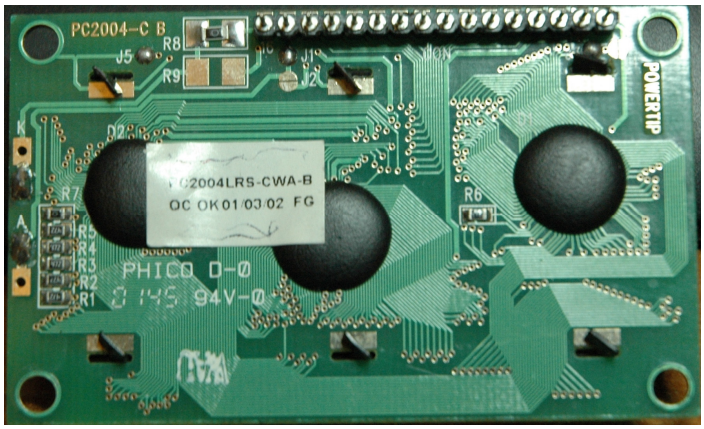
Nella parte superiore notare i contatti elettrici che vanno da 1 a 16. La disposizione dei contatti varia a seconda della marca modello grandezza, ecc.ecc.

Alcuni modelli hanno 14 contatti in quanto è assente la retroilluminazione, difatti con la retroilluminazione i contatti sono 16.

AVVISO IMPORTANTE, LA DISPOSIZIONE DEI SEGNALI RISPETTO ALLA NUMERAZIONE PUO' VARIARE DA MODELLO A MODELLO, DUNQUE PRIMA DI ESEGUIRE QUALSIASI COLLEGAMENTO CONTROLLARE IL DATASHEET DEL VOSTRO DISPLAY.



Questa è la foto anteriore di un display 20x4



Questa è la foto posteriore di un display 20x4

Ora vediamo i segnali rispettivamente alla numerazione del display, di seguito riporto la tabella dei segnali.

PIN ASSIGNMENT		
Pin no.	Symbol	Function
1	Vss	Power supply(GND)
2	Vdd	Power supply(+)
3	Vo	Contrast Adjust
4	RS	Register select signal
5	R/$\overline{W}$	Data read / write
6	E	Enable signal
7	DB0	Data bus line
8	DB1	Data bus line
9	DB2	Data bus line
10	DB3	Data bus line
11	DB4	Data bus line
12	DB5	Data bus line
13	DB6	Data bus line
14	DB7	Data bus line
15	A	Power supply for LED B/L (+)
16	K	Power supply for LED B/L (–)

Vista la disposizione dei segnali del dispaly LCD possiamo procedere al collegamento con il PIC.

Prendiamo tre tipi di schemi con altrettanti PIC diversi, per diversificare le possibilità di collegamento e di controllo.

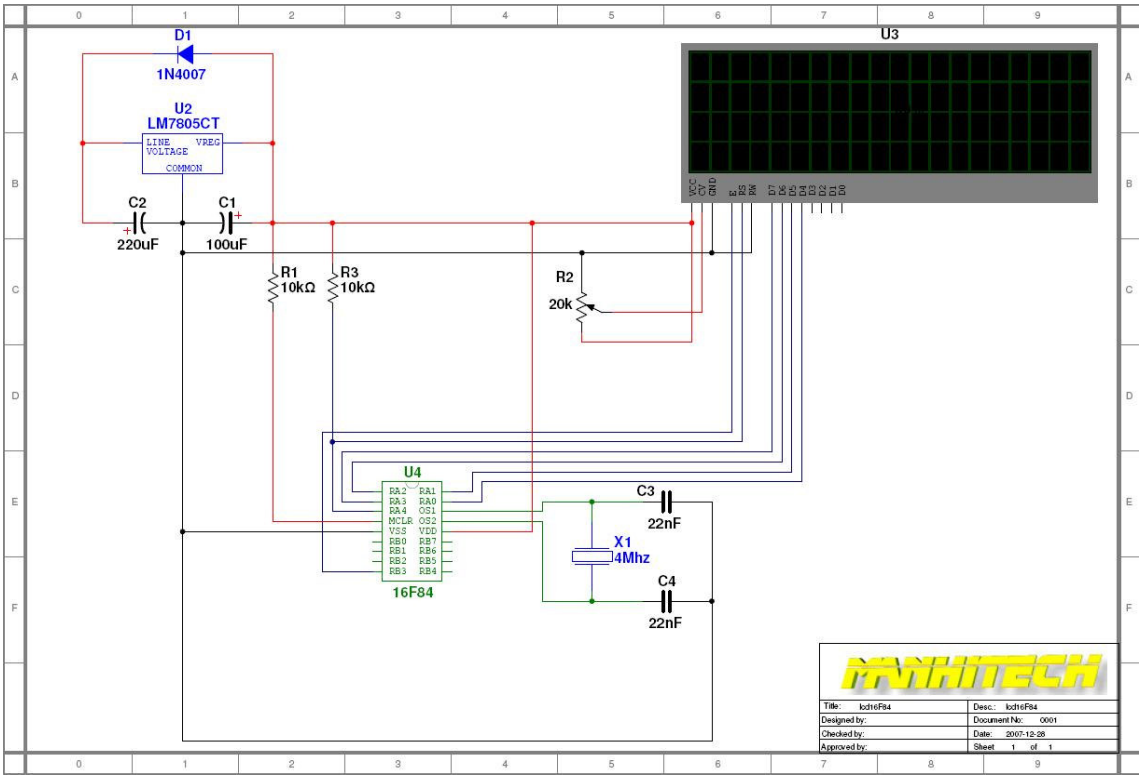


Fig.2

Questo è un esempio con il PIC16F84, da notare che non tutti i pin del display sono collegati.

Questa situazione avviene in quanto utilizziamo solo gli ultimi 4 bit da DB4..a..DB7.

Difatti possiamo decidere di pilotarlo a 4 o 8 bit.

Naturalmente pilotarlo a 8 bit significa impegnare 4 porte in più sul nostro PIC, ma avere più prestazioni dal nostro display.

Per i nostri esperimenti sono sufficienti 4 bit.

Il segnale R/W è messo direttamente a massa in quanto dobbiamo scrivere sul nostro display, ovvero inviare i dati dal PIC al display e visualizzarli e dunque per risparmiare un'altra porta del PIC colleghiamo il R/W direttamente a massa.

Il segnale CV (in realtà è il Vo, non capisco perché ha la sigla CV?!!) tramite il trimmer abbiamo la possibilità di regolarne il contrasto.

Il display viene alimentato con i 5 volt, ora non resta che analizzare il firmware.

Prima di affrontare la programmazione bisogna sapere che esistono dei comandi con i quali possiamo spostarci nell'area di visualizzazione, cancellare completamente lo schermo ecc.ecc.

I comandi sono sotto riportati,

LCDOut $fe,1 Cancella il display

LCDOut $fe,2 Porta il cursore in alto a sinistra

LCDOut $FE, $0C Spegne il cursore

LCDOut $FE, $0E Trasforma il cursore in una linea bassa

LCDOut $FE, $0F Il cursore lampeggia

LCDOut $FE, $10 Muove il cursore di una posizione a sinistra

LCDOut $FE, $14 Muove il cursore di una posizione a destra

LCDOut $fe, $80 Porta il cursore alla prima linea di scrittura

LCDOut $fe, $C0 Porta il cursore alla seconda linea di scrittura

LCDOut $fe, $94 Porta il cursore alla terza linea di scrittura

LCDOut $fe, $D4 Porta il cursore alla quarta linea di scrittura

Naturalmente se abbiamo un display con una sola linea esempio un 16x1 i comandi per spostare il cursore dalla seconda in poi non sono da utilizzare.

Con il PIC16F84 la gestione firmware è semplificata utilizzando lo schema in fig.2, difatti tutti i settaggi di controllo display sono sotto intesi (ci pensa il compilatore) e non vanno utilizzati, dunque arriviamo al nocciolo vediamo il codice sorgente.

Si ricorda che il linguaggio utilizzato è il PICBASIC PRO.

CODICE SORGENTE

LCD_16F84

Come si nota sul codice sorgente ho creato delle sub routine da richiamare ogni qual volta devo spedire un comando al display, come per esempio cancellare il display dovrei scrivere ogni volta che necessita il comando **LCDOut $fe,1.**

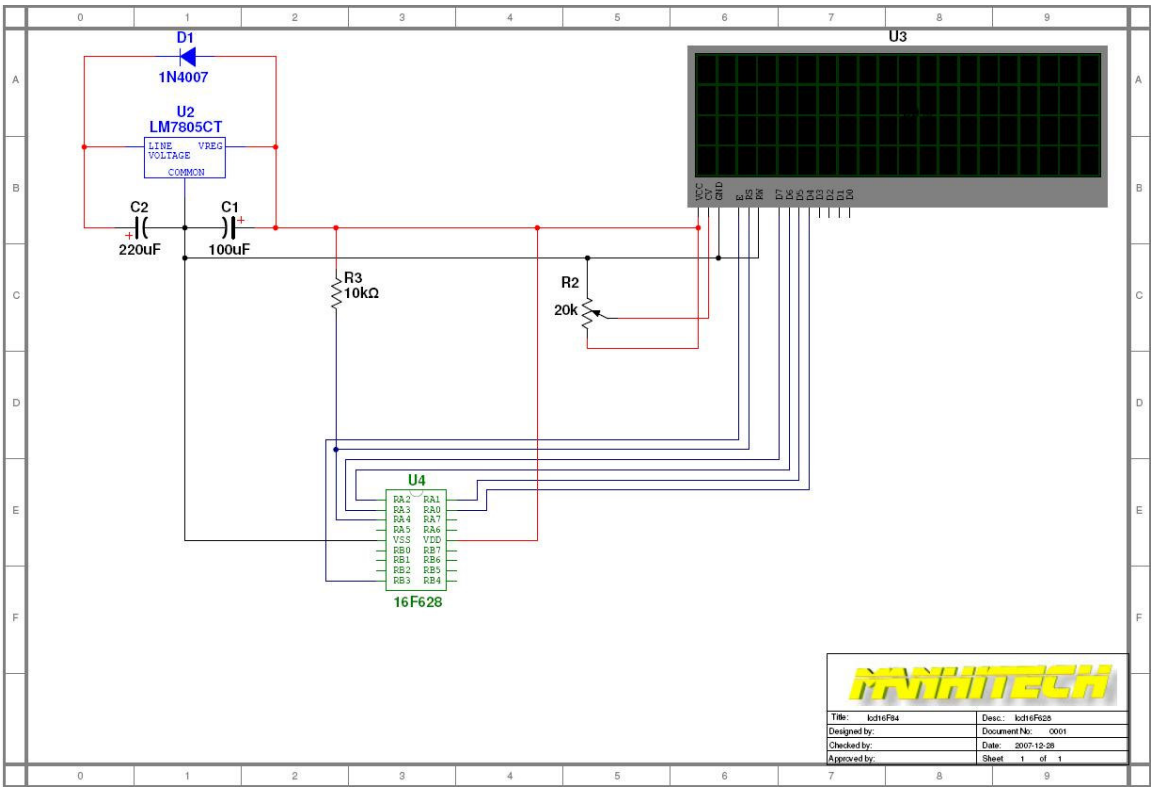
Per evitare di ricordarsi comandi assurdi gli stessi li inserisco dentro delle sub routine battezzandole con nomi più umani. Il comando per cancellare il display diventa **GoSub LCD_CLS .** Nessuno vieta di creare le sub routine con nomi italiani, esempio

```
CANCELLA_LCD:

LCDOut $fe,1

Return
```

Ora, richiamare il comando per cancellare il dispaly sarà **GoSub CANCELLA_LCD.**

Ecco il risultato nel display.

Ora vediamo lo stesso pilotato da un PIC16F628



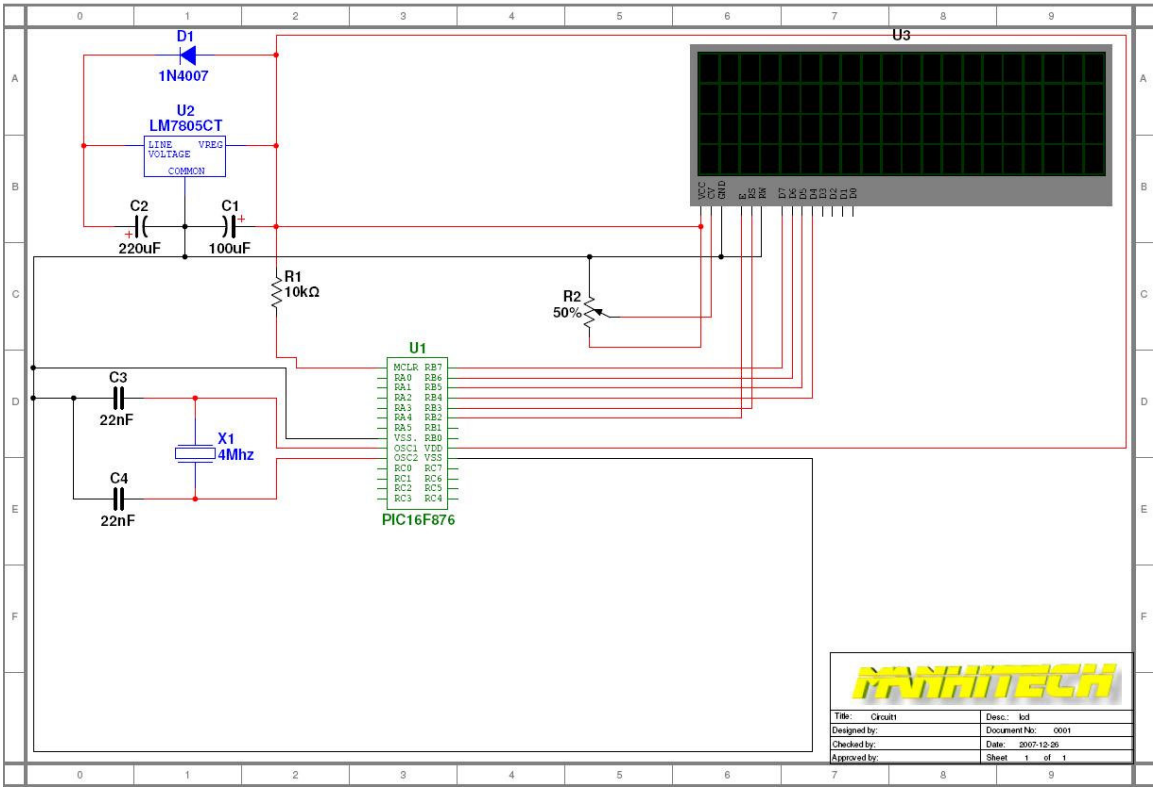
Praticamente identico al 16F84 difatti è pin compatibile, unica differenza che manca il circuito del quarzo in quanto il PIC16F628 ha al suo interno un oscillatore a 4Mhz, dunque possiamo evitare di inserire il quarzo.

Cosi il firmware è praticamente identico cambiano alcuni comandi per la gestione del PIC in questione.

CODICE SORGENTE

LCD_16F628

Ora analiziamo la situazione con un PIC16F876.



Essendo un PIC completamente diverso dal 16F84 / 628 cambiano le connessioni sullo stesso pic.

Come lo schema cambia in parte anche il firmware, difatti sono presenti i comandi per la gestione del display.

CODICE SORGENTE

LCD_16F876

Naturalmente se per esigenze di schema si ha la necessità di pilotare il bus con la porta C del PIC cambiero come segue

DEFINE LCD_DREG PORTC 'bus dati sulla porta C

Così per i restanti **DEFINE** che dovranno rispecchiare i collegamenti tra il PIC e il dispaly.

Da notare che nello schema ho collegato il segnale R/W a massa ma l'ho definito nel firmware,

DEFINE LCD_RWREG PORTC 'definisco su quale porta è il R/W

DEFINE LCD_RWBIT 2 'definisco il pin per il R/W

Questo solo per far conoscere il comando, difatti se utilizzassimo il comando LCDIN il segnale R/W andrebbe collegato al PIC e definito.

Il resto del firmware è uguale ai precedenti.

CONCLUSIONE

Spero di aver dato chiarimenti su questo argomento, se ci sono dubbi potete contattarmi.

Un saluto by Doberman (IS0GUP)