M.M.Electronics - http://www.mmetft.it

Michele Marino - michele.marino@mmetft.it

Controllo di un display LCD 16x2

V 0.2

Novembre 2007

INFORMATIVA

Come prescritto dall'art. 1, comma 1, della legge 21 maggio 2004 n.128, l'autore avvisa di aver assolto, per la seguente opera dell'ingegno, a tutti gli obblighi della legge 22 Aprile del 1941 n. 633, sulla tutela del diritto d'autore. Tutti i diritti di questa opera sono riservati. Ogni riproduzione ed ogni altra forma di diffusione al pubblico dell'opera, o parte di essa, senza un'autorizzazione scritta dell'autore, rappresenta una violazione della legge che tutela il diritto d'autore, in particolare non ne è consentito un utilizzo per trarne profitto. La mancata osservanza della legge 22 Aprile del 1941 n. 633 è perseguibile con la reclusione o sanzione pecuniaria, come descritto al Titolo III, Capo III, Sezione II. A norma dell'art. 70 è comunque consentito, per scopi di critica o discussione, il riassunto e la citazione, accompagnati dalla menzione del titolo dell'opera e dal nome dell'autore.

AVVERTENZE

Chiunque decida di far uso delle nozioni riportate nella seguente opera o decida di realizzare i circuiti proposti, è tenuto pertanto a prestare la massima attenzione in osservanza alle normative in vigore sulla sicurezza.
L'autore declina ogni responsabilità per eventuali danni causati a persone, animali o cose derivante dall'utilizzo diretto o indiretto del materiale, dei dispositivi o del software presentati nella seguente opera.
Si fa inoltre presente che quanto riportato viene fornito cosi com'è, a solo scopo didattico e formativo, senza garanzia alcuna della sua correttezza.
L'autore ringrazia anticipatamente per la segnalazione di ogni errore.

Indice

Elenco delle figure

Elenco delle tabelle

1 Introduzione

In questo articolo verrà descritto l'utilizzo di un display LCD a sedici caratteri per due righe. Il display utilizzato si basa su un microcontrollore standard HD44780 della Hitachi per cui la trattazione qui esposta è valida per un range piuttosto ampio di display LCD in commercio.

2 Il controller HD44780

Nella figura 1 viene mostrata la struttura interna del controller per il display CDL4462 preso in considerazione. Il funzionamento dell'HD44780 si basa sull'utilizzo dei registri IR e DR. Il registro IR (Instruction Register) contiene i codici istruzione e l'indirizzamento per la RAM dati del display (DDRAM - Display Data RAM) e la RAM per la generazione dei caratteri (CGRAM - Character Generation RAM). Il registro DR contiene i dati che possono essere scritti o letti dalla DDRAM o dalla CGRAM. Il segnale RS permette la selezione tra i due registri.

E' presente inoltre un flag che segnala se il controller è occupato in una operazione interna oppure può eseguire una nuova istruzione. Tale flag viene segnalato sulla linea DB7 quando RS=0 e $R/\overline{W} = 1$. L'istruzione successiva può essere scritta e quindi eseguita, solo quando tale bit (busy) viene riportato a zero. L'address counter (AC) consente di indirizzare la DDRAM e la CGRAM. Quando l'istruzione presente nel registro istruzione (IR) contiene il riferimento ad un indirizzo, questo viene passato all'AC.

Dopo una lettura o una scrittura nella DDRAM o CGRAM, l'AC viene rispettivamente decrementato o incrementato. Il valore dell'AC viene quindi riportato sulle linee DB0...DB6 quando RS=0 e $R/\overline{W} = 1$.

La RAM dati del display contiene i dati relativi alla visualizzazione sotto forma di codici a 8 bit. Questa è una memoria di 80x8 bit ed è quindi in grado di memorizzare 80 caratteri. L'indirizzo delle celle della DDRAM è contenuto nell'AC (Address Counter). Quando la visualizzazione avviene su due linee, l'indirizzamento avviene secondo lo schema di figura 2.

Display position	1	2	3	4	5	.....	39	40
DDRAM address	00	01	02	03	04	.....	26	27
(hexadecimal)	40	41	42	43	44	.....	66	67

Figura 2: Indirizzamento a due righe

Come si può notare l'indirizzo finale della prima riga e quello iniziale della seconda riga non sono consecutivi. La figura 3 mostra invece l'indirizzamento relativo alla visualizzazione di 16 caratteri su due righe con relative operazioni di shift a destra e/o sinistra.

Display position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DDRAM address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F
	HD44780U display								Extension driver display							
For shift left	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10
	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50
For shift right	27	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E
	67	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E

Figura 3: Indirizzamento 16x2

La lista dei caratteri visualizzabili sul display è contenuta in una memoria chiamata CGROM. Ogni carattere è rappre-

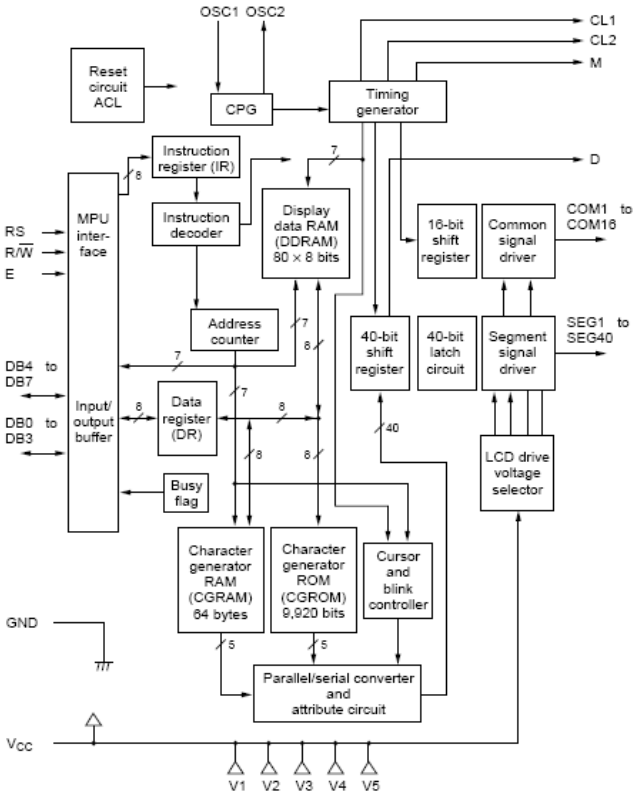

Figura 1: Struttura controller display

sentato da un codice che viene selezionato attraverso una memoria DDRAM (Display Data RAM). A tale codice corrisponde un indirizzo della CGRAM che punta al valore contenuto nella relativa locazione della CGRAM. Tale valore specifica i pixel che vengono accesi sul display per visualizzare il carattere selezionato.

Il controller HD44780 possiede 16 linee per il potenziale comune (back plain) e 40 linee per il controllo dei segmenti a cristalli liquidi. Le linee COM1..COM16 vengono collegate sulle righe della matrice di punti a cristalli liquidi, mentre le linee SEG1..SEG40 rappresentano le colonne. La figura 4 mostra un esempio di connessione dell'HD44780 al display LCD.

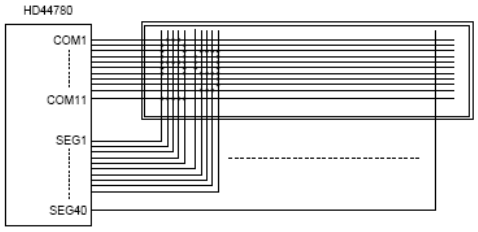

Figura 4: Esempio display con caratteri a 5x10 punti

3 Il display

La figura 5 mostra la numerazione dei pin sulla scheda contenente il display e il chip di controllo. La piedinatura viene riportata nella tabella 1. La tabella 2 mostra invece l'impostazione delle linee RS e $R/\overline{W}$ e le relative operazioni sul display.

Il display può essere pilotato con BUS a 8 bit oppure a 4 bit con conseguente riduzione delle linee di I/O necessarie.

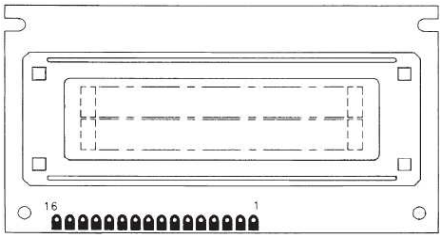

Figura 5: Vista piedinatura display

Con l'utilizzo di sole quattro linee, il codice ASCII relativo ad ogni carattere viene multiplexato verso il display, inviando prima i quattro bit più significativi e, successivamente quelli meno significativi. La figura 6 mostra la connessione di un trimmer per la regolazione del contrasto del display. Il trimmer viene collegato col cursore centrale sul piedino 5 del display (V_O) e con gli estremi sull'alimentazione positiva e verso massa.

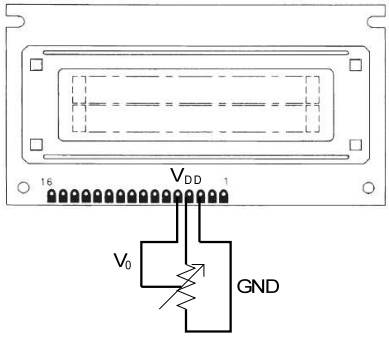

Figura 6: Regolazione contrasto display

All'accensione il controller riceve un reset interno dopodiché vengono eseguite le seguenti istruzioni:

1. pulisci display
2. funzioni settaggio display:
   - DL=1; intefaccia a 8 bit

pin	nome segnale	funzione
1	BL+	Alimentazione positiva LED retroilluminazione
2	BL-	Alimentazione negativa LED retroilluminazione
3	GND	Alimentazione a 0V
4	V_{DD}	Alimentazione a +5V
5	V_O	Regolazione contrasto LCD
6	RS	alto:invio codici istruzione; basso:invio/ricezione dati
7	$R/\overline{W}$	alto:lettura dati; basso:scrittura dati
8	E	Abilitazione logica display
9	DB0	Linea 0 BUS dati
10	DB1	Linea 1 BUS dati
11	DB2	Linea 2 BUS dati
12	DB3	Linea 3 BUS dati
13	DB4	Linea 4 BUS dati
14	DB5	Linea 5 BUS dati
15	DB6	Linea 6 BUS dati
16	DB7	Linea 7 BUS dati

Tabella 1: Significato pin display

- N=0; display a 1 riga
- F=0; dimensioni carattere 5x8 punti

3. controllo on/off display:
   - D=0; spegni display
   - C=0; disattiva cursore
   - B=0; disattiva blink
4. settaggio scrittura display:
   - I/D=1; incrementa cursore di una posizione
   - S=0; nessuno shift

Qualora il reset non vada a buon fine è necessario inizializzare il display via software. La tabella 3 mostra la lista delle istruzioni per la gestione del display.

Il significato dei vari bit presenti nella tabella 3 viene riportato nella tabella 4.

L'inizializzazione del display consiste in una serie di operazioni interne che permettono di utilizzare il display con interfaccia a 8 bit o a 4 bit. La figura 7 mostra il flusso di inizializzazione del display per l'utilizzo con interfaccia a 8 bit. L'inizializzazione consiste nei seguenti passi:

- alimentazione display
- attesa di 15 ms
- impostazione interfaccia a 8 bit - (♯1)
- attesa di 4.1ms
- impostazione interfaccia a 8 bit - (♯2)
- attesa di 100μs
- impostazione interfaccia a 8 bit - (♯3)

Bit	Significato
I/D=1	Incrementa posizione cursore
I/D=0	Decrementa posizione cursore
S=1	Abilita shift display
S/C=1	Shift display
S/C=0	Muovi cursore
R/L=1	Shift a destra
R/L=0	Shift a sinistra
DL=1	Interfaccia a 8 bit
DL=0	Interfaccia a 4 bit
N=1	Visualizzazione a due righe
N=0	Visualizzazione a una riga
F=1	Caratteri 5x10 punti
F=0	Caratteri 5x8 punti
BF=1	Operazione interna
BF=0	Display operativo
DDRAM	RAM dati display
CGRAM	RAM generazione caratteri
AGC	Indirizzo CGRAM
ADD	Indirizzo DDRAM
AC	Contatore indirizzo

Tabella 4: Significato bit istruzioni

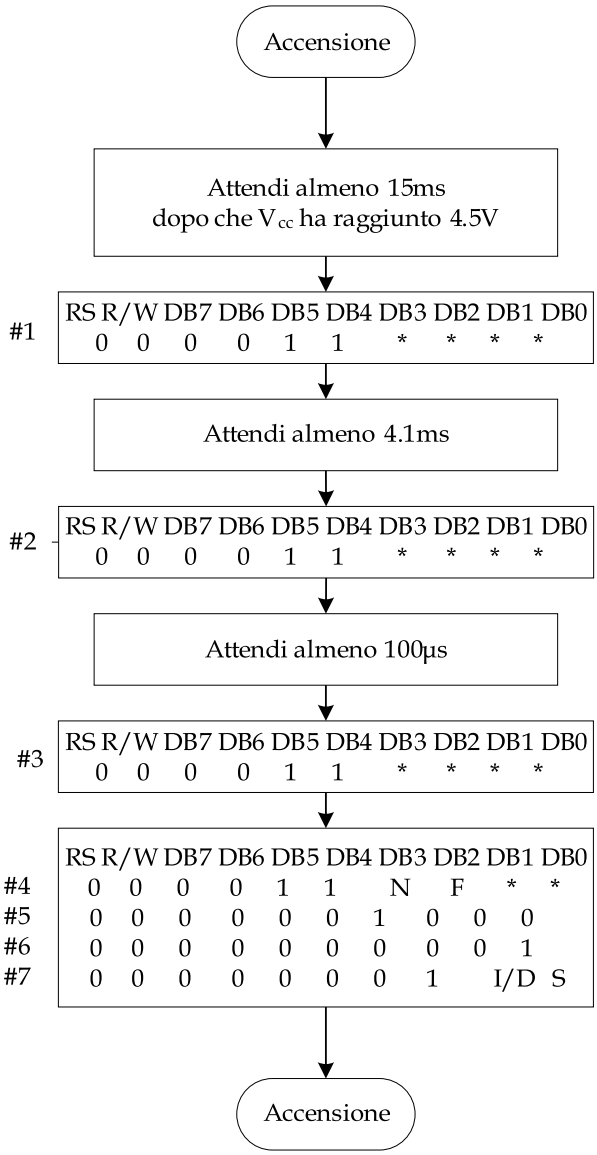

Figura 7: Inizializzazione interfaccia a 8 bit

RS	$R/\overline{W}$	operazione
0	0	scrittura operazioni interne
0	1	verifica se il display sta eseguendo operazioni (DB7)
1	0	scrittura operazioni interne - memoria video
1	1	lettura operazioni interne - memoria video

Tabella 2: Selezione operazioni display

Istruzione	RS	$R/\overline{W}$	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Pulisci display	0	0	0	0	0	0	0	0	0	1
Posizione iniziale	0	0	0	0	0	0	0	0	1	-
Entry mode set	0	0	0	0	0	0	0	1	I/D	S
Display On/Off	0	0	0	0	0	0	1	D	C	B
Shift cursore/display	0	0	0	0	0	1	S/C	R/L	-	-
Imposta funzione	0	0	0	0	1	DL	N	F	-	-
Indirizzamento CGRAM	0	0	0	1	ACG	ACG	ACG	ACG	ACG	ACG
indirizzamento DDRAM	0	0	1	ADD	ADD	ADD	ADD	ADD	ADD	ADD
Stato BUS	0	1	BF	AC	AC	AC	AC	AC	AC	AC
Scrivi nella DDRAM/CGRAM	1	0	DT	DT	DT	DT	DT	DT	DT	DT
Leggi dalla DDRAM/CGRAM	1	1	DT	DT	DT	DT	DT	DT	DT	DT

Tabella 3: Istruzioni display

- impostazione interfaccia a 8 bit, selezione del numero di righe visualizzate (N) e dimensione caratteri (F) - (♯4)
- spegni display - (♯5)
- pulisci display - (♯6)
- entry mode set, impostazione incremento/decremento cursore (I/D), shift display (S) - (♯7)

L'inizializzazione è identica alla precedente tranne che per l'istruzione numero 4, dove il bit DB4 viene posto a zero, selezionando quindi l'interfaccia a 4 bit. Con l'interfaccia a 4 bit si utilizzano solo i bit più significativi del BUS e le istruzioni (a 8 bit) vengono divise e inviate in nibble da 4 bit. L'esecuzione delle istruzioni avviene inviando prima i quattro bit più significativi e poi quelli meno significativi. La figura 8 mostra il flusso di inizializzazione per l'interfaccia a 4 bit.

4 Il codice assembler

Nella sezione che segue viene riportata la trattazione relativa al codice assembler per l'inizializzazione e la gestione del display LCD a 16 caratteri per due righe. Il codice verrà sviluppato per il controllo tramite BUS a 4 bit. Questo rende la parte software leggermente più complessa, ma consente di dimezzare il numero di linee di I/O utilizzabili per implementare altre funzioni. Il microcontrollore utilizzato è il PIC16F877 mostrato nella figura 9.

Il BUS a 4 bit viene gestito attraverso le linee RB7, RB6, RB5, RB4, il segnale di abilitazione del display attraverso la linea RA4 e il controllo istruzioni/dato

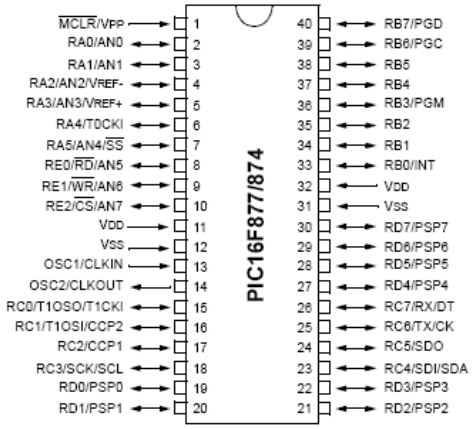

Figura 9: Piedinatura PIC16F877

(RS) attraverso la linea RA2[1] (figura 10). La linea 7 ($R/\overline{W}$) viene portata costantemente al livello logico 0 per cui il display rimane sempre nella configurazione di scrittura dati/istruzioni. La figura 11 mostra l'inizializzazione dei registri utilizzati nel controllo del display.

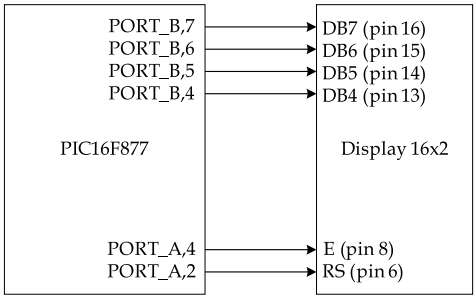

Figura 10: Connessione PIC-LCD

Nel seguito si farà uso della procedura di ritardo riportata nella figura 12. Questa non è altro che la cascata di tre registri decrementati in sequenza. Il clock

[1] La linea RA2 è di tipo open collector per cui è necessario inserire un pull-up verso l'alimentazione positiva a 5V.

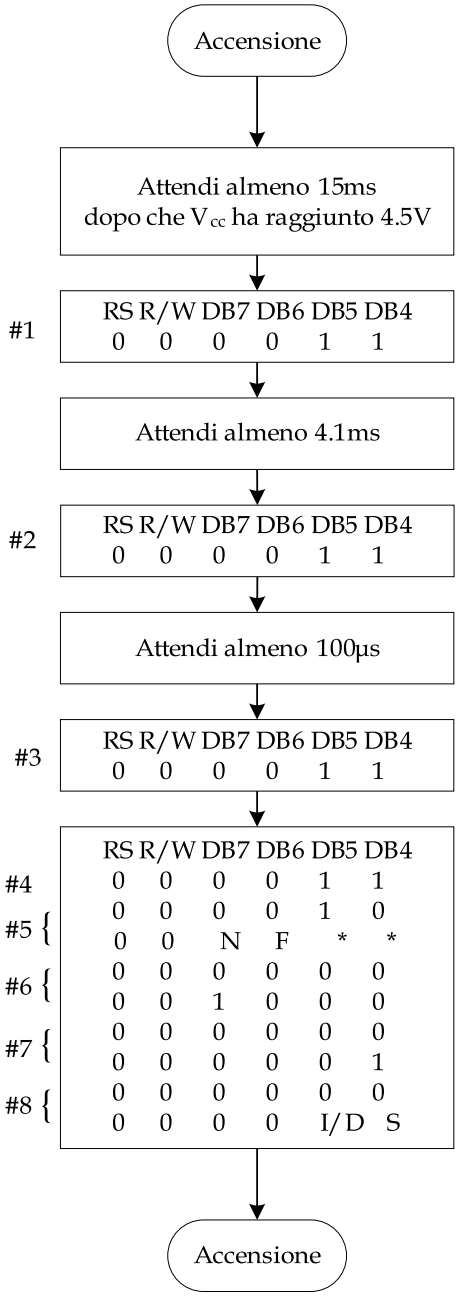

Figura 8: Inizializzazione interfaccia a 4 bit

```
Del2    Movwf D2     ;Routine doppio ritardo
DelR2   Movlw 0xFF   ;[0.83s]MAX
        Call Del1
        Decfsz D2,1
        Goto DelR2
        Return
Del1    Movwf D1     ;Routine singolo ritardo
DelR1   Movlw 0xFF   ;[3.25ms]MAX
        Call Del0
        Decfsz D1,1
        Goto DelR1
        Return
Del0    Movwf D0     ;Routine ritardo
DelR0   Decfsz D0,1  ;[12.75us]MAX
        Goto DelR0
        Return
```

Figura 12: Subroutine di ritardo

```
STATUS  EQU 0X03
PORT_A  EQU 0X05
PORT_B  EQU 0X06
PORT_C  EQU 0X07
PORT_D  EQU 0X08
PORT_E  EQU 0X09
TRIS_A  EQU 0X85
TRIS_B  EQU 0X86
TRIS_C  EQU 0X87
TRIS_D  EQU 0X88
TRIS_E  EQU 0X89
INTCON  EQU 0X0B
D0      EQU 0X20
D1      EQU 0X21
D2      EQU 0X22
Write   EQU 0X23
ADCON1  EQU 0x9F
```

Figura 11: Dichiarazione registri

di sistema è fissato a 20MHz e ogni ciclo istruzione è lungo 200ns.

Supponendo di caricare la variabile D0 con il suo valore massimo $(255)_d$, passando tale valore ala routine Del0, il registro D0 viene decrementato 255 volte prima di uscire e ritornare al path di esecuzione originario. Ad ogni decremento viene eseguita l'istruzione di salto incondzionato Goto che richiede due cicli istruzione. Questo vuol dire che il path di esecuzione viene ritardato per un tempo pari a:

$$\begin{aligned} T_{0d} &= D0 \cdot (T_{DECFSZ} + T_{GOTO}) = \\ &= 255 \cdot (200 \cdot 10^{-9} + 400 \cdot 10^{-9}) = \\ &= 153\mu s \end{aligned} \quad (1)$$

Se il tempo di ritardo ottenuto è insufficiente, viene aumentato attraverso il controllo del registro D1 e la routine Del1. Supponiamo di caricare il valore massimo in D1 $(255)_d$ e chiamiamo la routine Del1. Tale routine non fa altro che eseguire la routine Del0, con $D0 = (255)_d$,

D1 volte. In questo caso è necessario tenere conto anche di due cicli aggiuntivi per la chiamata a Del0 e due cicli relativi all'istruzione Return. Con i valori supposti si ottiene quindi un ritardo pari a:

$$\begin{aligned} T_{1d} &= D1 \cdot (T_{DECFSZ} + T_{CALL} + \\ &+ T_{0d} + T_{RETURN} + T_{GOTO}) = \\ &= 255 \cdot (200 \cdot 10^{-9} + 400 \cdot 10^{-9} + \\ &+ T_{0d} + 400 \cdot 10^{-9} + 400 \cdot 10^{-9}) = \\ &= 39.4ms \end{aligned} \quad (2)$$

Analogamente, supponendo di caricare D2 con il suo valore massimo e chiamando la routine Del2, si ottiene un ritardo massimo pari a:

$$\begin{aligned} T_{2d} &= D2 \cdot (T_{DECFSZ} + T_{CALL} + \\ &T_{1d} + T_{RETURN} + T_{GOTO}) = \\ &= 255 \cdot (200 \cdot 10^{-9} + 400 \cdot 10^{-9} + \\ &+ T_{1d} + 400 \cdot 10^{-9} + 400 \cdot 10^{-9}) \\ &= 10.04s \end{aligned} \quad (3)$$

Facendo riferimento alla figura 8, sono richiesti tre ritardi di cui il primo di 15ms, il secondo di 4.1ms e il terzo di $100\mu s$. I primi due ritardi si ottengono mediante la routine Del1 caricando D1, rispettivamente, con i valori $(100)_d$ e $(27)_d$. I ritardi effettivi sono di 15.4ms e 4.2ms. Per quanto riguarda il ritardo di $100\mu s$, questo si ottiene attraverso la routine Del0 ponendo $D0 = (170)_d$. Il ritardo effettivo è di $102\mu s$.

Prima di eseguire la procedura di inizializzazione del display dobbiamo inzializzare i registri attraverso le istruzioni riportate nella figura 13. Come si può vedere tutte le porte vengono configurate come uscite e si nota in particolare la scrittura sul registro ADCON1 che consente di settare le linee RA0, RA1 e RA3 come ingressi analogici, mentre RA2, RA4, RA5 vengono configurate come uscite digitali[2].

La scrittura delle istruzioni sul display display si basa sulla procedura WrDisp riportata nella figura 14. L'istruzione o il dato da inviare al display viene caricato nel registro di lavoro WREG e trasferito successivamente nel registro Write. A questo punto viene effettuata una chiamata alla sub-routine SetReg. Qui viene eseguito un AND logico tra il valore binario "00001111" e la porta B. L'operazione consiste nell'azzeramento dei bit più significativi, lasciando inalterati i 4 bit meno significativi non utilizzati nel controllo del display. A questo punto viene eseguito un AND logico tra il valore binario "11110000" e il valore contenuto in Write. Questo serve a selezionare i 4 bit più significativi del registro Write, azzerando quelli meno significativi. Il risultato viene salvato nel registro di lavore WREG. A questo punto si ritorna dalla sub-routine SetReg e viene eseguita una chiamata alla sub-routine SetPB. Qui viene eseguito uno XOR tra il valore del registro WREG appena salvato e la porta B. Lo scopo è quello di scrivere i quattro bit più significativi di Write sulla porta B senza modificarne i 4 bit meno significativi. Fatto ciò viene inviato un impulso sulla linea RA4 collegata al segnale di Enable del display. Al ritorno dalla sub-routine, vengono invertiti i 4 bit meno significativi di Write con quello più significativi. Lo scopo è quello di eseguire nuovamente le sub-routine SetReg, SetPB e Enable al fine di inviare

[2] Per maggiori chiarimenti sul settaggio della porta A si faccia riferimento all'articolo sul controllo analogico di una tastiera a matrice presente sul sito www.mmetft.it

```
Init    ORG 0x008
        Bcf INTCON,7        ;Disabilita interrupt
        Bsf STATUS,5        ;Passa al banco registri 1
        Movlw B'00000100'   ;Linee RA0,RA1,RA3 ingressi analogici
        Movwf ADCON1
        Clrf TRIS_A         ;Porte configurate come uscite
        Clrf TRIS_B
        Clrf TRIS_C
        Clrf TRIS_D
        Clrf TRIS_E
        Bcf STATUS,5        ;Passa al banco registri 0
```

Figura 13: Inizializzazione registri

```
WrDisp  Movwf Write
        Call SetReg
        Call SetPB
        Swapf Write,1
        Call SetReg
SetPB   Xorwf PORT_B,1
Enable  Bsf PORT_A,4
        Movlw 0x20
        Call Del0
        Bcf PORT_A,4
        Return
SetReg  Movlw 0x0F
        Andwf PORT_B,1
        Movlw 0xF0
        Andwf Write,0
        Return
```

Figura 14: Routine scrittura display

i 4 bit meno significativi del registro Write senza alterare il valore dei 4 bit meno significativi della porta B. La larghezza dell'impulso di enable è di circa $20\mu s$.

A questo punto è possibile eseguire la procedura di inizializzazione a 4 bit (figura 8) attraverso il codice assembler riportato in figura 15.

La prima istruzione setta un livello logico basso sulla linea RA2 collegata alla linea RS del display. Questo significa che stiamo inviando istruzioni sul display. A questo punto viene eseguita la routine Del1 al fine avere un ritardo[3] di circa 15ms. L'istruzione 5 invia l'istruzione relativa all'interfacciamento a 8 bit. A questo punto viene nuovamente ritardata l'esecuzione per un tempo pari a circa 4ms. L'istruzione 9 consente di inviare di nuovo l'istruzione per l'impostazione dell'interfaccia a 8 bit. A questo punto viene inserito un ritardo pari a circa $100\mu s$ e viene inviata per l'ultima volta l'istruzione per l'impostazione del BUS a 8 bit. Subito dopo (istruzione 13) viene inviata l'istruzione per l'impostazione dell'interfaccia a 4 bit. Da questo punto in poi le

[3] I valori riportati nel codice assembler sono espressi in esadecimale.

```
InDisp  Bcf PORT_A,2          ;#1
        Movlw 0x64            ;#2
        Call Del1             ;#3
        Movlw B'00110000'     ;#4
        Movwf PORT_B          ;#5
        Call Enable           ;#6
        Movlw 0x1B            ;#7
        Call Del1             ;#8
        Call Enable           ;#9
        Movlw 0xAA            ;#10
        Call Del0             ;#11
        Call Enable           ;#12
        Movlw B'00100000'     ;#13
        Movwf PORT_B          ;#14
        Call Enable           ;#15
        Movlw B'00101000'     ;#16
        Call WrDisp           ;#17
        Movlw B'00001000'     ;#18
        Call WrDisp           ;#19
        Movlw B'00000001'     ;#20
        Call WrDisp           ;#21
        Movlw B'00000011'     ;#22
        Call WrDisp           ;#23
        Movlw B'00001110'     ;#24
        Call WrDisp           ;#25
```

Figura 15: Codice inizializzazione display

istruzioni vengono ricevute sul BUS a 4 bit inviando prima la parte più significativa seguita da quella meno significativa (routine WrDisp).

Alla riga 16 troviamo l'istruzione relativa all'impostazione dell'interfaccia a 4 bit su due righe del display. L'istruzione 18 consente di spegnere il display, la 20 di cancellarne il contenuto e la 22 di entrare nella modalità Entry mode set. L'ultima istruzione (24) consente di riaccendere il display visualizzando anche il cursore. Per il significato dei codici e quindi delle istruzioni del display si faccia riferimento alla tabella 3.

Ora abbiamo il display pronto per poter ricevere stringhe da visualizzare. La figura 16 mostra un semplice esempio di visualizzazione di caratteri. L'esempio riportato imposta in prima battuta la posizione nella quale andremo a scrivere i caratteri ovvero, la quinta posizione nella prima riga. Si ricordi che l'istruzione relativa all'impostazione della CGRAM è $(10000000)_B$ per cui, essendo la quinta locazione della prima riga $(00000100)_B$, il comando relativo alla quinta locazione nella prima riga è $(10000100)_B$. Successivamente viene scritta la stringa "Questa e' " inviando i caratteri sequenzialmente. A questo punto viene impostata la scrittura nella quinta posizione della seconda riga e viene inviata la stringa "una prova". Si ricordi anche in questo caso che la quinta locazione della seconda riga è $(01000100)_B$ ed essendo l'istruzione di indirizzamento $(10000000)_B$ il comando per il settaggio della posizione nella seconda riga è $(11000100)_B$. Il passaggio dalle istruzioni per l'impostazione della posizione di scrittura e la scrittura dei caratteri avviene mediante il controllo della linea RS collegata alla RA2 della por-

```
Main    ORG 0x030
        Bcf PORT_A,2        ;Scrivi istruzione
        Movlw 0x84
        Call WrDisp         ;Posizione 5 riga 1
        Bsf PORT_A,2        ;Scrivi dato
        Movlw B'01010001'
        Call WrDisp         ;Scrivi Q
        Movlw B'01110101'
        Call WrDisp         ;Scrivi u
        Movlw B'01100101'
        Call WrDisp         ;Scrivi e
        Movlw B'01110011'
        Call WrDisp         ;Scrivi s
        Movlw B'01110100'
        Call WrDisp         ;Scrivi t
        Movlw B'01100001'
        Call WrDisp         ;Scrivi a
        Movlw B'00100000'
        Call WrDisp         ;Spazio
        Movlw B'01100101'
        Call WrDisp         ;Scrivi e
        Movlw B'00100111'
        Call WrDisp         ;Scrivi '
        Bcf PORT_A,2        ;Scrivi istruzione
        Movlw 0xC4
        Call WrDisp         ;Posizione 5 riga 2
        Bsf PORT_A,2        ;Scrivi dato
        Movlw B'01110101'
        Call WrDisp         ;Scrivi u
        Movlw B'01101110'
        Call WrDisp         ;Scrivi n
        Movlw B'01100001'
        Call WrDisp         ;Scrivi a
        Movlw B'00100000'
        Call WrDisp         ;Spazio
        Movlw B'01110000'
        Call WrDisp         ;Scrivi p
        Movlw B'01110010'
        Call WrDisp         ;Scrivi r
        Movlw B'01101111'
        Call WrDisp         ;Scrivi o
        Movlw B'01110110'
        Call WrDisp         ;Scrivi v
        Movlw B'01100001'
        Call WrDisp         ;Scrivi a
```

Figura 16: Codice di esempio

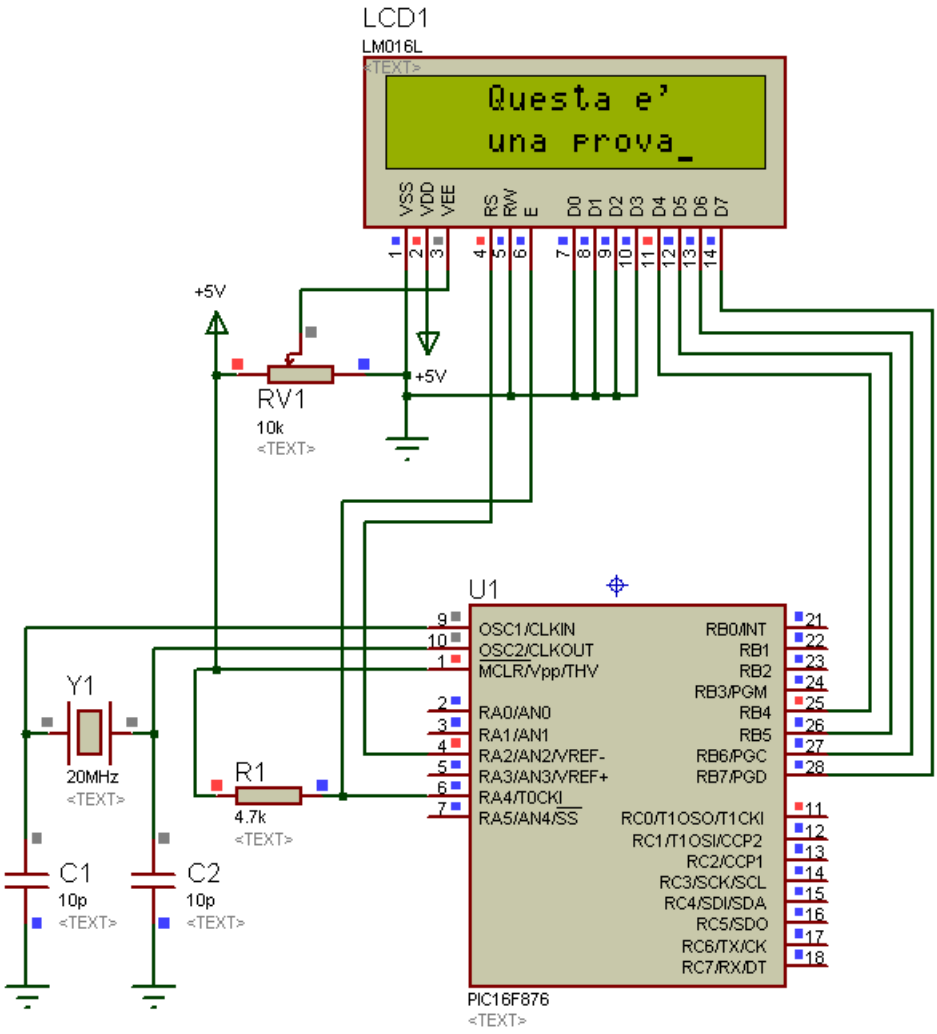

Figura 17: Simulazione Proteus del codice assembler

ta A del PIC. La figura 17 infine, mostra la simulazione del PIC programmato con l'assemlber riportato nell'articolo e del display a cristalli liquidi 16x2.

5 Conclusioni

Lo scopo di questo articolo è quello di fornire una linea guida per il controllo del display. Tutte le istruzioni relative allo shift, alla visualizzazione del cursore o al blinking dello stesso e di altre istruzioni possono essere testate facilmente dal lettore. Infine, la figura 18 mostra la corrispondenza dei codici associati ai caratteri visualizzabili sul display.

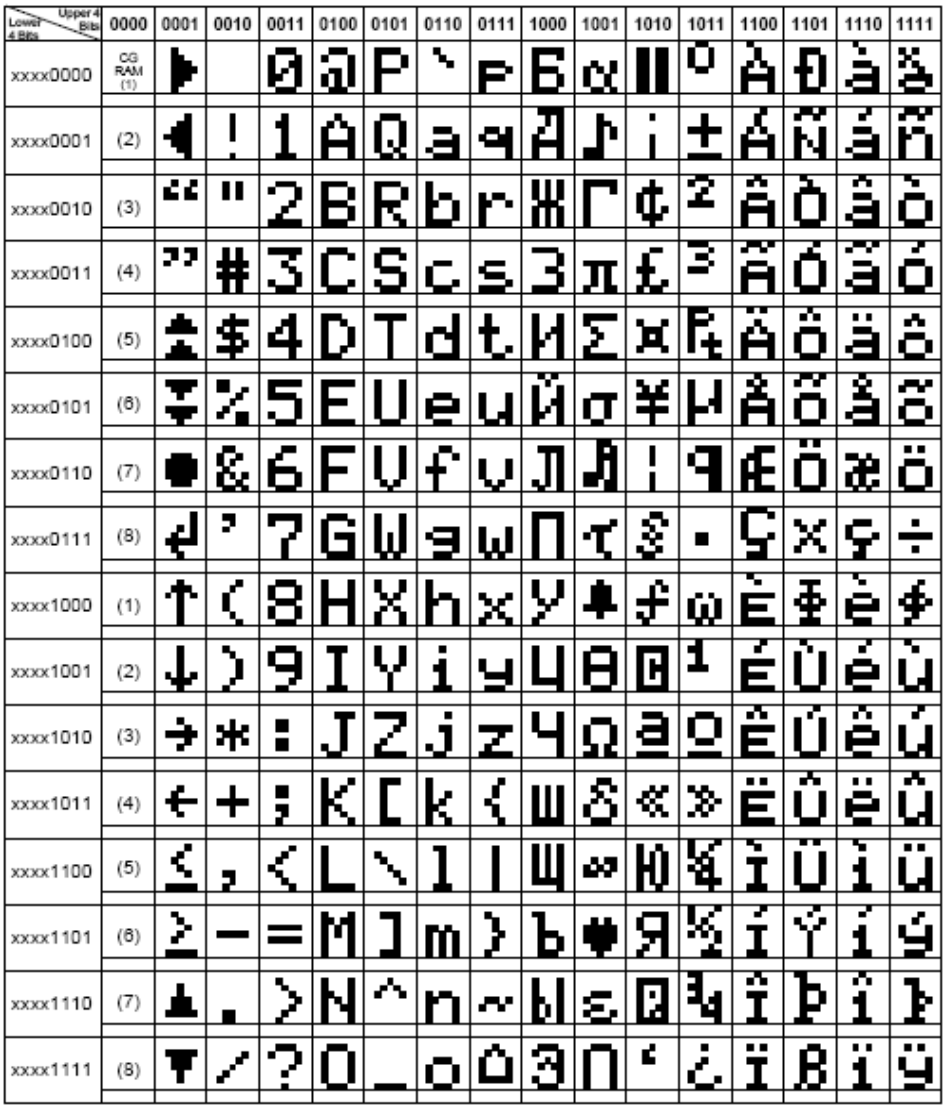

Figura 18: Codici caratteri

Riferimenti bibliografici

[1] Hitachi, *'Dot Matrix Liquid Crystal Display Controller/Driver'*, HD44780U Datasheet.

[2] Microchip, *'PIC16F87X Datasheet'*, Microchip Arizona.