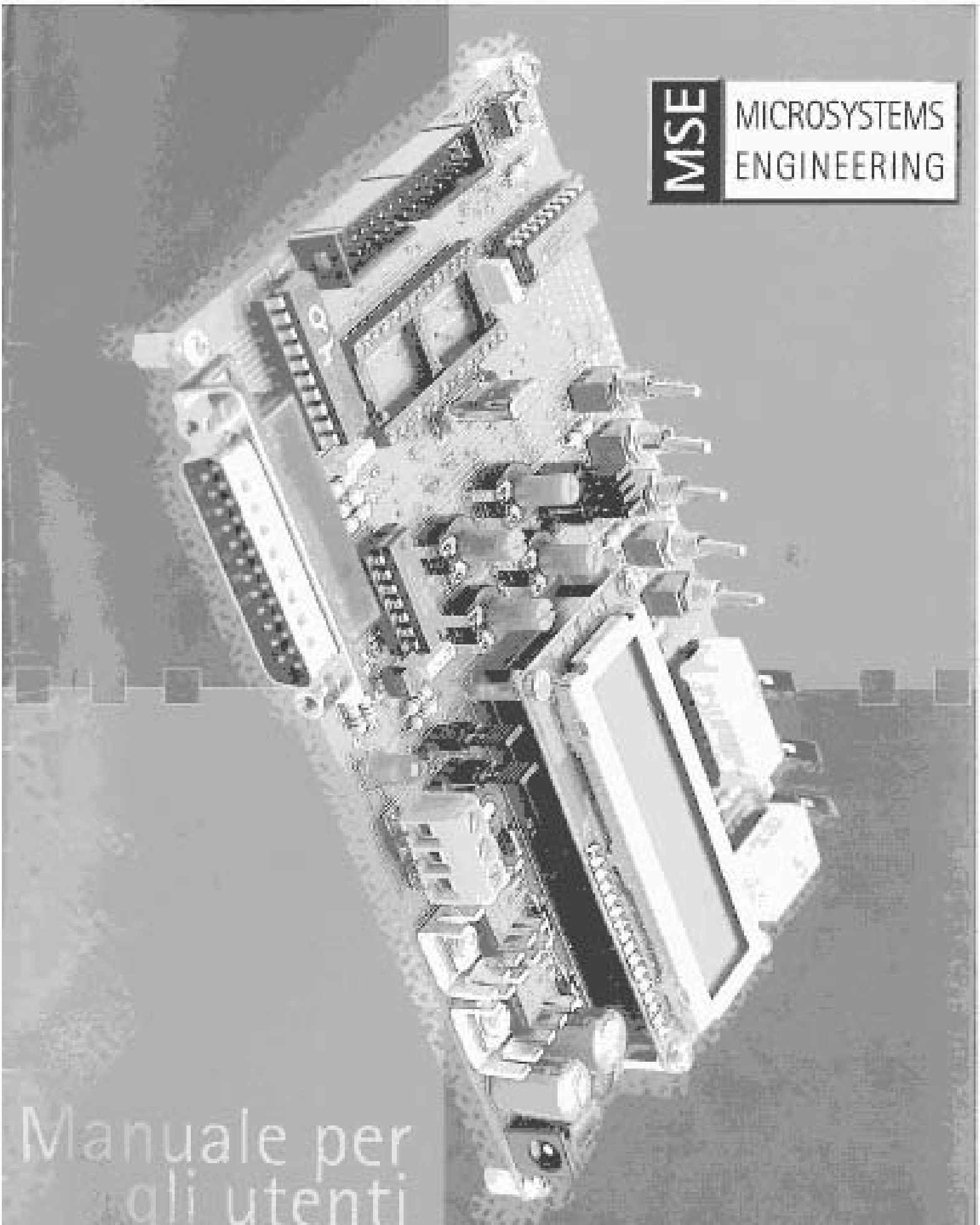
MSE MICROSYSTEMS ENGINEERING



Manuale per gli utenti

«MICRO PIC'TRAINER»

MANUALE PER GLI UTENTI DEL "MICRO PIC' TRAINER"

INDICE

MSE MICROSYSTEMS ENGINEERING

CAPITOLO 1: Hardware del µPIC Trainer

Nella figura 1-1 è mostrato lo schema elettrico delle connessioni del sistema di valutazione µPIC Trainer. Nei paragrafi successivi verranno spiegate le diverse sezioni che lo compongono, facendo riferimento ai diversi componenti di questo schema.

1.1 INTRODUZIONE

Il sistema µPIC Trainer consiste in una placca didattica di valutazione per applicazioni basate sui microcontrollers PIC di gamma media della ARIZONA MICROCHIPS INC.. È dotato di una serie di periferiche di base di E/U, con le quali poter verificare il funzionamento di una applicazione, ma anche l'insieme dei circuiti necessari per la registrazione di tutti i modelli di microcontrollers PIC di gamma media a 18 e 28 piedini.

Le caratteristiche del sistema sono le seguenti:

1.- *Alimentazione da un trasformatore da 12VAC o da 2 batterie da 9VDC. Il circuito di stabilizzazione è incluso nella placca. Con esso si ottiene la tensione generale di +5V (Vcc) e quella di registrazione di +13,8V.*
2.- *Supporta qualsiasi microcontroller di gamma media sia a 18 che a 28 piedini. La frequenza di lavoro di default è di 4MHz, che può essere modificata cambiando il cristallo di quarzo.*
3.- *Generazione automatica di RESET (POR) e manuale, mediante pulsante azionato dall'utente.*
4.- *5 entrate digitali attivabili mediante 5 commutatori.*
5.- *4 entrate di tensione analogica, variabili mediante potenziometri.*
6.- *8 uscite digitali connesse a 8 indicatori luminosi tipo LED e/o a display a 7 segmenti. Mediante dei jumpers si seleziona un tipo o l'altro.*
7.- *Connessione di un modulo di visualizzazione tipo LCD di 2 X 16 (2 linee di 16 caratteri ognuno) e di controllo del contrasto mediante potenziometro.*
8.- *Sia le periferiche di entrata che quelle di uscita si possono attivare/disattivare mediante i rispettivi jumpers. Vengono disconnesse quelle che non sono utilizzate in una determinata applicazione, con l'obiettivo di ridurre il carico elettrico nelle linee di E/U del PIC.*
9.- *Connettore di espansione PIC-BUS, dove tutti i segnali di E/U del microcontroller sotto prova sono disponibili per essere utilizzati da altre periferiche diverse da quelle di cui è dotato µPIC Trainer.*
10.- *Il segnale RA4/TOCKI può essere utilizzato dall'esterno come entrata o uscita digitale, o come entrata degli impulsi di clock per il timer 0.*
11.- *Il segnale RB0/INT si può utilizzare come entrata o uscita digitale, o come entrata di interrupt esterno.*
12.- *Circuito di registrazione "in circuito". Consente, con il necessario software di controllo, di registrare il PIC che in quel momento è inserito in uno qualsiasi degli zoccoli. È particolarmente utile per il modello 16C84 che essendo dotato di memoria EEPROM di programma, può essere registrato e riregistrato tutte le volte che è necessario, senza doverlo estrarre dallo zoccolo.*
13.- *LED indicatore della presenza della tensione Vpp di registrazione.*
14.- *Connettore DB25 per la connessione con il canale parallelo di qualsiasi personal computer.*
15.- *Software in italiano per effettuare tutte le operazioni tipiche di registrazione: lettura, registrazione, verifica, cancellazione, modifica, ecc.*
16.- *Tutti i componenti che formano il µPIC Trainer si inseriscono su una placca di circuito stampato di tipo professionale. Il tutto viene fornito completamente montato, o in forma di KIT, in modo che sia l'utente a poterlo assemblare.*

1.2 LA FONTE DI ALIMENTAZIONE

Si incarica di ottenere le tensioni stabilizzate interne di lavoro: alimentazione generale di +5V (Vcc) e tensione di registrazione di +13,8V.

La tensione di entrata viene fornita mediante 2 batterie da 9V, oppure con un trasformatore da 12VAC connesso a J1. In questo caso, la tensione alternata viene rettificata mediante il ponte di diodi D1 e viene filtrata mediante i condensatori C1 e C2, al fine di ottenerne una continua senza doverla stabilizzare.

In ogni caso, la tensione continua così ottenuta si stabilizza a +5V mediante il regolatore U6 – uA7805 e a +13,8V mediante il regolatore uA7812, assieme ai 3 diodi D2-D4 1N4007.

1.3 IL MICROCONTROLLER

Il sistema µPIC Trainer è progettato per lavorare con qualsiasi modello di microcontroller PIC di gamma media incapsulato a 18 o 28 piedini. Questo è il motivo dell'esistenza di 2 zoccoli: supportare entrambi i tipi di capsule.

In base allo schema della figura 1-1, la differenza di base tra i due modelli a 18 e 28 piedini consiste nel fatto che nel primo sono disponibili una porta A a 5 linee (RA0-RA4) e una porta B a 8 linee (RB0-RB7), mentre nel secondo sono disponibili una porta A a 6 linee (RA0-RA5), una porta B a 8 linee (RB0-RB7) e una porta C a 8 linee (RC0-RC7).

Nel µPIC Trainer si usano le 5 linee della porta A e le 8 della porta B, che sono comuni ai 2 modelli di PICs. In ogni caso, le linee di queste porte, assieme a quelle della porta C, sono disponibili per l'utente attraverso il connettore PIC-BUS di espansione.

La frequenza di lavoro di entrambi i 2 tipi di PICs viene stabilita dal cristallo al quarzo X1 e i condensatori C6 e C7. Tale frequenza, per default, è di 4MHz, anche se l'utente è in grado di modificarla cambiando i valori del cristallo e dei condensatori, secondo le caratteristiche tecniche del modello di PIC scelto.

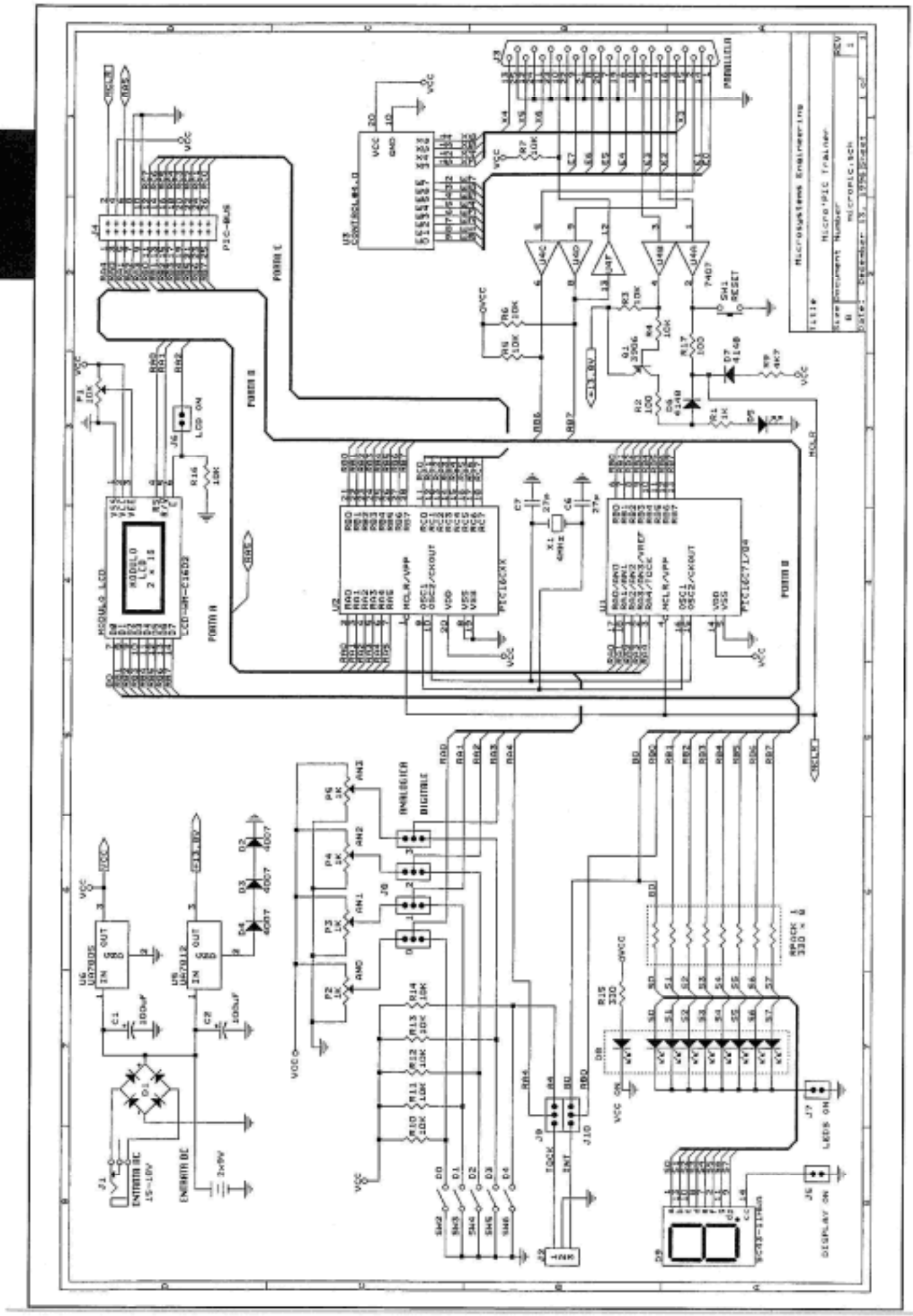


Figura 1-1. Schema elettrico del µPIC Trainer

1.4 ENTRATE DIGITALI

Sono formate da 5 interruttori (SW2-SW6) connessi alle linee RA0-RA4 della porta A, in grado di introdurre livelli logici "1" e "0" per tali linee. Le entrate possono essere programmate per agire in modi diversi. Pertanto, RA0-RA3 possono agire anche come entrate analogiche e RA4 come entrata esterna degli impulsi di clock per il timer 0 (TOCKI).

Queste linee sono associate ad alcuni jumpers, con i quali è possibile selezionare il tipo di entrata che gli verrà applicata. I jumpers J8(0) – J8(3) selezionano individualmente se attraverso RA0-RA3 saranno inserite entrate digitali con gli interruttori SW2-SW5, entrate analogiche con i potenziometri P2-P5 o, semplicemente, se tali linee resteranno sconnesse sia dagli interruttori che dai potenziometri, potendo così essere usate con altre periferiche diverse.

Con il jumper J9 è possibile selezionare nel caso RA4 funzioni come entrata digitale proveniente dall'interruttore SW6, come entrata del segnale di clock per il timer 0 (TOCKI) che si applica dall'esterno mediante il connettore J2(1) o, semplicemente, se questa linea resta non connessa e può essere così usata con un'altra periferica diversa.

1.5 ENTRATE ANALOGICHE

Sono formate da 4 resistenze variabili o potenziometri (P2-P5). Questi si alimentano con la tensione generale di +5V. In funzione di come si muove l'asse di uno qualsiasi di questi potenziometri, si ottiene una tensione variabile tra 0 e 5V nei loro terminali centrali (cursori).

La tensione variabile presente in uno qualsiasi dei 4 cursori dei potenziometri P2-P5 si fermerà rispettivamente ai jumpers J8(0)-J8(3). Se uno qualsiasi di essi è nella posizione "analogica", la linea corrispondente (RA0-RA3) riceverà questa tensione per la sua successiva elaborazione software.

È opportuno ricordare che, sebbene a livello di hardware una o più linee siano configurate come entrate analogiche o digitali mediante i jumpers J8(0)-J8(3), il software di controllo del PIC deve essere perfettamente in accordo, programmando adeguatamente in ogni momento le caratteristiche di tali linee di entrata.

1.6 USCITE DIGITALI

Sono connesse alla porta B e consistono, da un lato, in una barra a diodi LEDs luminosi (D8), che rappresentano lo stato logico dei segnali RB0-RB7 e, dall'altro lato, in un display numerico a 7 segmenti.

Queste periferiche sono connesse tra loro in parallelo alla porta B. Per evitare un eccesso di consumo in questa porta, sia i LEDs che il display sono dotati rispettivamente dei jumpers J7 e J5, con cui possono essere sconnessi se l'applicazione aperta non utilizza uno dei due.

Il pack di resistenze RPACK 1 contiene 8 resistenze da 330W che agiscono come resistenze di assorbimento o limitazione.

La barra a diodi luminosi D8 contiene 10 diodi tipo LED. I primi 8 (S0-S7) rappresentano, nel caso siano attivati (J7), lo stato dei segnali RB0-RB7, il nono LED non si usa e il decimo serve per indicare la presenza della tensione di +5V (Vcc ON).

La linea RB0 può funzionare come entrata di interrupt esterna INT. Tale segnale di entrata viene fornito attraverso il connettore J2(3). Il segnale RB0 viene connesso con il jumper J10 ai LEDs e al display, all'entrata di interrupt INT o, semplicemente, si lascia senza connessione.

Come è già stato spiegato, le periferiche connesse alla porta B possono essere disattivate mediante i rispettivi jumpers. In questo modo, non hanno alcun effetto su questa porta e, pertanto, i segnali RB0-RB7 disponibili nel connettore di espansione PIC-BUS possono essere impiegati per un altro tipo di periferica in altre configurazioni.

1.7 IL MODULO LCD

Si tratta di un modulo di visualizzazione alfanumerica a cristalli liquidi, in grado di rappresentare 2 linee di 16 caratteri ognuna.

Inserendo i codici necessari, è possibile creare diversi effetti di visualizzazione come battito di ciglia, scroll, attivazione di un cursore, ecc. È possibile anche la creazione di caratteri differenti, definiti dall'utente. Il capitolo 4 spiega il funzionamento di questo modulo.

Da un lato, le 8 linee dei dati D0-D7 sono connesse con le 8 linee della porta B (RB0-RB7). Tale porta, a volte, funziona come uscita del PIC e come entrata verso il modulo. Attraverso essa vengono inseriti i diversi codici di controllo per la realizzazione di differenti effetti di visualizzazione e anche i codici ASCII dei caratteri da visualizzare. Altre volte, la porta B deve funzionare come entrata verso il PIC, dato che suo tramite il modulo LCD mostra i codici che indicano il suo stato interno, il contenuto del buffer di memoria interna, ecc.

Dall'altro lato, il modulo è connesso alle linee RA0, RA1 e RA2 della porta A del PIC. Queste linee funzionano come uscita e sono utilizzate per inviare al modulo i seguenti segnali di controllo:

RS=0	Selezione del registro delle istruzioni
RS=1	Selezione del registro dei dati
R/W=0	Ciclo di scrittura sul modulo LCD
R/W=1	Ciclo di lettura del modulo LCD
E=0	Modulo LCD disattivato
E=1	Modulo LCD attivato

Bisogna sottolineare che quest'ultimo segnale E è connesso a RA2 mediante il jumper J6. Se tale jumper è chiuso, RA2 controllerà l'attivazione o meno del modulo LCD. Al contrario, se è aperto, il segnale E resta connesso a terra (livello "0") attraverso R16. In questo modo, il modulo è sempre sconnesso e le linee sia della porta A (RA0-RA2) che della B (RB0-RB7) possono essere usate per altre periferiche, siano esse quelle disponibili nel µPIC Trainer, oppure qualsiasi altra connessa al connettore di espansione PIC-BUS.

Per concludere, si sottolinea come il modulo LCD sia una periferica di visualizzazione molto usata nelle applicazioni reali. Si tratta di una forma di visualizzazione alfanumerica, ma anche grafica, comoda, piacevole e versatile.

1.8 IL CIRCUITO DI REGISTRAZIONE

Si tratta di una delle sezioni più interessanti del sistema µPIC Trainer. La sua importanza consiste nella possibilità di effettuare la registrazione del PIC "in circuito". Ciò significa che il PIC può essere trattato (letto, registrato, cancellato, ecc.) inserito direttamente nel suo corrispondente zoccolo.

Questo fattore è ancora più rilevante se si utilizza il modello 16F84 di PIC. Questo modello dispone di memoria FLASH come memoria di programma. Nonostante non sia volatile, questa memoria può essere registrata e cancellata sino a quasi un migliaio di volte. Pertanto, questo PIC inserito nel corrispondente zoccolo potrà essere registrato con molteplici programmi e applicazioni nello stesso istante in cui si stanno lanciando e provando direttamente sul µPIC Trainer.

In ogni caso, il circuito di registrazione consente di registrare tutti i modelli a 18 e 28 piedini della gamma media dei PICs. Attraverso il connettore J3, il circuito viene connesso al canale parallelo di qualsiasi personal computer del tipo PC/XT/AT. Il software di controllo, totalmente in italiano e compreso nel kit µPIC Trainer, si incarica di gestire tutte le operazioni, quali leggere il contenuto del PIC (se non è protetto), registrare, verificare, cancellare (solo il 16F84), definire la parola di configurazione, ecc. Il capitolo 3 è dedicato alla spiegazione e all'uso del software di registrazione.

Come è già stato accennato, la connessione tra il PC e µPIC Trainer avviene attraverso il canale parallelo. Il circuito integrato U3-CONTROL#4.0 si incarica di verificare l'hardware del µPIC Trainer e la comunicazione tra questo e il PC.

La lettura/registrazione dei PICs di gamma media è realizzata in serie. I bits dei dati si leggono o scrivono in sequenza uno dopo l'altro attraverso il segnale RB7 e vengono forniti al canale parallelo mediante gli amplificatori U4D e U4F, per il successivo trattamento con il software di controllo.

Il segnale RB6 funziona come entrata di clock per sincronizzare i bits dei dati durante il processo di lettura/registrazione del PIC. Questo segnale di clock è generato dal software di controllo e viene fornito per il canale parallelo attraverso l'amplificatore U4C.

Il software di controllo genera anche un segnale che, mediante U4B, viene fornito al transistor di commutazione Q1. Quando si attiva, questo transistor invia i 13,8V generati dalla fonte di alimentazione al piedino MCLR/VPP del PIC che è in lettura o in registrazione. Contemporaneamente, il diodo LED DS si illumina, indicando che esiste attività nel circuito di registrazione del µPIC Trainer.

Infine, attraverso U4A, il software di controllo genera il segnale MCLR che resetta e fa ripartire il microcontroller. Normalmente, ciò avviene quando si conclude qualsiasi operazione di lettura e/o registrazione e il PIC inizia immediatamente a lanciare il programma contenuto al suo interno. Bisogna sottolineare come in ogni caso, e in qualsiasi momento, l'utente può effettuare manualmente il reset, agendo sul pulsante RESET SW1.

Dato che i segnali utilizzati per la lettura/registrazione di un PIC sono RB6 e RB7 e che questi a loro volta sono connessi ai LEDs, al display e al modulo LCD, è importante disconnettere queste periferiche mediante i rispettivi jumpers J7, J5 e J6, in modo che non inviino alcuna carica a tali segnali, alterando così l'informazione letta o registrata.

1.9 IL CONNETTORE DI ESPANSIONE

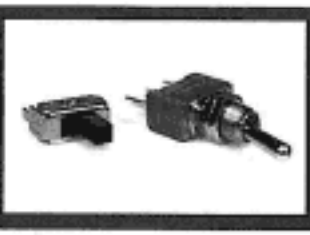
Come è stato accennato nei paragrafi precedenti, il µPIC Trainer è dotato di un connettore di espansione chiamato PIC-BUS.

Si tratta di un connettore a 26 contatti in cui sono disponibili tutti i segnali delle diverse porte di E/U, oltre a quelli di alimentazione e di MCLR. Ciò consente di collegare più schede di espansione sviluppate dalla Microsystems Engineering, o dall'utente stesso, disponendo così di nuove periferiche.

Bisogna aggiungere che perché non ci sia conflitto tra le nuove periferiche connesse al PIC-BUS e quelle già esistenti nel µPIC Trainer, è necessario disconnettere mediante i corrispondenti jumpers quelle che saranno sostituite dalle nuove espansioni.

AVVERTENZA

La scheda Micro'PIC Trainer può essere dotata di due tipi di interruttori per i segnali RA0-RA4: a leva o a cursore. Nel caso di quelli a leva, il livello logico "1" si ottiene spostando la leva verso l'alto (come viene indicato nel manuale e nella serigrafia della scheda). Al contrario, nel caso degli interruttori a cursore il livello logico "1" si ottiene spostando la leva verso il basso.



CAPITOLO 2: Montaggio del µPIC Trainer

2.1 INTRODUZIONE

Questo capitolo è dedicato al montaggio del sistema di valutazione del PICs µPIC Trainer. Anche se il dispositivo è venduto montato e collaudato, è possibile acquistarlo in kit. Si tratta di una piastra professionale formato EUROCARD di 100 X 160, a doppio strato con fori metallizzati, maschera antisaldante e serigrafia per la collocazione dei componenti nel punto e nella posizione giuste.

Si consiglia di utilizzare un saldatore a stilo a punta fine e con una potenza non superiore ai 30W. Lo stagno deve essere almeno al 60% e di uno spessore di 1 mm come massimo.

Nella figura 2-1 è mostrata la disposizione e l'orientamento dei componenti sulla piastra del circuito stampato.

2.2 ELENCO DEI MATERIALI

Di seguito è presentato l'elenco dei materiali necessari per il montaggio e il funzionamento del sistema µPIC Trainer.

RIF.	DESCRIZIONE
CIRCUITI INTEGRATI	
U1	PIC 16C84. A scelta, altro modello di PIC a 18 piedini.
U2	Modello di PIC a 28 piedini (opzionale).
U3	Circuito di controllo CONTROL#4.0.
U4	SN7407. Amplificatore.
U5	uA7812. Stabilizzatore di tensione a 12V.
U6	uA7805. Stabilizzatore di tensione a 5V.
DIODI E TRANSISTOR	
D1	Ponte rettificatore da 1A.
D2-D4	3 diodi modello 1N4007.
D5	Diodo luminoso tipo LED di colore rosso da 5 mm.
D6-D7	2 diodi modello 1N4148.

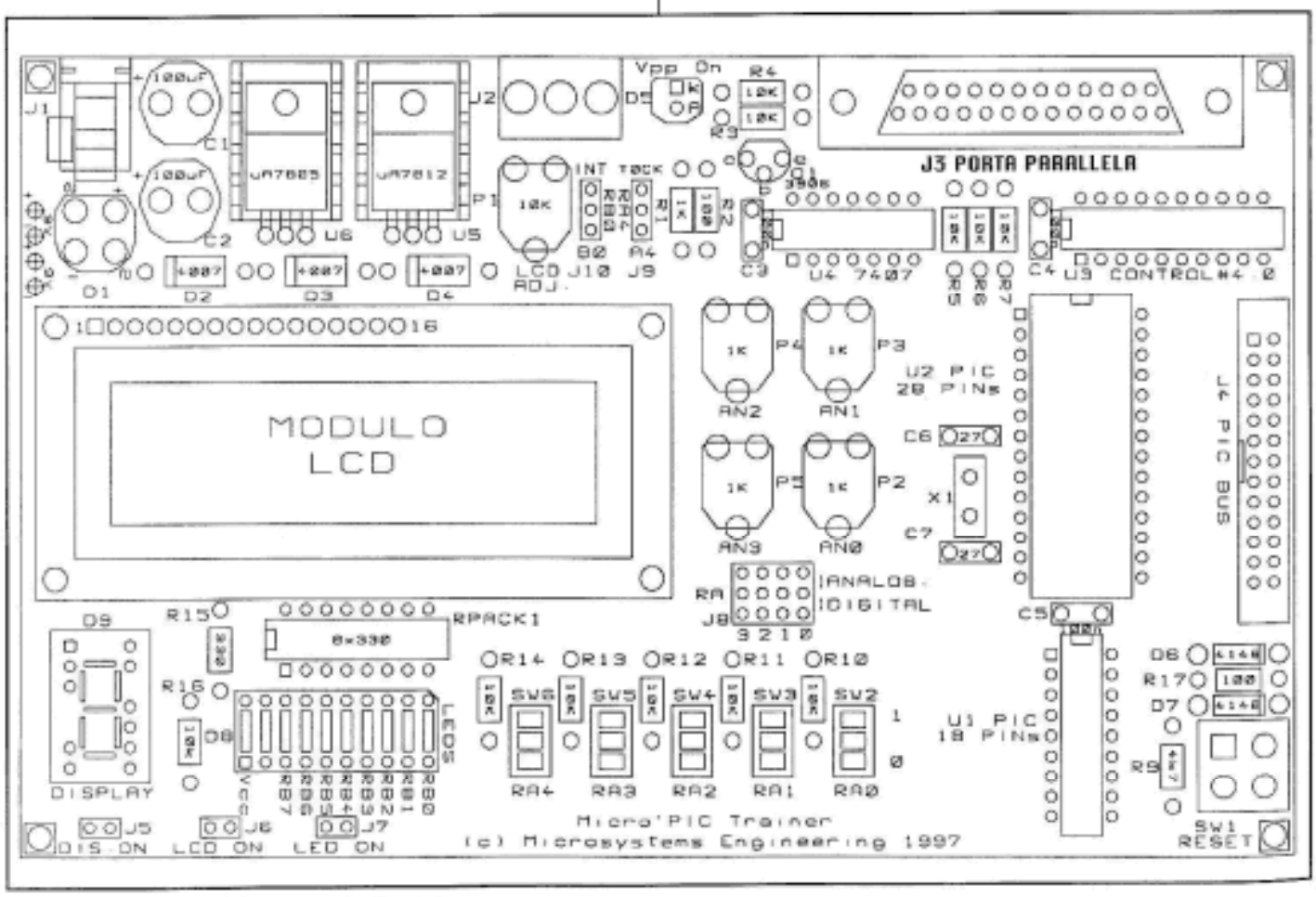


Figura 2-1. *Disposizione dei componenti sulla piastra.*

D8	1 barra DIL di 10 diodi LED Kingbright DC-10EWA.
D9	Display a 7 segmenti a catodo comune Kingbright SC43-11HWA.
Q1	Transistor PNP 3906.

RESISTENZE

R1	Resistenza da 1K – 1/4W (marrone-nero-rosso).
R2	Resistenza da 100 – 1/4W (marrone-nero-marrone).
R3-R7	5 resistenze da 10K – 1/4W (marrone-nero-arancione).
R9	Resistenza da 4K7 – 1/4W (giallo-viola-rosso).
R10-R14	5 resistenze da 10K – 1/4W (marrone-nero-arancione).
R15	Resistenza da 330 – 1/4W (arancione-arancione-marrone).
R16	Resistenza da 10K – 1/4W (marrone-nero-arancione).
R17	Resistenza da 100 – 1/4W (marrone-nero-marrone).
RPACK1	Pack di 8 resistenze DIL (2 x 8) da 330.
P1	Resistenza variabile da 10K.
P2-P5	4 resistenze variabili da 1K per comandi o cursori.

CONDENSATORI

C1-C2	2 condensatori elettrolitici da 100mF/63V radiali.
C3-C5	3 condensatori da 100nF.
C6-C7	2 condensatori da 27pF.

MATERIALE VARIO

X1	Cristallo al quarzo a 4MHz.
SW1	Pulsante di RESET.
SW2-SW6	5 commutatori a leva.
J1	Base di alimentazione per l'entrata di ACV.
J2	Morsetto a 3 contatti per circuito stampato da 5.08.
J3	Connettore DB25 maschio, gomito di bassa sezione per circuito stampato.
J4	Connettore a 26 contatti (2 x 13), maschio dritto.

1 zoccolo a 28 piedini curvi.
2 zoccoli a 20 piedini curvi.
1 zoccolo a 18 piedini curvi.
2 zoccoli a 14 piedini curvi.
1 striscia a 40 contatti diritti dorati.
Mezza striscia a 18 contatti femmina curvi.
9 cappucci o jumpers.
4 separatori metallici esagonali da 7 mm, diametro 10.
4 separatori metallici esagonali da 10 mm, diametro 10.
10 dadi, diametro 10.
6 viti da 6 mm, diametro 10.
2 alette di raffreddamento per i circuiti di stabilizzazione.
4 cursori per i potenziometri P2-P5.
1 modulo LCD 2 x 16 WM-C1606M.
1 piastra di circuito stampato.
2 cavi portabatterie da 9V.
1 cavo a 25 fili maschio-femmina per il canale parallelo.
1 manuale di istruzioni.
1 dischetto con il software di registrazione.
1 trasformatore da 12VAC (opzionale).
2 batterie da 9V (opzionali).

2.3 Fasi da seguire nel montaggio

Con l'obiettivo di semplificare il montaggio del µPIC Trainer, per quegli utenti che non sono esperti di "bricolage" elettronico, di seguito è presentato l'ordine con cui devono essere collocati i diversi elementi sul lato dei componenti della piastra del circuito stampato.

1° La prima cosa da fare è identificare, separare e classificare tutti gli elementi che formano il kit, sulla base dell'elenco dei componenti presentata nel paragrafo precedente, facendo attenzione a formati, valori, riferimenti, ecc. e cercando di non perdere nulla.

2° Si deve iniziare sistemando tutte le resistenze, dal momento che sono i componenti con minor ingombro. Il loro valore viene identificato con i colori:

VALORE	COLORI	RIFERIMENTI
100	marrone-nero-marrone	R2, R17
330	arancione-arancione-marrone	R15
1K	marrone-nero-rosso	R1
4K7	arancione-viola-rosso	R9
10K	marrone-nero-arancione	R3-R7, R10-R14 e R16

Si piegano i piedini a 90° e si inseriscono al loro posto, secondo il loro valore. Guardare la serigrafia della piastra o la figura 2-1 per verificare la collocazione esatta di ognuna.

Devono restare tutte a livello della piastra e, dopo aver effettuato le saldature, sul lato delle saldature della piastra del circuito stampato, si possono tagliare i pezzi di piedino che fuoriescono dal livello della saldatura.

3° Collocare i diodi D6 e D7 (1N4148), considerando che la frangia disegnata su una delle estremità corrisponde al piedino del catodo e deve essere orientata secondo la serigrafia della piastra e al suo livello. Tagliare il pezzo dei piedini che avanza.

4° Collocare i 3 diodi D2-D4 (1N4007), seguendo le stesse indicazioni della fase precedente.

5° Collocare gli zoccoli dei circuiti integrati. Indipendentemente dal modello di zoccolo disponibile, viene sempre rappresentato mediante qualche riferimento, di solito quale è il piedino numero 1. Questo riferimento deve a sua volta coincidere con il riferimento riportato sulla serigrafia del circuito stampato.

Nello zoccolo dove verrà inserito il display sarà necessario tagliare alcuni piedini al livello del display stesso, in modo che possa essere inserito nella piastra del circuito stampato.

È consigliabile, prima di tutto, fissare, a livello della piastra, tutti gli zoccoli mediante due punti di saldatura ognuno. Quindi assicurarsi che siano tutti perfettamente allineati e orientati e che tutti i piedini fuoriescano nel lato opposto della piastra (a volte

MONTAGGIO

alcuni si piegano, non fuoriescono e non si possono saldare). Infine, saldare tutti i piedini, evitando assolutamente che si tocchino tra loro.

6° Collocare il ponte rettificatore D1. Osservare l'orientamento che devono avere i 4 piedini (~, ~, +, -) rispetto alla serigrafia della piastra e tenerlo allo stesso livello. Tagliare i pezzi di piedino che avanzano.

7° Saldare a livello della piastra i 2 condensatori da 27pF, C6 e C7. Non importa l'orientamento. Tagliare i pezzi di piedino che avanzano.

8° Saldare il pack di resistenze RPACK1. Il riferimento deve coincidere con quello della serigrafia.

9° Collocare le 5 resistenze variabili o potenziometri. Esiste un solo modo possibile di inserirle. P1 è da 10K e P2-P5 sono da 1K. Sistemarle correttamente nei loro posti.

10° Collocare il transistor Q1 secondo il profilo disegnato sulla serigrafia della piastra. Deve restare a circa 1 cm di altezza rispetto alla piastra. Tagliare i pezzi di piedino che avanzano

11° Collocare il diodo LED D5 a livello della piastra. Il piedino corto corrisponde al catodo. Tagliare i pezzi di piedino che avanzano.

12° Collocare i 3 condensatori di disaccoppiamento C3-C5 da 100nF, a livello della piastra. Tagliare i pezzi di piedino che avanzano.

13° Collocare il pulsante di RESET SW1. Deve essere inserito in modo che 2 dei sui piedini restino orientati verso il bordo destro della piastra e gli altri 2 verso la resistenza R9.

Prima di inserire questo pulsante sarà necessario orientare i suoi piedini con l'aiuto di una piccola pinzetta.

14° Scegliere la striscia da 40 contatti dorati e tagliare, con estrema cura, 6 pezzi da 3 contatti e 3 pezzi da 2. Il resto della striscia con i 16 contatti deve essere conservata per una fase successiva.

15° I 6 pezzi da 3 contatti devono essere collocati e saldati i più diritti possibile nei punti destinati ai jumpers J8(0), J8(1), J8(2), J8(3), J9 e J10.

16° I 3 pezzi da 2 contatti devono essere collocati e saldati i più diritti possibile nei punti destinati ai jumpers J5, J6 e J7.

17° Saldare il connettore J1 di entrata ACV dell'alimentazione.

18° Saldare il connettore J4 a 26 contatti. Il segno di riferimento (normalmente un piccolo triangolo) che indica il contatto numero 1, deve essere mirato verso l'interno della piastra e non verso il bordo.

19° Collocare e saldare il connettore J2. Si tratta di un morsetto a 3 connettori per circuito stampato. Deve essere orientato in modo che le connessioni possano essere realizzate dal bordo esterno della piastra.

20° Collocare e saldare il connettore J3. Si tratta del connettore DB25 che servirà da interfaccia con la porta parallela del computer. Esiste un unico modo di inserimento, ma bisogna assicurarsi che sia perfettamente a livello della placca.

21° Collocare lo stabilizzatore di tensione U6 (uA7805). Piegare i 3 piedini ad angolo retto verso il basso. Collocare l'aletta di raffreddamento a livello della piastra e sopra questo lo stabilizzatore, in modo che i 3 piedini si inseriscano nei 3 fori e, contemporaneamente, che il foro grande della capsula dello stabilizzatore coincida con quello dell'aletta di raffreddamento e questo, a sua volta, con quello della piastra. Passare una vite da 6 mm e fissare tutti i componenti mediante un dado sull'altro lato della piastra e quindi saldarlo. Tagliare i pezzi di piedino che avanzano. Vedere la figura 2-2.

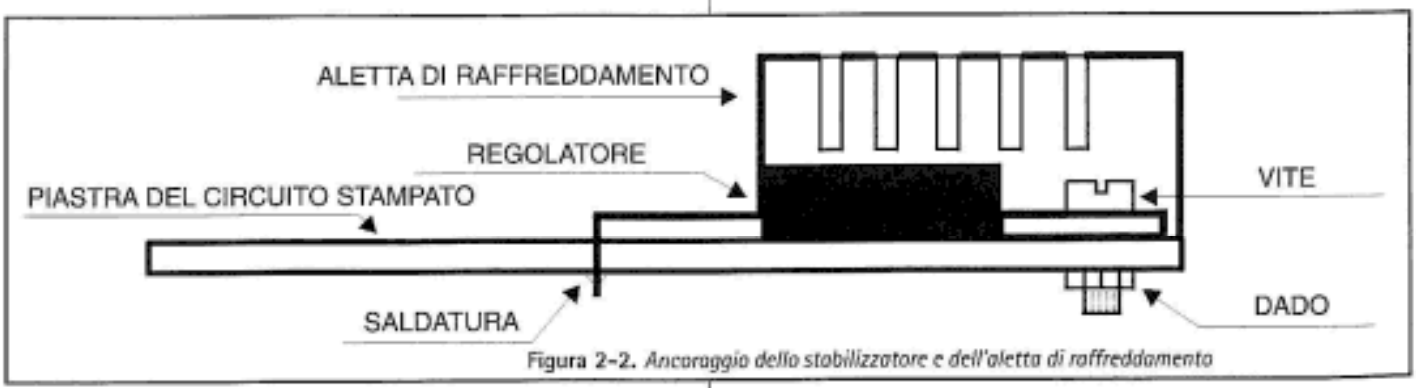


Figura 2-2. *Ancoraggio dello stabilizzatore e dell'aletta di raffreddamento*

22° Collocare, seguendo le indicazioni del punto precedente, lo stabilizzatore di tensione U5 (uA7812).

23° Collocare il cristallo al quarzo X1. Non importa l'orientamento, ma deve essere a livello della piastra. Tagliare i pezzi di piedino che avanzano.

MONTAGGIO

24° Collocare i condensatori C1 e C2 a livello della piastra. Uno dei loro piedini è segnato con il simbolo (-) e deve essere orientato verso gli stabilizzatori.

25° Saldare la striscia da 16 contatti curvi femmina nel punto dove dovrà essere collocato il modulo LCD. Fare in modo che resti ad angolo retto rispetto alla piastra.

26° Saldare la striscia da 16 contatti diritti, avanzata al punto 14, al modulo LCD. Le estremità corte di questa striscia devono essere introdotte nel modulo dalla parte inferiore e saldate sulla sua parte superiore. Fare in modo che questa striscia formi un angolo retto rispetto alla piastra del modulo LCD.

27° Fissare con 4 dadi i 4 separatori da 10 mm introdotti nei 4 fori di ancoraggio del modulo LCD.

Inserire il modulo LCD. La striscia di contatti diritti saldata al modulo deve coincidere con la striscia di contatti femmina saldata alla piastra del µPIC Trainer. I 4 separatori devono coincidere con i 4 fori che si trovano ai 4 angoli del modulo LCD. È probabile che questi ultimi debbano essere leggermente allargati con le punte delle forbici, o meglio ancora con un punteruolo da 3 mm. Infine, fissare tutto l'insieme con 4 viti da 6 mm.

28° Collocare e saldare a livello della piastra i 5 commutatori a leva SW2-SW6. Fare in modo che restino ben allineati tra loro.

29° Inserire i circuiti U3 (CONTROL#4.0) e U4 (SN7407) nei corrispondenti zoccoli, senza confondere i rispettivi riferimenti. Fare in modo che i piedini siano ben allineati e non esercitare troppa forza per evitare che si pieghino.

30° Inserire il display D9 nello zoccolo. I punti decimali devono essere orientati verso il bordo inferiore della piastra. Fare in modo che i piedini siano ben allineati e non esercitare troppa forza per evitare che si pieghino.

31° Inserire la barra a LEDs D8 nel suo zoccolo. Se si fa attenzione, si può notare come uno dei suoi bordi sia smussato. Farlo coincidere con la serigrafia della piastra.

32° Collocare i 4 cursori di comando nei 4 potenziometri P2-P5.

33° Fissare, mediante i corrispondenti dadi, i 4 separatori da 7 mm nei 4 fori che si trovano agli angoli della piastra del µPIC Trainer.

34° Saldare, assieme al ponte rettificatore D1, i 2 portabatterie per le batterie da 9V di alimentazione.

35° Collocare i 9 cappucci o jumpers sui contatti, secondo la configurazione di E/U desiderata

JUMPER	POSIZIONE	DESCRIZIONE
J5	*Aperto*	*Display D9 sconnesso.*
	Chiuso	*Display D9 connesso alla porta B.*
J6	*Aperto*	*Modulo LCD sconnesso.*
	Chiuso	*Modulo LCD connesso alla porta B.*
J7	*Aperto*	*Barra a LEDs D8 sconnessa.*
	Chiuso	*Barra a LEDs D8 connessa alla porta B.*
J8(0)	*Aperto*	*Linea RA0 senza connessione.*
	Analogica	*Linea RA0 entrata analogica AN0.*
	Digitale	*Linea RA0 entrata digitale.*
J8(1)	*Aperto*	*Linea RA1 senza connessione.*
	Analogica	*Linea RA1 entrata analogica AN1.*
	Digitale	*Linea RA1 entrata digitale.*
J8(2)	*Aperto*	*Linea RA2 senza connessione.*
	Analogica	*Linea RA2 entrata analogica AN2.*
	Digitale	*Linea RA2 entrata digitale.*
J8(3)	*Aperto*	*Linea RA3 senza connessione.*
	Analogica	*Linea RA3 entrata analogica AN3.*
	Digitale	*Linea RA3 entrata digitale.*
J9	*Aperto*	*Linea RA4 senza connessione.*
	TOCKI	*Linea RA4 entrata di clock per il Timer 0.*
	A4	*Linea RA4 entrata digitale.*
J10	*Aperto*	*Linea RB0 senza connessione.*
	INT	*Linea RB0 entrata di interrupt esterna.*
	B0	*Linea RB0 uscita digitale.*

CAPITOLO 3: Il Software

3.1 INTRODUZIONE

Il circuito di registrazione del sistema µPIC Trainer è connesso a un personal computer tipo PC/XT/AT attraverso il canale parallelo. Il computer, a sua volta, ha bisogno di un software che gestisca questo canale parallelo per poter realizzare le differenti opzioni di registrazione.

In questo capitolo vengono spiegate l'installazione e l'uso di tale software, chiamato PICME-TR. Si tratta di un software in italiano progettato per lavorare con il sistema operativo MS-DOS. In un ambiente grafico con finestre e pulsanti, l'utente potrà selezionare le diverse opzioni disponibili con l'aiuto del mouse. Dalle pagine Internet del nostro sito è possibile scaricare versioni e esempi aggiornati.

Dato che il software PICME-TR effettua una costante verifica dell'hardware del µPIC Trainer, è necessario che questo sia connesso al canale parallelo del computer. Per lo stesso motivo, dato che nella registrazione vengono usate le linee RB7 e RB6 della porta B, è necessario che tutte le periferiche connesse a questa porta siano disabilitate.

3.2 REQUISITI MINIMI

I requisiti necessari perché il programma PICME-TR possa funzionare sono minimi. Qualsiasi personal computer anche non recente li soddisfa.

- *Personal computer con microprocessore 286 o superiore.*
- *Memoria RAM 640K minimo.*
- *Monitor DGA da 640 x 480 (si consiglia a colori).*
- *Mouse.*
- *Canale parallelo libero (LPT1:).*
- *Lettore di floppy disk da 3.5" da 1.4 Mb.*
- *Hard disk.*
- *Sistema operativo MS-DOS V 3.0 o superiore.*

3.3 INSTALLAZIONE

Il software PICME-TR viene fornito su un dischetto da 3.5" da 1.4 Mb. È preferibile lavorare sull'hard disk e quindi si consiglia di copiare il contenuto del dischetto in una subdirectory dove, inoltre, si trovi altro software, ad esempio l'assembler o il simulatore.

1. Creare una subdirectory di lavoro partendo dalla radice:
   MD PIC (ENTER)

2. Selezionare questa directory come directory attuale:
   CD PIC (ENTER)

3. Copiare tutti i file del disco PICME-TR inserito nell'unità A:
   COPY A:*.* (ENTER)

4. Alimentare e connettere il sistema µPIC Trainer al canale parallelo LPT1: del computer. Lanciare il programma:
   PICME-TR* (ENTER)

La schermata dovrà avere l'aspetto mostrato nella figura 3-1.

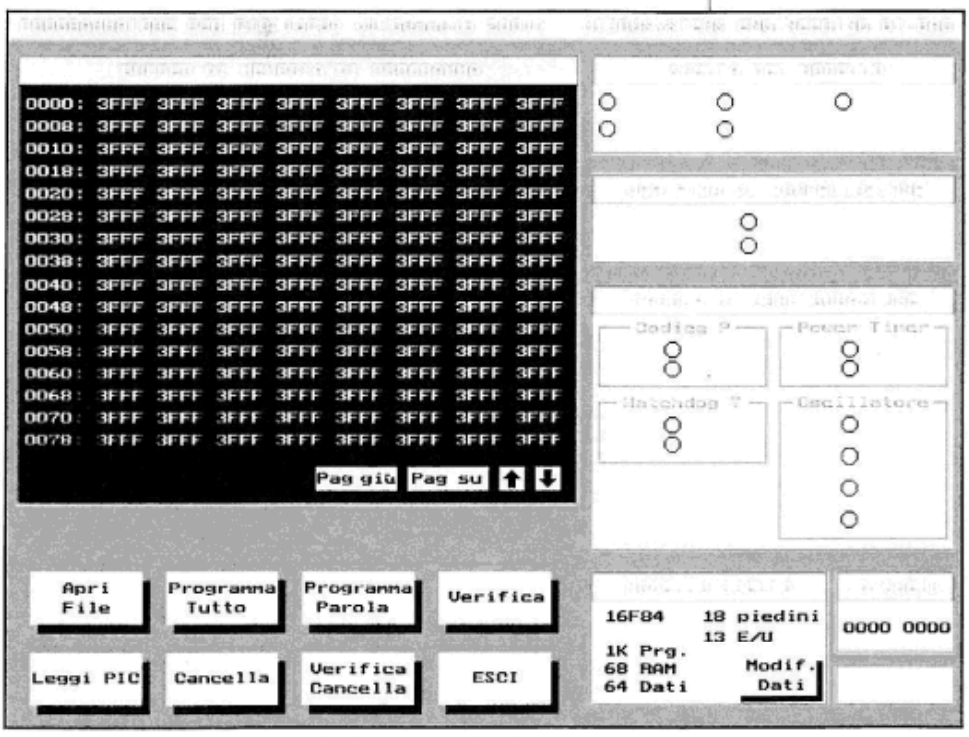


Figura 3-1. Aspetto della finestra del Programma PICME-TR

3.4 USO DEL PICME-TR

Guardando la figura 3-1, si può notare come l'utilizzo del programma sia molto intuitivo e sia sufficiente cliccare con il pulsante sinistro del mouse per selezionare una delle molteplici opzioni disponibili. La schermata è divisa in diverse aree o finestre.

Nel caso questa schermata non compaia, si deve ricontrollare la connessione tra il µPIC Trainer e il canale parallelo LPT1:. Per lo stesso motivo, dato che il software PICME-TR effettua un controllo dell'hardware, bisogna accertarsi del suo corretto stato.

3.4.1 Buffer di Memoria

Questa finestra compare in alto a sinistra e rappresenta o la memoria del programma del microcontroller scelto, oppure la memoria EEPROM dei dati, in funzione di quello che è attiva. Nella finestra vengono visualizzate le informazioni che verranno registrate nel chip e che sono contenute in un file eseguibile in formato INTEL-HEX. Questo file, a sua volta, si

ottiene come conseguenza dell'assemblaggio di un programma principale mediante uno dei molteplici assembler per PICs esistenti sul mercato.

In questo buffer possono comparire anche le informazioni contenute nella memoria di programma o EEPROM dei dati del PIC, nel caso questo sia stato letto e non sia protetto dalla lettura. Per default, la finestra attiva corrisponde alla memoria di programma e, se non è stato letto alcun file né tantomeno alcun PIC, apparirà completamente riempita con il dato 03FF.

Il buffer di memoria di programma è rappresentato da una serie di pagine di 16 linee di 8 words ognuna e può essere visualizzato nella sua totalità premendo i seguenti pulsanti:

Pag Giù	Vai alla pagina successiva
Pag Su	Ritorna alla pagina precedente
(:	Avanza di una linea
(:	Retrocedi di una linea

Si può anche intervenire sul contenuto del buffer per modificare una o più words, prima di procedere con la registrazione. È sufficiente selezionare una word con il mouse e modificare il suo valore.

Se il buffer di memoria mostra la memoria EEPROM dei dati (ciò è possibile solo nel modello 16C84), i 64 bytes disponibili saranno rappresentati in 8 linee di 8 words.

3.4.2 Scelta del Modello

In questa finestra l'utente può scegliere il modello di PIC con cui potrà lavorare. Il software, così come il sistema µPIC Trainer, permettono di usare qualsiasi modello di gamma media incapsulato a 18 o 28 piedini.

Con il pulsante sinistro del mouse si sceglie tra uno dei 4 gruppi di PICs disponibili. A loro volta, ognuno di questi gruppi può disporre di più modelli diversi. In tale ipotesi, caso, i modelli comparirebbero in un piccolo menu a scomparsa:

GRUPPO	MODELLI
16C6X	*16C61*
	16C62
	16C63
16C62X	*16C620*
	16C621
	16C622
16C7X	*16C71*
	16C73
16C84	*16C84*
16F84	*16F84*

3.4.3 Modello Scelto

È una finestra che compare nella parte inferiore dello schermo. Essa informa sulle caratteristiche più rilevanti del modello scelto:

N° di piedini
N° di linee di E/U
Dimensione della memoria di programma
Dimensione della memoria RAM dei dati

Il modello 16C84 è dotato inoltre di una memoria EEPROM per i dati da 64 bytes di capacità. Nel caso particolare in cui venga selezionato questo modello, nella finestra verrà indicata anche la dimensione di tale memoria EEPROM dei dati (disponibile solo nel 16C84), assieme al pulsante MODIFICA DATI, che fa in modo che la finestra del buffer di memoria attiva corrisponda al buffer di memoria EEPROM.

Questa finestra mostra i 64 bytes dell'area EEPROM dei dati, organizzati in 8 linee di 8 words ciascuna e può essere modificata in qualsiasi momento.

Attivando i pulsanti che si trovano nella parte inferiore della finestra si possono effettuare le seguenti operazioni:

Apri File:	Legge un file su disco, in formato INTEL-HEX, il cui contenuto viene mostrato nel buffer della memoria dei dati.
Leggi PIC:	Legge il contenuto della EEPROM dell'area dei dati del PIC 16C84 e lo mostra nel buffer di memoria dei dati.
Programma:	Registra fisicamente il contenuto del buffer di memoria dei dati sull'area dei dati EEPROM del PIC.
Cancella:	Cancella tutta la EEPROM dell'area dei dati del PIC.
Verifica:	Compara il contenuto del buffer di memoria dei dati con quello della EEPROM dei dati del PIC, per verificare la sua corretta registrazione.
Accetta:	Chiude la finestra di modifica dell'area dei dati. Lo schermo ha lo stesso aspetto di quello che compare quando si lancia il programma PICME-TR.

3.4.4 Verifica della Cancellazione

Attivando questa opzione, il software PICME-TR si incarica di assicurarsi che il PIC che dovrà essere registrato sia completamente cancellato.

In questo modo, il processo di registrazione viene un po' rallentato, dato che il PIC deve essere prima letto, ma in cambio si ottiene la completa sicurezza di registrare su un PIC totalmente cancellato.

Se viene trovato qualche punto non cancellato, nella finestra comparirà un messaggio di avviso in cui si chiede una conferma nel caso si voglia continuare con la registrazione.

Se questa opzione viene disattivata, il processo di registrazione è più rapido, ma è possibile che si sovrascriva un PIC che conteneva delle informazioni. In questo caso, il risultato finale della registrazione potrebbe non essere perfetto.

IMPORTANTE: guardare la presentazione zoccoli e adattatori per lavorare con diversi microcontroller e con i PIC 16F87X

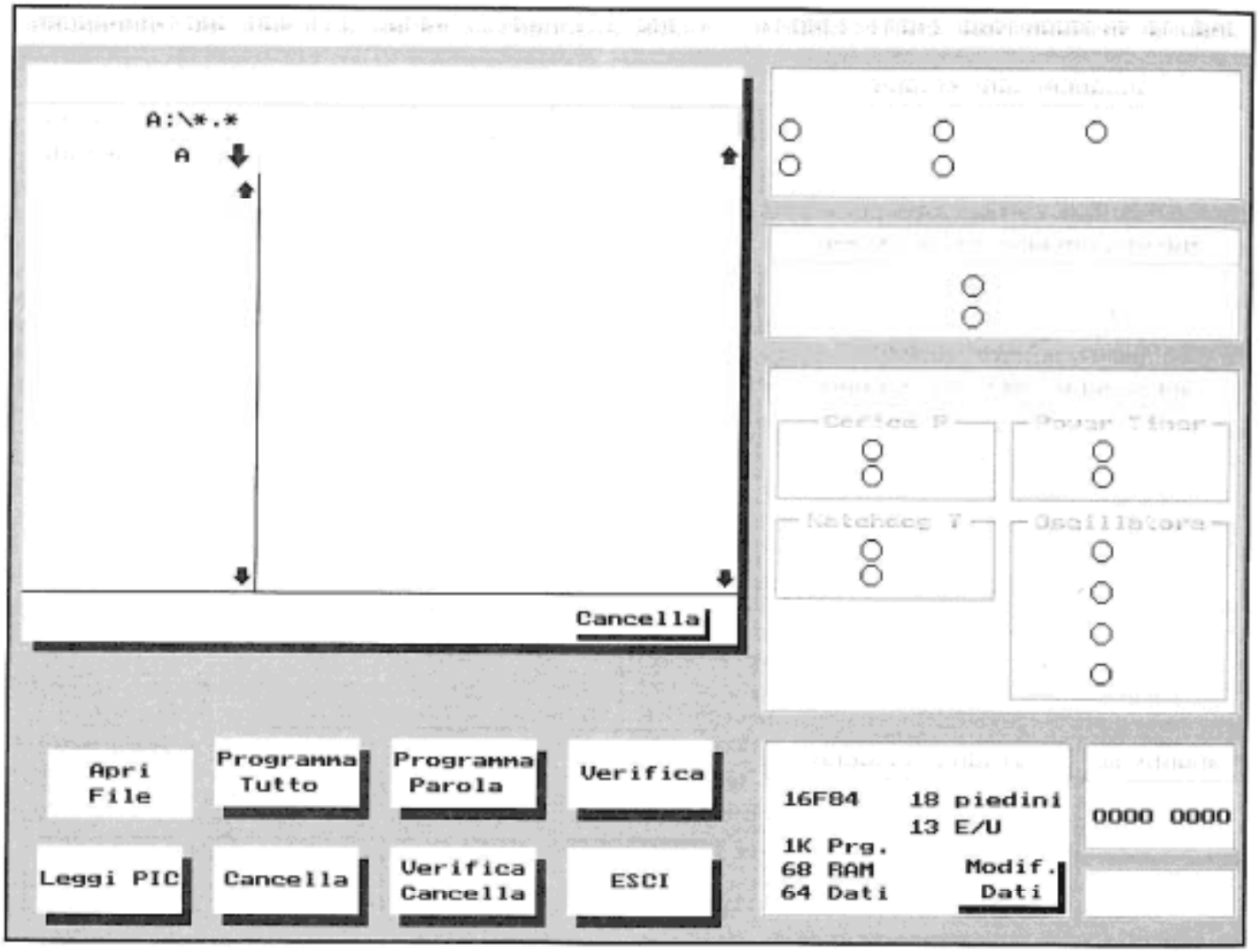


Figura 3-2. Finestra di scelta dei file.

3.4.5 Parola di Configurazione

È una finestra che permette di selezionare diverse configurazioni di lavoro per il PIC da registrare. La parola di configurazione è una parola interna del PIC a cui non si può accedere attraverso il software di applicazione. È possibile accedere ad essa unicamente nel momento di lettura/registrazione del PIC attraverso il software PICME-TR. Sono disponibili diversi parametri che possono essere attivati o meno a scelta dell'utente.

Codice P: Protezione del codice. Attivando questa opzione, quando si registra il PIC, al suo interno si rompe un fusibile. In questo modo, si evita che esso possa essere letto successivamente e con ciò il codice del programma di applicazione resta protetto.
Se un PIC è protetto contro la lettura e, nel caso si cerchi di leggerlo, le informazioni che compaiono nel buffer di memoria non saranno valide. Si leggeranno solo la parola di configurazione e il numero ID.
Se questa opzione viene disattivata, il codice del programma di applicazione può essere letto. Non è protetto.

Watchdog T: Temporizzatore Watchdog. Il watchdog è un temporizzatore il cui oscillatore di lavoro è indipendente dall'oscillatore principale del sistema. Il suo scopo è provocare un RESET e reinizializzare il microcontroller, nel caso il software di applicazione non provveda ad aggiornarlo periodicamente ogni dato intervallo. Ciò può succedere quando, per esempio, il microcontroller ha perso il controllo del software per cause interne, o per cause esterne, ad esempio per un'interferenza elettrica.
Se questa opzione è attiva, il watchdog funzionerà in modo permanente e il software di controllo dovrà incaricarsi di aggiornarlo per evitare il RESET.
Se questa opzione viene disattivata, il watchdog resta sconnesso. Il software di applicazione non avrà la

necessità di aggiornarlo periodicamente, ma in caso di anomalia non ci sarà nulla che reinizi il microcontroller che, pertanto, resterà bloccato.

Brown-OUT: Questa caratteristica è disponibile solo nei modelli del gruppo 16C62X e l'opzione è visibile solo se viene selezionato uno di questi modelli. Selezionando un altro qualsiasi modello, scompare.

Il brown-OUT un circuito in grado di generare un RESET ogni volta che viene individuata una mancanza di alimentazione. Individuare questa situazione può servire per fare in modo che il PIC salvi i dati più importanti, o effettui operazioni immediate, prima della sconnessione totale del circuito. Attivando questa opzione si attiva il circuito di ricerca.

Power Timer: Si tratta di un temporizzatore che, se attivato, ritarda il processo di reinizializzazione del microcontroller di circa 70 mS dopo il RESET, allo scopo di dare tempo alla stabilizzazione sia della tensione di alimentazione che dell'oscillatore principale del sistema.
Attivando questa opzione, il temporizzatore si attiva, rendendo più affidabili le reinizializzazioni.

Oscillatore: Permette di scegliere tra i 4 tipi di oscillatore diversi con cui possono lavorare i PICs, secondo la versione del modello disponibile.

- RC Economico. L'oscillatore interno è stabilizzato esternamente mediante una rete RC. Non ha grande stabilità alla frequenza e non è consigliabile per frequenze superiori ai 4MHz.
- HS Alta velocità. L'oscillatore interno è stabilizzato con un cristallo al quarzo. La velocità massima è di 20MHz, ma si alza anche il consumo.
- XT Velocità standard. L'oscillatore è stabilizzato con un cristallo al quarzo di circa 4MHz.
- LP Basso consumo. La velocità massima è di 200KHz.

ID: Corrisponde al numero di identificazione. Può essere modificato e consente sino a 4 cifre esadecimali che possono servire per identificare il chip o l'applicazione che contiene, il numero di serie, ecc. Generalmente, viene indicato il numero che risulta dal checksum del buffer di memoria.

3.4.6 Pulsanti di Comando

È un insieme di 8 pulsanti che consentono di realizzare le operazioni pratiche.

Apri File: Permette di leggere un file in formato INTEL-HEX e di mostrare il suo contenuto nel buffer di memoria attivo (quello di programma o quello EEPROM dei dati). Mostra una finestra dove si può selezionare l'unità del disco, le subdirectory e i file esistenti. Si sceglie quello che si vuole.

Nella figura 3-2 è mostrato l'aspetto della finestra di scelta dei file.

Leggi PIC: Legge il contenuto della memoria di programma o quello della EEPROM dei dati, nel caso venissero modificati (modello 16C84) e lo memorizza nel buffer di memoria attivo per la sua visualizzazione e/o modifica.

Programma: Registra il contenuto del buffer di memoria attivo (memoria di programma o memoria EEPROM dei dati) nella memoria fisica del microcontroller. Viene anche registrata la parola di configurazione corrente.

Se la verifica della cancellazione è attiva, viene verificato che il PIC sia cancellato completamente. Nel caso non sia così, comparirà un messaggio che indica la situazione e che chiederà una conferma per continuare.

Cancella: Cancella il contenuto della memoria di programma del PIC o della memoria EEPROM dei dati, in funzione di quale delle due sia attiva.
Questo pulsante compare solo se si seleziona il modello PIC 16C84. Questo tipo di microcontroller è dotato di memoria EEPROM sia per il programma che per i dati. Questo tipo di memoria è l'unica che può essere cancellata elettricamente sotto il controllo del software.
Gli altri modelli di PIC, nel caso siano dotati di memoria EPROM con finestra, possono essere cancellati esponendoli a una fonte di luce ultravioletta.

Programma Parola: Programma solo la parola di configurazione selezionata attualmente.
Non viene registrata né la memoria di programma né la memoria EEPROM dei dati.

Verifica Cancellazione: Realizza un controllo, verificando che il PIC sia totalmente cancellato.
Di solito, si consiglia di effettuare questa operazione prima di iniziare il processo di registrazione di qualsiasi PIC, allo scopo di sapere se è usato o meno.

Verifica: Legge il contenuto della memoria di programma o della EEPROM dei dati del PIC e lo compara con il contenuto del buffer di memoria attivo. Se esiste qualche differenza comparirà un messaggio di errore.

Esci: Chiude l'esecuzione del programma PICME-TR e riporta il controllo al sistema operativo MS-DOS.

3.5 MESSAGGI DEL SOFTWARE PICME-TR

Di seguito sono mostrate le serie di messaggi e di messaggi di errore che possono apparire durante una sessione di lavoro con il programma PICME-TR e con il µPIC Trainer.

3.5.1 Errore di Hardware. Controlla le Connessioni con il µPIC Trainer

Causa: Il programma PICME TR verifica costantemente la connessione tra il canale parallelo del personal computer e il sistema µPIC Trainer, così come l'hardware di quest'ultimo.
Controllare la connessione e anche il corretto montaggio del µPIC Trainer.

3.5.2 Errore Unità non Pronta

Causa: Compare quando si effettua qualsiasi operazione con i file e si deve accedere a una unità inesistente, chiusa male o senza dischetto.

3.5.3 Errore di Programmazione nell'Indirizzo XXXX

Causa: Compare quando si sta registrando un PIC e in un dato momento un'informazione non può essere registrata nel suo corrispondente indirizzo, dopo aver seguito i passi necessari.

Assicurarsi che le periferiche del µPIC Trainer connesse alla porta B siano disabilitate e che il PIC sia totalmente cancellato. Riprovare. Nel caso l'errore continui, sostituire il PIC difettoso.

3.5.4 Errore di Verifica nella nell'Indirizzo XXXX

Causa: In modalità verifica, il contenuto del buffer di memoria non coincide con il contenuto della memoria del PIC.

Accertarsi che le periferiche del µPIC Trainer connesse alla porta B siano disabilitate.

3.5.5 Questo PIC non è Cancellato

Causa: Questo messaggio compare quando si registra un PIC, l'opzione di verifica di cancellazione è attiva e, chiaramente, il PIC contiene informazioni nella sua memoria di programma e/o dei dati.
Selezionare CONTINUA se si vuole continuare in ogni caso con la registrazione, oppure CANCELLA in caso contrario.

3.5.6 Questo PIC non è Cancellato

Causa: Questo messaggio compare quando si verifica se la memoria di programma e/o dei dati di un PIC sia cancellata. In questo caso, si consiglia di cancellarlo prima di procedere alla sua registrazione.

3.5.7 Questo PIC è Cancellato

Causa: Il messaggio compare quando si effettua una verifica di cancellazione. Indica che la memoria del PIC non contiene né dati né istruzioni. Il PIC può pertanto essere utilizzato per una registrazione.

3.5.8 Sei Sicuro?

Causa: È un messaggio di conferma che può comparire in diverse occasioni, ad esempio quando si cancella la memoria di un PIC, si esce dal MS DOS, ecc.

Premere OK per accettare o CANCELLA in caso contrario.

3.5.9 Verificare Parola di Configurazione e ID

Causa: Questo messaggio compare quando si vuole registrare solo la parola di configurazione, per verificare che il suo stato corrente sia quello voluto.

CAPITOLO 4: Il modulo LCD

4.1 INTRODUZIONE

Questo capitolo è dedicato a una breve descrizione del modulo LCD compreso nel sistema di valutazione µPIC Trainer.

Si tratta di un modulo in grado di rappresentare 2 linee di 16 caratteri ognuna. Mediante 8 linee di dati è possibile inviargli il carattere ASCII che si vuole visualizzare, così come dati codici di controllo che consentono di realizzare diversi effetti di visualizzazione. Mediante queste linee di dati, il modulo può anche comunicare le informazioni sul suo stato interno.

Con altri 3 segnali addizionali è possibile controllare il flusso di informazioni tra il modulo LCD e il sistema informatico che lo gestisce.

Di seguito è presentata la descrizione dei segnali utilizzati dal modulo LCD con il numero di piedini a cui corrispondono.

PIEDINO N°	SIMBOLO	DESCRIZIONE
1	Vss	Piedino di terra dell'alimentazione
2	Vdd	Piedino di alimentazione da +5V
3	Vo	Piedino di contrasto del cristallo liquido. Generalmente è connesso a un potenziometro attraverso cui viene fornita una tensione variabile tra 0 e +5V che consente di regolare il contrasto del cristallo liquido
4	RS	Scelta del registro di controllo/registro dei dati: RS=0 Scelta del registro di controllo RS=1 Scelta del registro dei dati
5	R/W	Segnale di lettura/scrittura: R/W=0 Il modulo LCD è in scrittura R/W=1 Il modulo LCD è in lettura
6	E	Segnale di attivazione del modulo LCD: E=0 Modulo sconnesso E=1 Modulo connesso
7-14	D0-D7	Bus dei dati bi-direzionali. Attraverso queste linee si effettua il trasferimento di informazioni tra il modulo LCD e il sistema informatico che lo gestisce

4.2 INTERFACCIA CON IL µPIC Trainer

L'interfaccia tra il modulo LCD e il sistema µPIC Trainer si effettua, così come è stato spiegato nello schema della figura 1-1, nel modo seguente:

RB0-RB7 — Sono connesse alle linee dei dati D0-D7 del modulo. Pertanto, attraverso la porta B vengono inviati i codici ASCII o di controllo al modulo, o vengono ricevute dal modulo stesso informazioni sul suo stato interno.

La porta B dovrà essere programmata come uscita quando devono essere inviati codici ASCII o di controllo, o come entrata quando si vuole conoscere lo stato interno del modulo.

RA0 — Si connette con il segnale di controllo R/S. Se si sceglie un livello logico "0" per questa linea, si seleziona il registro di controllo del modulo. Se invece si sceglie un livello logico "1", si seleziona il registro dei dati. Questa linea deve essere programmata come uscita.

RA1 — Si connette con il segnale R/W. Se si sceglie un livello logico "0" per questa linea, il modulo verrà scritto con l'informazione presente in quel momento nella porta B, che dovrà agire come uscita. Scegliendo invece un livello "1", si leggerà lo stato interno del modulo LCD. Tale stato viene ricevuto attraverso la porta B, che dovrà essere programmata come entrata. La linea RA1 deve essere programmata come uscita.

RA2 — Si connette con il segnale E. Se si sceglie un livello logico "1" per questa linea, il modulo resta abilitato ed è possibile quindi il trasferimento di informazioni tra la porta B e le linee dei dati D0-D7. Se invece si sceglie un livello logico "0", il modulo resta sconnesso e le sue linee dei dati D0-D7 in alta impedenza. Anche RA2 deve essere programmata come uscita.

ATTENZIONE: Quando funziona il modulo LCD, gli interruttori RA0, RA1 e RA2 devono essere posti a livello logico "1".

4.3 FUNZIONE DELLE ISTRUZIONI

Sono presentate di seguito. Grazie a loro possono essere configurate diverse opzioni di lavoro del modulo LCD e si possono ottenere diversi effetti di visualizzazione. Consistono in codici differenti che vengono introdotti attraverso il bus dei dati del modulo LCD connesso alla porta B del µPIC Trainer.

4.3.1 Clear Display

Cancella il modulo LCD e colloca il cursore nella prima posizione (indirizzo 0). Per default, pone il bit I/D a "1".

Codice:

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	0	1

Tempo di esecuzione: 1.64 µS

4.3.2 Home

Colloca il cursore nella posizione iniziale (indirizzo 0) e fa in modo che il display inizi a spostarsi dalla posizione originale. Il contenuto della memoria RAM dei dati di visualizzazione (DD RAM) rimane invariabile. L'indirizzo della memoria RAM dei dati per la visualizzazione (DD RAM) è posto a 0.

Codice:

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	1	X

Tempo di esecuzione: 1.64 µS

4.3.3 Entry Mode Set

Stabilisce la direzione di movimento del cursore e specifica se la visualizzazione si sposta verso la posizione successiva dello schermo o meno. Queste operazioni si realizzano durante la lettura o la scrittura della DD RAM o CG RAM. Per visualizzare normalmente porre il bit S a "0".

Codice:

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	1	I/D	S

Tempo di esecuzione: 40 µS

3.3.4 Display On/Off Control

Attiva o disattiva ponendo su On/Off sia il display (D) che il cursore (C) e stabilisce se quest'ultimo debba o meno lampeggiare (D).

Codice:

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	1	D	C	B

Tempo di esecuzione: 40 µS

4.3.5 Cursor or Display Shift

Muove il cursore e sposta il display senza cambiare il contenuto della memoria dei dati di visualizzazione DD RAM.

Codice:

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	1	S/C	R/L	X	X

Tempo di esecuzione: 40 µS

4.3.6 Function Set

Definisce la dimensione dell'interfaccia con il bus dei dati (DL), numero di linee del display (N) e tipo di carattere (F).

Codice:

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	DL	N	F	X	X

Tempo di esecuzione: 40 µS

4.3.7 Set the CG RAM Address

Il modulo LCD, oltre a poter usare tutto il set di caratteri ASCII, consente all'utente di definire 4 o 8 caratteri grafici. La composizione di questi caratteri verrà memorizzata in una memoria chiamata CG RAM, con capacità di 64 bytes. Ogni carattere grafico definito dall'utente è composto da 16 o 6 bytes che vengono memorizzati in posizioni successive della CG RAM.

Mediante questa istruzione viene definito l'indirizzo della memoria CG RAM a partire da cui verranno memorizzati i bytes che definiscono un carattere grafico. Con questo comando, tutti i dati che vengono scritti o letti successivamente, proverranno da questa memoria CG RAM.

Codice:

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	1	Indirizzo della DD RAM					

Tempo di esecuzione: 40 µS

4.3.8 Set the DD RAM Address

I caratteri o dati che verranno visualizzati, saranno prima memorizzati in una memoria chiamata DD RAM, da cui poi passeranno allo schermo.

Mediante questa istruzione viene definito l'indirizzo della memoria DD RAM a partire da cui verranno memorizzati i dati da visualizzare. Concomando, tutti i dati che vengono scritti o letti successivamente, proverranno da questa memoria DD RAM. Gli indirizzi da 80h a 8hF corrispondono ai 16 caratteri della prima riga, mentre da C0h a CFh ai 16 caratteri della seconda riga, per questo specifico modello.

Codice:

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0		Indirizzo della CG RAM					

Tempo di esecuzione: 40 µS

4.3.9 Read Busy Flag & Address

Quando il modulo LCD sta eseguendo una qualsiasi di queste istruzioni, si avrà un certo ritardo nel tempo di esecuzione durante il quale non deve essere inviata alcuna altra istruzione. Per questo motivo, è dotato di un flag chiamato BUSY (BF) che indica che si sta eseguendo una precedente istruzione.

Questa istruzione di lettura informa lo stato di tale flag, oltre a fornire il valore del contatore degli indirizzi della CG RAM o della DD RAM, a seconda di quale sia l'ultima utilizzata.

Codice:

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	1	BF	Indirizzo della CG RAM o della DD RAM						

Tempo di esecuzione: 40 µS

4.3.10 Write Data To CG or DD RAM

Mediante questo comando vengono scritti nella memoria DD RAM i dati che si vogliono mostrare sullo schermo e che saranno i diversi codici ASCII dei caratteri da visualizzare.

Allo stesso modo, si scrivono nella memoria CG RAM i diversi bytes che permettono di creare i caratteri grafici scelti dall'utente.

Scrivere un tipo o l'altro di memoria dipende dal fatto che in precedenza siano state utilizzate le istruzioni di indirizzamento DD RAM oppure quelle di indirizzamento CG RAM.

Codice:

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	1	Codice ASCII o byte del carattere grafico							

Tempo di esecuzione: 40 µS

4.3.11 Read Data From CG or DD RAM

Mediante questo comando vengono letti dalla memoria DD RAM i dati che ha memorizzato e che saranno i codici ASCII dei caratteri visualizzati.

Allo stesso modo, si leggono dalla memoria CG RAM i diversi bytes con cui è stato creato un determinato carattere grafico.

Leggere un tipo o l'altro di memoria dipende dal fatto che si siano utilizzate in precedenza le istruzioni di indirizzamento DD RAM oppure quelle di indirizzamento CG RAM.

Codice:

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	1	Codice ASCII o byte del carattere grafico							

Tempo di esecuzione: 40 µS

4.4 ABBREVIAZIONI

Di seguito sono elencate le abbreviazioni utilizzate nei codici precedenti e il loro relativo significato.

S	=1	Sposta la visualizzazione ogni volta che si scrive un dato
S	=0	Modalità normale
I/D	=1	Aumento del cursore
I/D	=0	Diminuzione del cursore
S/C	=1	Muove il display
S/C	=0	Muove il cursore
R/L	=1	Spostamento a destra
R/L	=0	Spostamento a sinistra
BF	=1	Modulo occupato
BF	=0	Modulo disponibile
DL	=1	Bus dei dati a 8 bits
DL	=0	Bus dei dati a 4 bits
N	=1	LCD a due linee
N	=0	LCD a una linea
F	=1	Carattere da 5 x 10 punti
F	=0	Carattere da 5 x 7 punti
B	=1	Lampeggio del cursore On
C	=1	Cursore On
D	=1	Display On
X	= Indeterminato	

4.5 FUNZIONE DEI CARATTERI

Sono quelle mostrate nella figura 4-1. Le posizioni segnate come CG RAM (n) corrispondono a uno degli 8 possibili caratteri grafici definiti dall'utente.

Figura 4-1. Funzione dei caratteri ASCII del modulo LCD.

4.6 CARATTERI GRAFICI

L'utente può definire sino a 8 caratteri da 5 x 7 punti o 4 da 5 x 10. Si possono selezionare e visualizzare assegnando alla DD RAM qualsiasi valore tra 00 e 07 o tra 08 e 0Fh, come se si trattasse di un codice ASCII.

Si definiscono inserendo alcuni bytes in successivi indirizzi della CG RAM, i cui numeri binari definiscono il carattere così come viene mostrato nella figura 4-2.

Un carattere da 5 x 7 ha bisogno di 8 bytes nella CG RAM per essere definito, mentre uno da 5 x 10 ne ha bisogno di 16. La CG RAM è una memoria a 64 posizioni totali.

Nell'esempio della figura 4-2, per definire la R in 5 x 7 vengono introdotti 8 bytes nelle prime 8 posizioni (dalla 0 alla 7) della CG RAM. Ogni bit di ognuno di questi bytes che abbia livello "1" implica che il suo corrispondente pixel nel LCD è attivo.

Dato che il primo insieme di 8 bytes è come se fosse il primo carattere della CG RAM, esso viene selezionato inserendo il codice 00 nella DD RAM, come se fosse un qualsiasi altro codice ASCII.

Figura 4-2. Generazione dei caratteri grafici

4.7 SEQUENZA DI INIZIALIZZAZIONE

Il modulo LCD esegue automaticamente una sequenza di inizio interna nel momento in cui gli viene fornita la tensione di alimentazione, a condizione che siano soddisfatti i requisiti di alimentazione indicati nella figura 4-3.

Tali requisiti sono: il tempo di ritardo perché la tensione si stabilizzi da 0,2V sino ai 4,5V minimi necessari deve essere tra 0,1mS e 10mS. Inoltre, il tempo di sconnessione deve essere come minimo di 1mS prima di procedere a una nuova connessione.

La sequenza di inizio eseguita è la seguente:

1.- CLEAR DISPLAY
Il flag Busy si mantiene a "1" (occupato) per 15mS, sino a quando termina l'inizializzazione.

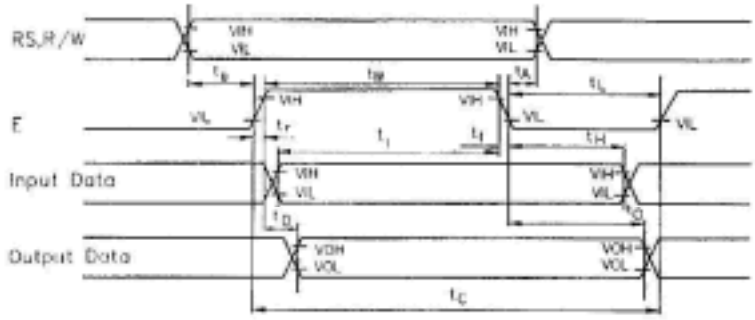
2.- FUNCTION SET
Per default viene scelta la dimensione del bus dei dati a 8 bits (DL=1) e il numero di righe del display a 1 (N=0).

3.- DISPLAY ON/OFF CONTROL
Per default viene scelto il display su Off (D=0), il cursore su Off (C=0) e il lampeggio del cursore su Off (B=0).

4.- ENTRY MODE SET
Per default viene scelto aumento del cursore (I/D=1) e modalità normale senza spostamento del display (S=0).

5.- Viene selezionata la prima posizione della DD RAM.

Se le condizioni di alimentazione non sono soddisfatte, la sequenza di inizializzazione dovrà essere realizzata dal software, dove le istruzioni scelte dall'utente potranno essere quelle indicate in precedenza o qualsiasi altra definita in base alle sue necessità. È importante che la prima istruzione inviata faccia una pausa di circa 15mS per consentire la completa reinizializzazione interna del modulo LCD.



PARAMETRI \ CHIP DI CONTROLLO		SAMSUNG KS0066	HITACHI HD44780	SANYO LC7985NA	EPSON SED1278	OKI MSM6222	RECOMMENDED TIMING	UNIT
Enable Cycle Time	tC (min)	1000	1000	1000	500	667	1000	nS
Enable Pulse Width High Level Low Level	tW (min) tL (min)	450 450	450 450	450 450	220 220	280 280	450 450	nS nS
E Raise Time	tr (max)	25	25	25	25	25	25	nS
E Fall Time	tf (max)	25	25	25	25	25	25	nS
Set-up Time	tB (min)	140	140	140	40	140	140	nS
Data Set-up Time	tI (min)	195	195	195	60	180	195	nS
Data Delay Time	tD (max)	320	320	320	120	220	320	nS
Address Hold Time	tA (max)	10	10	10	10	10	10	nS
Hold Time Input Data Output Data	tH (min) tO (min)	10 20	10 20	10 20	10 20	10 20	10 20	nS nS

Figura 4-4. Diagramma dei tempi.

4.8 DIAGRAMMA DEI TEMPI

È quello mostrato nella figura 4-4. Assieme alla tavola che lo accompagna si possono conoscere i tempi dei segnali per diversi chips di controllo e moduli LCD.

4.9 ROUTINES DI CONTROLLO

In questo paragrafo viene presentata una serie di routines scritte nell'assembler MPASM della MICROCHIP, con l'obiettivo di fornire all'utente un'idea di come gestire le diverse operazioni da effettuare con il modulo LCD. Sono realizzate basandosi su un PIC 16C84 a 4MHz di velocità.

4.9.1 LCD_E

Genera un impulso per il piedino RA2 (segnale E) per attivare il modulo LCD. L'impulso è di 1mS di durata a 4MHz. Nel caso di frequenze di lavoro maggiori, bisognerà aumentare questo tempo per non superare la durata minima imposta dal produttore del modulo LCD.

```
LCD_E   bsf     RA,2    ;Attiva segnale E
        nop             ;Pausa di 1mS
        bcf     RA,2    ;Disattiva segnale E
        return
```

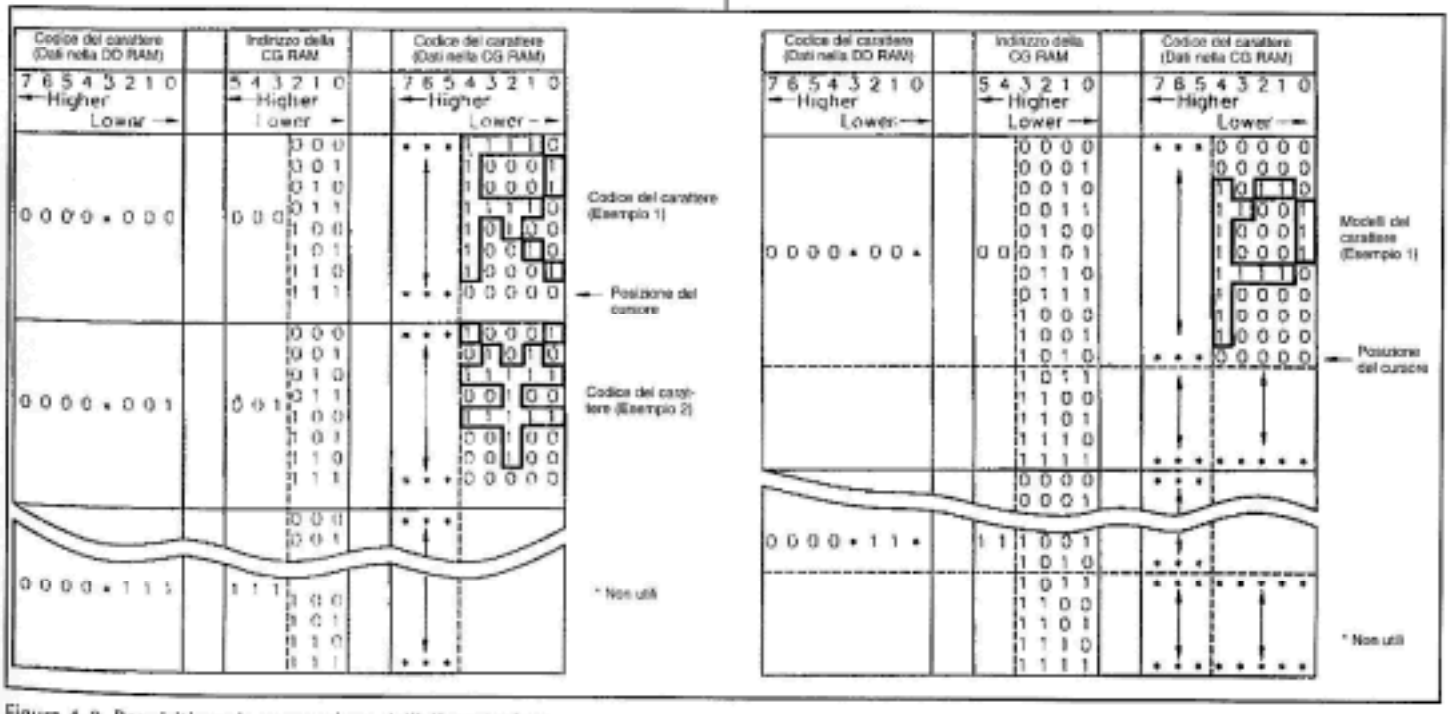


Figura 4-3. Requisiti per la connessione dell'alimentazione.

MODULO LCD

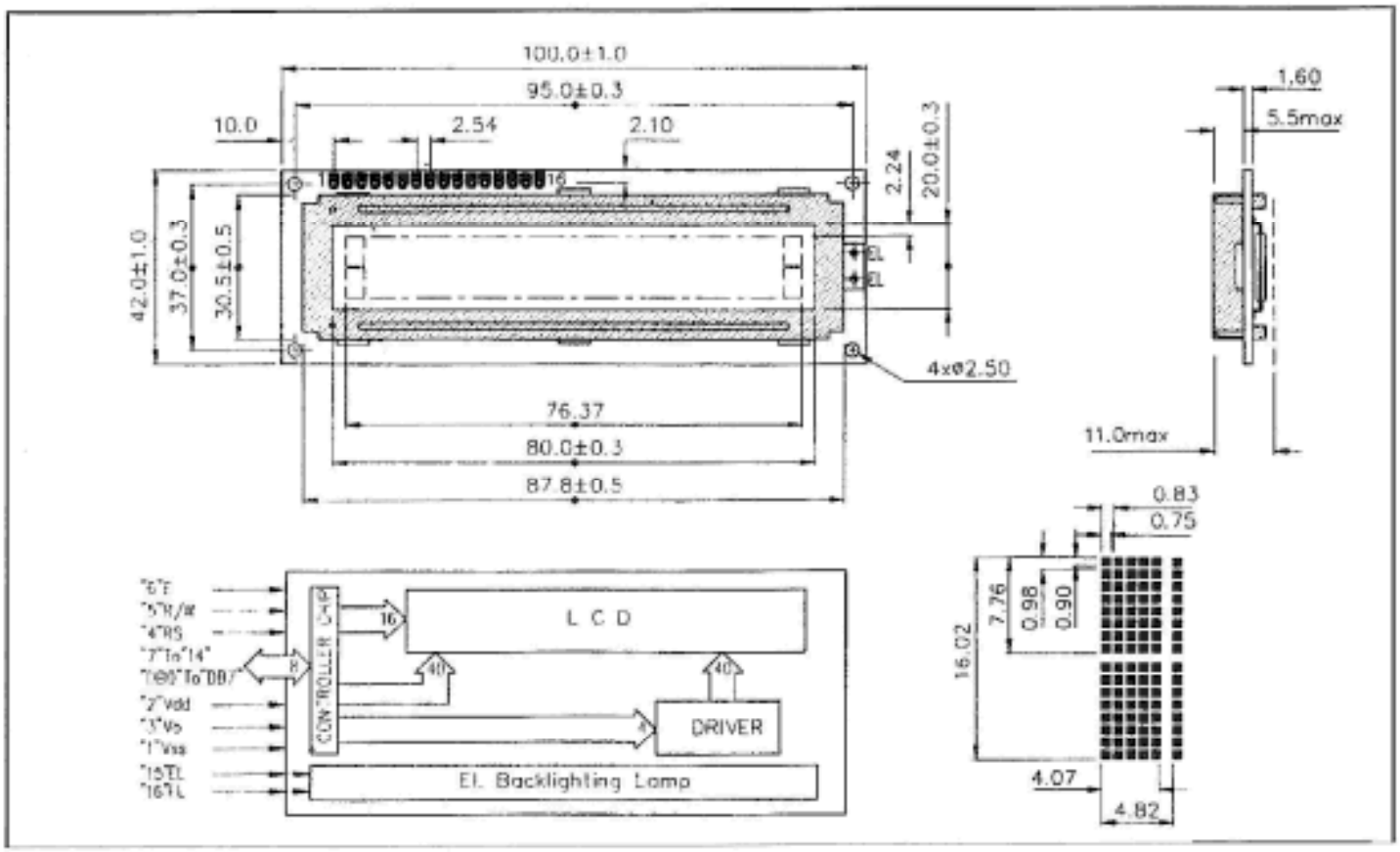


Figura 4-5. Dimensioni del modulo LCD

4.9.2 LCD_BUSY

Verifica lo stato del flag Busy del modulo LCD e prima di tornare indietro aspetta che termini qualsiasi istruzione precedente.

```
LCD_BUSY  bsf     RA,1      ;Mette il LCD in modalità lettura
          bsf     STATUS,5  ;Seleziona il banco 1
          movlw   0xff
          movwf   TRISB     ;Porta B funziona da entrata
          bcf     STATUS,5  ;Seleziona il banco 0
          bsf     RA,2      ;Attiva il LCD (segnale E)
          nop

L_BUSY    btfsc   RB,7      ;Verifica il bit busy
          goto    L_BUSY    ;è a "1" (occupato)
          bcf     RA,2      ;Disattiva il LCD (segnale E)
          bsf     STATUS,5  ;Seleziona il banco 1
          clrf    TRISB     ;Porta B funziona come uscita
          bcf     STATUS,5  ;Seleziona il banco 0
          bcf     RA,1      ;Mette il LCD in modalità scrittura
          return
```

4.9.3 LCD_REG

Deposita il codice di istruzioni presente nel registro W del PIC sulla porta B. Attende che il modulo LCD esegua l'ultima operazione e genera l'impulso di attivazione nel segnale E.

```
LCD_REG   bcf     RA,0      ;Disattiva RS (modalità istruzione)
          movwf   RB        ;Mostra il codice di istruzione
          call    LCD_BUSY  ;Attende che si liberi il LCD
          goto    LCD_E     ;Genera impulso nel segnale E
```

4.9.4 LCD_DATI

Deposita il codice ASCII del carattere da visualizzare presente nel registro W sulla porta B. Attende che il modulo LCD esegua l'ultima operazione e genera l'impulso di attivazione nel segnale E.

```
LCD_DATI  bcf     RA,0      ;Disattiva RS (modalità istruzione)
          movwf   RB        ;Valore ASCII da mostrare per RB
          call    LCD_BUSY  ;Attende che si liberi il LCD
          bsf     RA,0      ;Attiva RS (modalità dato)
          goto    LCD_E     ;Genera impulso nel segnale E
```

4.9.5 LCD_INI

Effettua l'inizializzazione del modulo LCD in base ai tempi indicati dal produttore (15mS). In questo esempio, parte con un'interfaccia da 8 bits del bus dei dati, 2 linee di visualizzazione e caratteri da 5 x 7 punti.

```
LCD_INI   movlw   b'00111000'
          call    LCD_REG     ;Codice di istruzione
          call    DELAY_5MS   ;Temporizza 5mS
          movlw   b'00111000'
          call    LCD_REG     ;Codice di istruzione
          call    DELAY_5MS   ;Temporizza 5mS
          movlw   b'00111000'
          call    LCD_REG     ;Codice di istruzione
          call    DELAY_5MS   ;Temporizza 5mS
          return
```

4.9.6 DELAY_5MS

Genera una temporizzazione di 5mS. Si utilizzano 2 variabili chiamate DATO_A e DATO_B che decrescono sino a completare la temporizzazione.

```
DELAY_5MS movlw   0x1A
          movwf   DATO_B    ;Carica la variabile DATO_B
          clrf    DATO_A    ;Carica la variabile DATO_A
DELAY_1   decfsz  DATO_A,1  ;Diminuisce la variabile DATO_A
          goto    DELAY_1
          decfsz  DATO_B,1  ; Diminuisce la variabile DATO_B
          goto    DELAY_1
          return
```

4.9.7 DELAY10

Genera, con l'aiuto del TMR0 e il prescaler da 256, una temporizzazione di circa 10mS che si ripete tante volte quante sono indicate dalla variabile TEMPO1.

```
DELAY10   bcf     INTCON,2  ;Cancella il flag di stato del TMR0
          movlw   0xD8
          movwf   TIMER0    ;Carica TMR0 perché conti 39
DELAY10_1 btfss   INTCON,2  ;Attende overflow del TMR0
          goto    DELAY10_1
          decfsz  TEMPO1,1  ;Ripete TEMPO1 volte
          goto    DELAY10
          return
```

4.9.8 DELAY1S

In base alla routine precedente, viene realizzata una temporizzazione di 1S, caricando nella variabile TEMPO1 il valore 100 (64h).

```
DELAY1S   movlw   0x64
          movwf   TEMPO1    ;Carica la variabile TEMPO1
          call    DELAY1S
          return
```

4.10 Dimensioni del Modulo LCD

Nella figura 4-5 sono mostrate le dimensioni meccaniche del modulo LCD, congiuntamente al suo schema a blocchi.

CAPITOLO 5: Tutorial

5.1 INTRODUZIONE

Nei tutorial che seguiranno verrà descritta la sequenza delle fasi che devono essere seguite dall'idea originale sino alla registrazione del microcontroller. Nella figura 5-1 è mostrato un diagramma a blocchi delle fasi da seguire.

La prima operazione da effettuare consiste nello scrivere il codice principale del programma. In seguito, esso verrà convertito in un codice eseguibile mediante l'utilizzo di un programma assembler. L'assembler qui proposto è quello fornito dalla ARIZONA MICROCHIP INC.. Si chiama MPASM e può essere scaricato direttamente da Internet, al sito http://www.microchip.com

Tutti gli esempi sono stati scritti usando l'editor di testi EDIT del MS-DOS. Si può utilizzare qualsiasi altro editor di testi, ma in questo caso, quando si salva su disco, bisogna salvarlo in formato testo DOS o in formato ASCII.

I controlli e la simulazione verranno realizzati mediante il simulatore software MPSIM della ARIZONA MICROCHIP INC., che si può scaricare dallo stesso sito precedente. La registrazione e l'esecuzione di questo primo esempio saranno effettuate sul sistema di valutazione µPIC Trainer, progettato e distribuito dalla Microsystems Engineering.

Si inizia con una breve descrizione del codice principale del programma ESEMPIO1, la cui esecuzione accende i diodi LEDs connessi alla porta B del µPIC Trainer, mostrando un numero binario.

5.2 SCRITTURA DEL CODICE PRINCIPALE

Come si può osservare, il codice principale è strutturato in colonne. Qualsiasi testo che inizia nella prima colonna, è considerato una etichetta ed è una parte del campo delle etichette.

Le successive tre colonne contengono il campo delle istruzioni, il campo dei dati e il campo dei commenti. I commenti devono iniziare con un punto e virgola (;) e si possono trovare anche nella prima colonna.

(1) Campo delle etichette

Le etichette sono nomi di subroutines o sezioni del codice principale. Dando dei nomi a parti del programma, si rende possibile saltare delle istruzioni, oppure fare riferimento a queste parti senza dover necessariamente ricordare gli indirizzi fisiche dove sono collocate.

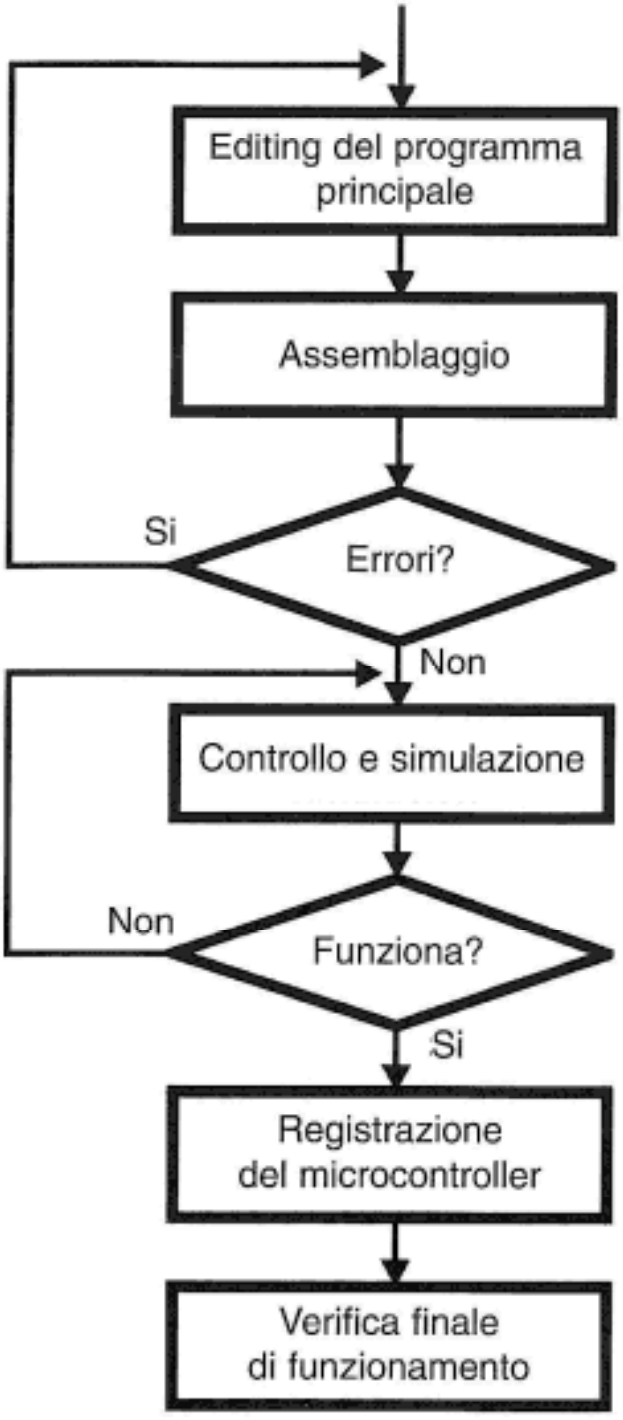


Figura 5-1. Diagramma della sequenza delle fasi da seguire.

Molti assembler stabiliscono un limite ridotto per la dimensione delle etichette e per i caratteri che possono essere usati per definirle. L'assembler MPASM permette etichette sino a 32 caratteri. Nel programma ESEMPIO1.ASM, Inizializza, Principale e Termina sono tre etichette.

Nell'assembler MPASM, un'etichetta può essere seguita da due punti (:), spazio, tabulazione o RETURN. Deve iniziare con un carattere alfanumerico o di sottolineatura (_) e può contenere qualsiasi combinazione di caratteri alfanumerici.

(2) Campo delle istruzioni

La seconda colonna corrisponde al campo delle istruzioni. Può trattarsi di una istruzione del microcontroller o di una istruzione per l'assembler, chiamata direttiva.

(3) Campo dei dati

La terza colonna contiene il campo dei dati. Come indica il nome stesso, contiene dati o operatori per le istruzioni. Nel caso dei microcontrollers PIC, i dati possono essere un registro, un bit di un registro, un'etichetta o un numero costante (chiamato costante). Alcune istruzioni non contengono dati. Se un'istruzione ha bisogno di dati multipli, è necessario separarli con virgole (,).

L'indicazione della base con cui sono espressi i dati o gli operatori è opzionale: d'65', b'01000001', 0x41, o'101' e 'A', sono nella stessa base. Il suffisso d o D si usa per esprimere un valore in base decimale. Il suffisso b o B esprime invece un valore in codice binario. Per esprimere un valore in base esadecimale si usa il suffisso 0x o 0X. Un numero in base otto viene rappresentato mediante il suffisso o o O. Infine, un valore può essere espresso mediante un carattere ASCII se viene chiuso tra virgolette semplici ('A').

(4) Campo dei commenti

L'ultimo campo è quello dei commenti, a condizione che ci sia sempre un punto e virgola (;) come primo carattere. Può essere collocato in qualsiasi punto del codice principale.

I commenti sono una delle operazioni più importanti che il programma deve sviluppare quando scrive il codice. Commenti scritti correttamente descrivono l'operazione realizzata da determinate istruzioni o subroutines. Senza commenti è molto difficile decifrare il codice, anche solo dopo poco tempo che qualcuno lo ha creato.

L'elenco che segue mostra l'aspetto del codice principale del programma ESEMPIO1.ASM, il primo esempio analizzato in questo capitolo.

Campo Etichette	Campo Istruzioni	Campo Dati	Campo Commenti
;Programma ESEMPIO1.ASM			
;Mostra nei LEDs connessi alla porta B di uscita un numero binario			
LIST	P=16C84	;Seleziona tipo di microcontroller	
	LIST	C=132	;File di elenco a 132 caratteri
PORTB	EQU	6	;Indirizzo della porta B
TRISB	EQU	6	;Indirizzo del registro TRISB
STATUS	EQU	3	;Indirizzo del registro di stato
	ORG	0	;Posizione del vettore di RESET
	GOTO	Inizializza	;Inizia il programma dopo il vettore ;di interrupt
	ORG	5	;Una posizione dopo il vettore di ;interrupt
Inizializza	BSF	STATUS,5	;Seleziona pagina 1 dell'area dei dati
	CLRF	TRISB	;Programma la porta B come uscita ; ponendo a 0 il registro TRISB
	BCF	STATUS,5	;Seleziona pagina 0 dell'area dei dati
Principale	MOVLW	b'01010101'	;Inserisce il numero 01010101 nel registro W
	MOVWF	PORTB	;Porta il numero binario ai LEDs connessi ;alla porta B del µPIC Trainer
Termina	SLEEP		;Ferma l'esecuzione del programma
	END		;Fine del codice principale

5.3 CONVENZIONI NELLA SCRITTURA DEL CODICE PRINCIPALE

Per rendere il compito del programmatore più facile si usano alcune convenzioni. Ognuno può scegliere quelle che più gli piacciono e che lo aiutano a essere più produttivo. In generale, le convenzioni sono qualsiasi azione che semplifica la revisione e la comprensione di un programma, in particolare quello che un utente ha scritto e che deve essere rivisto alcuni mesi dopo. Di seguito sono presentate alcune delle convenzioni che saranno usate.

- *I file del codice principale avranno l'estensione *.ASM o *.SRC*
- *I file di elenco avranno l'estensione *.LST*
- *I file di codice oggetto avranno l'estensione *.OBJ*
- *I file esecutivi in formato Intel Hex avranno l'estensione *.HEX*
- *I file di riferimento incrociato avranno l'estensione *.XRF*
- *I file di errore di programmazione avranno l'estensione *.ERR*
- *I file dei simboli e di pulizia per la successiva simulazione avranno l'estensione *.COD*
- *I file coe comandi di inizializzazione per il simulatore avranno l'estensione *.IN*
- *Gli mnemonici scritti in maiuscolo fanno in modo che il codice scritto sia più visibile*
- *I commenti spiegano ogni linea del codice*
- *Un paragrafo di commento spiega le routines o l'insieme di istruzioni, data che i campi di commento sono in genere piccoli*
- *Lo spazio tra caratteri si indica con "_" RB0_ES_1, più semplice da leggere di RB0ES1*

Bisogna ricordare che le convenzioni sono qualsiasi cosa che rende più facile la lettura e la comprensione del proprio codice, ad esempio:

1.- *Una intestazione standardizzata.*
2.- *Mettere le subroutines nello stesso punto, tutte vicine.*
3.- *Creare diagrammi di flusso o scrivere uno pseudocodice.*

5.4 IL PRIMO ESEMPIO

Di seguito vengono esaminate una per una le diverse parti del programma ESEMPIO1.ASM.

5.4.1 Direttive

Normalmente, all'inizio di tutti i programmi principali, nell'intestazione, si inserisce una serie di commenti che indicano il nome del codice stesso e una descrizione sulle operazioni da realizzare. Da qui in poi, queste informazioni saranno sintetizzate con la sigla LIST.

Questa sigla e quelle che saranno indicate di seguito sono delle direttive per l'assembler. Possono variare da un assembler all'altro, in base al produttore, anche se in genere sono abbastanza simili tra loro. A questo punto, si presentano quelle utilizzate nell'esempio e che corrispondono all'assembler MPASM della ARIZONA MICROCHIP INC..

Una direttiva è un comando scritto nel codice principale per realizzare un controllo diretto o per risparmiare tempo quando si programma. Il risultato dell'inserimento delle direttive può essere visto nel file *.LST, dopo aver assemblato il programma.

```
LIST    P=16C84  ;Seleziona tipo di microcontroller
LIST    C=132    ;File di elenco a 132 caratteri
```

La direttiva LIST ha diverse opzioni che permettono di scegliere, tra le altre cose, il tipo di chip da usare (P), il numero di caratteri per linea (C), la dimensione dei tabulatori (B), la base di numerazione di default (R), i livelli dei messaggi di uscita (W), ecc. Si ribadisce ancora che ogni assembler ha le proprie direttive, che dovranno essere sufficientemente documentate, anche se tutte perseguono obiettivi similari.

```
PORTB   EQU   6   ;Indirizzo della porta B
TRISB   EQU   6   ;Indirizzo del registro TRISB
STATUS  EQU   3   ;Indirizzo del registro di stato
```

Le successive direttive che si incontreranno nel codice principale dell'esempio sono le EQU. Esse assegnano dei valori alle etichette volute. In questo caso, PORTB ha assegnato il valore 6 che corrisponde all'indirizzo dell'area dei dati della porta B (banco 0), STATUS all'indirizzo 3 e TRISB il valore 6, che corrisponde all'indirizzo dell'area dei dati del registro TRISB (banco 1).

```
ORG     0       ;Posizione del vettore di RESET
```

La direttiva ORG indica all'assembler dove deve iniziare a inserire le istruzioni nella memoria del programma, vale a dire l'ORiGine per tutto il codice che segue. L'indirizzo di inizio (origine) è nella posizione 0, in quanto la famiglia di microcontrollers PIC di gamma media, dopo l'accensione o il RESET, esegue sempre l'istruzione che si trova all'indirizzo 0. Si chiama Vettore di Reset.

L'indirizzo 4 è il Vettore di Interrupt. Se si crea un'interrupt, il microcontroller esegue l'istruzione che si trova in questo punto. È una buona scelta lasciare libero l'indirizzo 4, per poterlo usare se più avanti si vuole aggiungere capacità di interrupt al programma. Il programma di esempio salta il Vettore di Interrupt e inizia all'indirizzo 5.

```
            ORG     0           ;Posizione del vettore di RESET
            GOTO    Inizializza ;Inizia il programma dopo il vettore
                                ;di interrupt
            ORG     5           ;Una posizione dopo il vettore
                                di interrupt
Inizializza BSF     STATUS,5    ;Seleziona pagina 1 dell'area dei dati
            CLRF    TRISB       ;Programma la porta B come uscita
                                ;ponendo a 0 il registro TRISB
            BCF     STATUS,5    ;Seleziona pagina 0 dell'area dei dati
```

L'istruzione GOTO Inizializza è stata inserita nell'indirizzo 0, che è quello successivo all'indirizzo ORG 0. La prima istruzione eseguita dal microcontroller quando viene acceso o dopo un RESET non è ORG 0, ma GOTO Inizializza, perché ORG 0 è una direttiva dell'assembler e non un'istruzione del microcontroller. Nella figura 5-2 è mostrata la mappa della memoria con le istruzioni così come si trovano al suo interno.

MEMORIA DI PROGRAMMA

INDIRIZZO	ISTRUZIONE	
0000	GOTO 5	(Vettore di RESET)
0001	.....	
0002	.....	
0003	.....	
0004	.....	(Vettore di Interrupt)
0005	BSF STATUS,5	
0006	CLRF TRISB	
0007	BCF STATUS,5	
0008	MOVLW 01010101	
0009	MOVWF PORTB	
000A	SLEEP	
000B	.....	
000C	.....	
000D	.....	
000E	.....	
000F	.....	
.....	.....	
.....	.....	
03FE	.....	
03FF	.....	

Figura 5-2. Mappa della memoria con l'indicazione delle direzioni dove viene caricata ogni istruzione del programma

L'assembler potrebbe inserire l'istruzione che segue GOTO Inizializza all'indirizzo 1, il successivo disponibile. Chiaramente, in questo caso, il microcontroller è saltato a un indirizzo segnato con l'etichetta Inizializza. Dal momento che ORG 5 precede tale etichetta, la posizione di memoria

5 contiene l'istruzione BSF STATUS,5 ed è l'indirizzo dell'istruzione successiva da eseguire.

5.4.2 Codice del Programma

Le tre istruzioni che seguono l'etichetta Inizializza programmano la porta B come uscita.

In primo luogo, l'istruzione BSF STATUS,5 (Bit Set F) pone a 1 il bit 5 del registro di stato (RP0). Ciò fa in modo che la prima pagina della memoria RAM dei dati diventi la pagina attiva (consultare le caratteristiche dei modelli di PIC).

Durante il processo di assemblaggio del codice principale, l'assembler sostituisce le etichette con il loro valore reale. Pertanto, STATUS è sostituito con il valore 3, PORTB con il valore 6, TRISB con il valore 6, Inizializza con l'indirizzo 5, Principale con l'indirizzo 8 e Termina con l'indirizzo 0xA della memoria di programma.

In secondo luogo, l'istruzione CLRF TRISB (Clear File Register, Registro Tristato della Porta B – TRISB) cancella il registro TRISB, configurando le linee di questa porta come uscite. Un bit 0 e un registro di indirizzo dei dati fanno si che la porta si configuri come uscita e un 1 invece come entrata (0 = uscita, 1 = entrata).

Infine, BCF STATUS,5 ripristina la pagina 0 dei dati RAM come pagina attiva, ponendo a 0 il bit 5 del registro di stato. La sequenza delle operazioni spiegate, realizza la commutazione tra la pagina 1 e la pagina 0.

```
Principale  MOVLW   b'01010101'  ;Inserisce il numero 01010101 nel registro W
            MOVWF   PORTB        ;Mostra il numero binario ai LEDs connessi
                                 ;alla porta B del µPIC Trainer
```

Principale è un'etichetta che corrisponde all'indirizzo 8 della memoria di programma. Le istruzioni che seguono questa etichetta collocano una costante nella porta B di uscita e accendono i LEDs del µPIC Trainer con questo numero binario.

MOVLW b'01010101' (Move Literal al registro di lavoro W) muove il numero binario 01010101 al registro di lavoro W. Se si usa un altro sistema di numerazione, si scriverà MOVLW 0x55 in esadecimale, MOVLW d'85' in decimale, o MOVLW 'U' nel codice ASCII.

MOVWF PORTB (Move Working Register to File Register, Port B) copia il valore del registro di lavoro W nel registro della porta B che è connessa ai diodi LEDs di uscita nel sistema µPIC Trainer.

```
Termina  SLEEP      ;Ferma l'esecuzione del programma
```

Quando il programma ha finito, viene indicato al microcontroller di smettere di eseguire le istruzioni. Termina è un'etichetta che corrisponde all'indirizzo 0xA della memoria di programma, dove è inserita l'istruzione SLEEP. Essa non ha bisogno di comandi e mantiene attive tutte le uscite, ma congela l'esecuzione del programma fermando il clock principale del microcontroller. Le esigenze di consumo di energia in modalità SLEEP sono minime e questo stato si chiama di "riposo" o di "basso consumo".

```
            END         ;Fine del codice principale
```

Quest'ultima direttiva indica all'assembler la fine del codice principale, dove deve essere concluso il processo di costruzione.

Come è stato osservato in precedenza, i programmi principale sono stati scritti con l'EDIT del MS-DOS. Una volta conclusa l'operazione, selezionare Esci o Exit del menu File. Se inavvertitamente è stato effettuato un cambiamento nel programma, EDIT chiederà se tale modifica debba essere salvata. Selezionare NO per mantenere intatto il programma originale.

5.4.3 Assemblaggio del Programma

Le istruzioni del codice principale dell'assembler sono comprensibili per il programmatore, mentre il microcontroller capisce solo i numeri binari. La funzione di un programma è di convertire il testo del codice principale nell'equivalente del linguaggio macchina numerico del microcontroller. Per esempio, il numero esadecimale 0x2805 rappresenta l'istruzione GOTO Inizializza.

Il passo successivo nel processo di programmazione è assemblare il file del codice principale e ricavare il file di codice eseguibile. Una volta creato l'eseguibile, si può già copiarlo nel simulatore per il controllo e le verifiche e, se funziona, nella memoria del microcontroller.

Il programma assembler MPASM converte il codice principale in codice eseguibile in due fasi. Nella prima, l'assembler verifica l'esatta sintassi delle istruzioni, i nomi delle etichette duplicati e assegna dei valori ai simboli. Nell'esempio precedente, l'etichetta Inizializza è sostituita dall'indirizzo numerico 5. Nella seconda fase, l'assembler converte tutte le istruzioni nei suoi codici macchina numerici.

Per l'assemblaggio sarà utilizzato, come è già stato detto, l'assembler MPASM della ARIZONA MICROCHIP INC.. Esso dovrà trovarsi in una subdirectory, che verrà chiamata PIC e che conterrà inoltre il codice principale da sviluppare (ESEMPIO1.ASM). Una volta entrati in questa directory si richiama l'assembler digitando:

C:\PIC> MPASM (ENTER)

Comparirà una finestra come quella mostrata nella figura 5-3, in cui dovranno essere riempiti i diversi campi.

Source File: deve essere completato con il nome del codice principale seguito da ENTER.

Processor Type: è opzionale. Si indica il nome del microcontroller per il quale è stato scritto il codice principale. Se non viene indicato, viene mostrato quello che è stato definito mediante la direttiva LIST P= del codice principale.

Error File: Yes. Viene creato per default un file con lo stesso nome di quello principale e con estensione .ERR che contiene un elenco dei possibili errori trovati durante la compilazione. Se non si vuole generare tale file, muovere il cursore sino a questo campo e indicare No premendo ENTER.

Cross Reference File: No. Per default, non viene creato il file di riferimento incrociato. Se si vuole creare questo file, muovere il cursore sino a questo campo e indicare Yes premendo ENTER. Tale file si genera con lo stesso nome di quello principale e l'estensione .XRF e contiene un elenco di tutte le etichette trovate.

Listing File: Yes. Per default viene creato un file con lo stesso nome di quello principale e l'estensione .LST che contiene l'elenco completo della compilazione. Se non si vuole creare questo file, muovere il cursore sino a questo campo e indicare No premendo ENTER.

Hex Dump Type: INHX8M. Questo è il file eseguibile che si ottiene per default. Tale file, con lo stesso nome di quello principale e l'estensione .HEX, è l'eseguibile che servirà per la successiva simulazione e/o registrazione nella memoria del microcontroller. Il formato predefinito è l'INTEL-HEX a 8 bits (INHX8M), che può essere cambiato muovendo il cursore in questo campo e scegliendo i diversi formati che compaiono premendo ENTER.

Assemble to Object File: No. Questa opzione è disattivata per default. Può essere attivata muovendo il cursore in questo campo e indicando Yes premendo ENTER. In questo modo, si genera un file con estensione .OBJ che contiene il modulo obiettivo riposizionabile e che potrà essere utilizzato, mediante un linker, per unirlo ad altri moduli e formare un unico eseguibile.

Dopo aver riempito tutti i campi della finestra, premere F10 per iniziare l'assemblaggio. Al termine, verrà indicato il numero di messaggi (Messages), avvisi (Warnings) e errori (Errors) verificatisi. Questi ultimi, nel caso compaiano, dovranno essere verificati nel file .ERR per apportare le opportune correzioni nel codice principale, ripetendo poi il processo di assemblaggio.

Se tutto funziona correttamente, l'assembler avrà già creato i file .LST e .HEX.

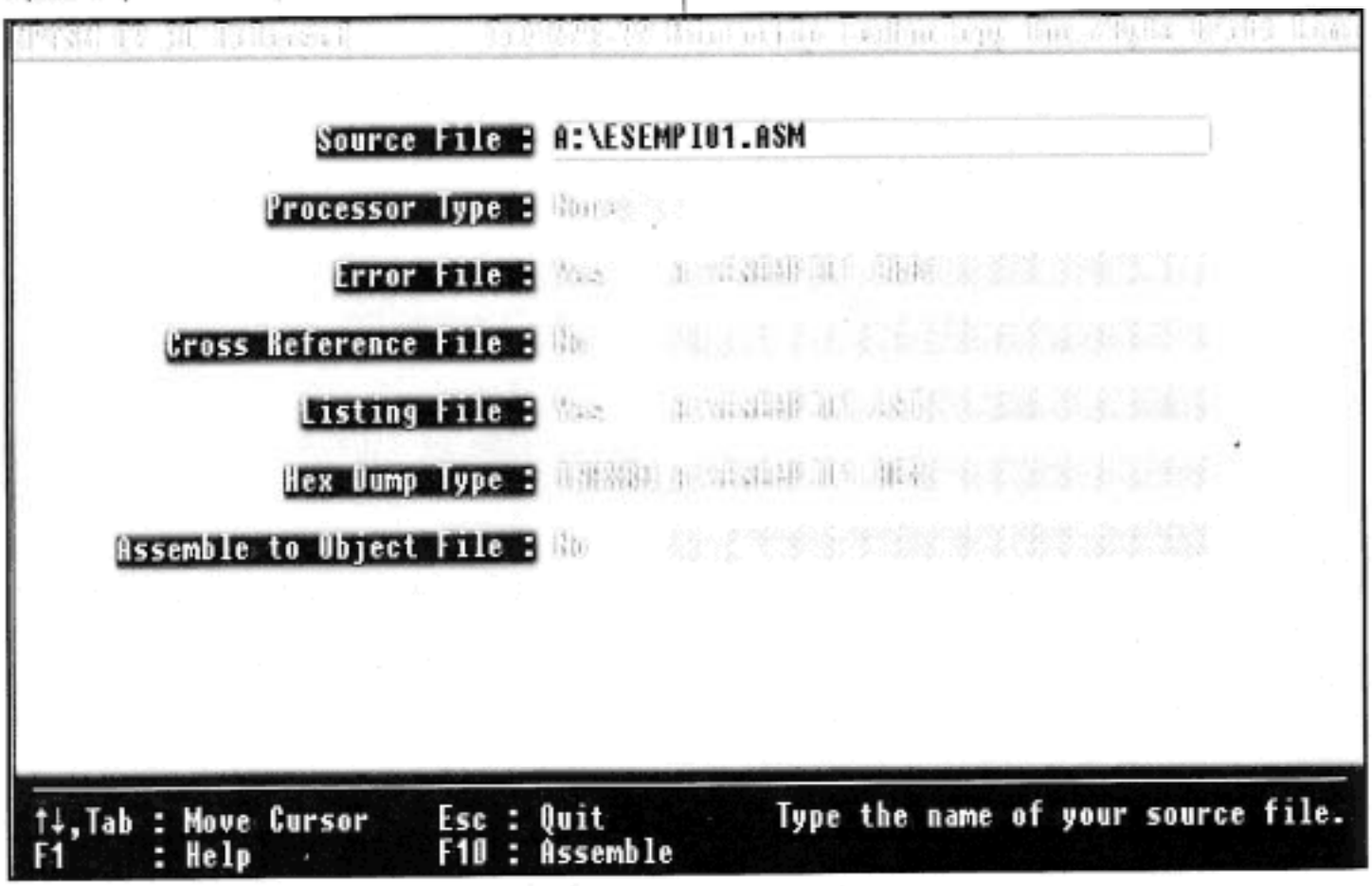


Figura 5-3. Finestra dell'assembler con i diversi campi da riempire.

Si inizia esaminando il file di elenco. Per farlo, dal prompt del DOS bisogna digitare:

C:\PIC> EDIT ESEMPIO1.LST (ENTER)

Innanzitutto, si può vedere come il file di elenco contenga una copia del file del codice principale, ma con tre colonne aggiunte alla sua sinistra.

L'assembler numera ogni linea del codice principale e identifica ogni indirizzo di memoria del programma con il suo contenuto. L'aspetto di tale elenco è quello mostrato nella tabella.

La colonna a sinistra rappresenta le posizioni di memoria del programma dove si trovano le istruzioni. La colonna successiva presenta il codice esadecimale di ogni istruzione. La terza colonna indica il numero di linea del

```
MPASM 01.21 Released ESEMPIO1.ASM 11-20-1996 8:45:06 PAGE 1

LOC OBJECT CODE LINE SOURCE TEXT
VALUE
                     00001 ;Programma ESEMPIO1.ASM
                     00002 ;Evidenzia un numero binario nei LEDs connessi alla porta B di uscita
                     00003
                     00004                LIST      P=16C84         ;Seleziona il tipo di microcontroller
                     00005                LIST      C=132           ;File di elenco a 132 caratteri
                     00006
0006                 00007 PORTB          EQU       6               ;Indirizzo della porta B
0006                 00008 TRISB          EQU       6               ;Indirizzo del registro TRISB
0003                 00009 STATUS         EQU       3               ;Indirizzo del registro di stato
                     00010
0000                 00011                ORG       0               ;Posizione del vettore di RESET
0000 2805 00012                           GOTO      Inizializza     ;Inizia il programma dopo il vettore
                     00013                                           ;di interrupt
0005                 00014                ORG       5               ;Una posizione dopo il vettore
                     00015                                           ;di interrupt
                     00016
0005 1683            00017 Inizializza                BSF       STATUS,5  ;Seleziona pagina 1 dell'area dei
                                                                    ;dati
0006 0186            00018                CLRF      TRISB           ;Programma la porta B come uscita ponendo
                     00019                                           ;a 0 il registro TRISB
0007 1283            00020                BCF       STATUS,5        ;Seleziona pagina 0 dell'area dei
                                                                    ;dati
                     00021
0008 3055            00022 Principale     MOVLW     b'01010101'     ;Mette il numero 01010101 nel
                                                                    ;registro W
0009 0086            00023                MOVWF     PORTB           ;Mostra il numero binario ai LEDs connessi
                                                                    ;alla porta B del µPIC Trainer
                                                                    ;du µPIC Trainer
                     00024
                     00025
000A 0063            00026 Termina        SLEEP                     ;Ferma l'esecuzione del programma
                     00027                END                       ;Fine del codice principale
MPASM 01.21 Released ESEMPIO1.ASM 11-20-1996 8:45:06 PAGE 2
SYMBOL TABLE
LABEL VALUE
Inizializza          00000005
PORTB                00000006
Principal            00000008
STATUS               00000003
TRISB                00000006
Termina              0000000A
_16C84               00000001

MEMORY USAGE MAP ('X' = Used, '-' = Unused)
0000 : X--XXXXXX-- ----- ----- -----
0040 : ----- ----- ----- -----

All other memory blocks unused.          Errors : 0    Warnings : 0    Messages : 0
```

codice principale. Infine, dopo il numero di linea compare il codice principale così come è stato inserito.

Alla fine dell'elenco compare un'informazione addizionale che indica il nome delle etichette utilizzate, il valore associato a ognuna, una mappa rappresentativa dello spazio di memoria utilizzato e, per ultimi, il numero di errori, avvisi e messaggi verificatisi.

Si può verificare come le direttive e i commenti del codice principale non generano istruzioni o dati associati per il programma che verrà registrato nella EPROM.

Il primo indirizzo si trova nella linea 12 del file .LST e contiene 0x2805, che è il codice macchina equivalente a GOTO Inizializza. Esaminando attentamente il file .LST è possibile conoscere il codice macchina di ogni istruzione e la sua localizzazione esatta. La mappa di memoria della tabella che segue mostra gli indirizzi della memoria di programma e il loro contenuto.

MEMORIA DI PROGRAMMA

NDIRIZZO	ISTRUZIONE	
0000	2805	(GOTO Inizializza)
0001	.....	
0002	.....	
0003	.....	
0004	.....	
0005	1683	(BSF STATUS,5)
0006	0186	
0007	1283	
0008	3055	
0009	0086	
000A	0063	(SLEEP)
000B	.....	
000C	.....	
000D	.....	
000E	.....	
000F	.....	
03FE	.....	
03FF	.....	

L'istruzione BSF STATUS,5 ha associata anche l'etichetta Inizializza. Il programma assembler assegna l'indirizzo 5 a questa etichetta durante il primo passaggio. Durante il secondo passaggio, qualsiasi istruzione che faccia riferimento a Inizializza sarà sostituita da 5. Per esempio, GOTO Inizializza nella linea 12 si è trasformato in GOTO (28) Inizializza (05).

Alcuni codici del file .LST sono ovvi. 28XX in codice macchina è GOTO, mentre XX86 è STATUS,5. Il compito dell'assembler è convertire tutti gli mnemonici in codice macchina numerico. Questi numeri rappresentano il programma. Di fatto, i numeri sono l'unica cosa che il microcontroller capisce.

5.4.4 Codice Eseguibile

Il file ESEMPIO1.HEX è l'altro file creato dall'assembler, con il codice eseguibile. Questo codice è la forma numerica del programma eseguibile in formato INTEL HEX. Sia il simulatore MPSIM che il dispositivo di registrazione utilizzano questo file per la simulazione e il controllo del programma e anche per la sua registrazione sulla EPROM del microcontroller.

Questo file si può inserire con l'EDIT del DOS e il suo aspetto è simile a quello mostrato di seguito:

```
:020000000528D1
:06000A008316860183123B
:0600100055308600630070C
:00000001FF
```

Per poterlo esaminare è necessario l'equivalente numerico delle istruzioni che formano il programma.

Se si esamina con cura, si trovano 2805 (GOTO Inizializza), 1683 (BSF STATUS,5), ecc, ecc. Il codice eseguibile è memorizzato nel formato della Intel chiamato Merged Hex con i bytes di minor peso seguiti dai bytes di maggior peso (per esempio, 2805 diventa 0528).

5.4.5 Simulazione e Controllo

La fase successiva, importante come le precedenti, è simulare il corretto funzionamento del programma eseguibile, prima di procedere con la registrazione della EPROM del microcontroller.

Esistono varie alternative. La più efficace, ma la più inaccessibile dal punto di vista economico, consiste nell'utilizzo di un Emulatore in tempo reale. Si tratta di un dispositivo professionale, connesso a un personal computer, sul quale si riversa il programma eseguibile perché esso emuli realmente il funzionamento e le reazioni del microcontroller nell'esecuzione del programma. Le istruzioni vengono eseguite in tempo reale alla velocità desiderata e anche le entrate o le uscite sono gestite in tempo reale, in base all'evoluzione dell'esecuzione del programma, ecc. L'emulatore è dotato di un connettore che trasporta tutti i segnali appartenenti al microcontroller e che si connette all'hardware dell'applicazione come se si trattasse di un PIC reale. È dotato di una serie di comandi con i quali si può eseguire passo per passo un programma, definire punti di rottura o Breakpoints, tracciare i diversi segnali che vengono generati nell'hardware sotto prova, ecc. Infine, dispone di una serie di strumenti con i quali mettere a punto un'applicazione, sia nell'hardware che nel software, risparmiando una notevole quantità di tempo per la ripulitura e abbassando così i costi di progetto. Tra gli emulatori più conosciuti, si può citare il PIC MASTER della stessa ARIZONA MICROCHIPS INC. e il ICEPIC della R.F. Solutions Ltd.

L'altra alternativa, più conosciuta, usata e economica è l'impiego dei cosiddetti simulatori software. Essi si basano su programmi che eseguiti su una piattaforma tipo PC/XT/AT, cercano di simulare il funzionamento del programma eseguibile sotto prova e quello del PIC. Oltre a sottolineare che l'esecuzione del programma non è in tempo reale (il PC non ha un PIC

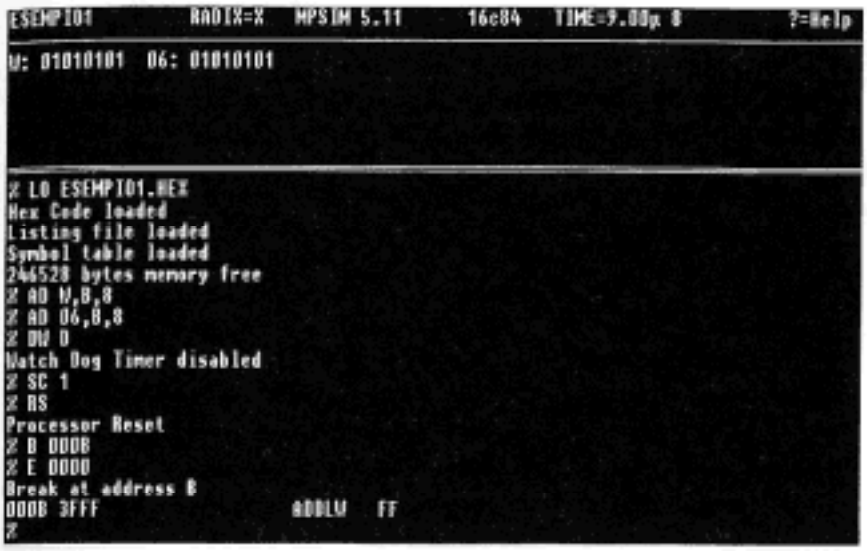


Figura 5-4. Finestra di lavoro del simulatore MPSIM.

si sta simulando (16C84), il tempo trascorso nell'esecuzione del programma (TIME=9.00mS) e il numero dei cicli di istruzione utilizzati (8). Inoltre, consente di entrare nel menu di aiuto (?=Help).

Di seguito, delimitato da due linee orizzontali, appare l'area che consentirà di visualizzare il contenuto dei registri e delle porte di E/U desiderate dall'utente (nell'esempio è mostrato il contenuto del registro W e quello di PORTB).

Infine, nella parte inferiore della finestra, si trova l'area dove l'utente può inserire i diversi comandi e ricevere le corrispondenti risposte da parte del simulatore. Nell'esempio compare la sequenza di comandi che hanno permesso l'esecuzione del programma ESEMPIO1 e che sono descritti qui di seguito.

al suo interno), bisogna aggiungere che il risultato della verifica non è soddisfacente e realistico come con l'emulatore. È necessario più tempo per mettere a punto il programma e individuare i possibili "bugs" che può avere, prima di poter registrare realmente la EPROM del microcontroller. Ciò si ripercuote direttamente sui costi del progetto dell'applicazione finale.

La ARIZONA MICROCHIP INC. mette a disposizione degli utenti diversi simulatori, sia in versione MS-DOS (MPSIM), sia in versione Windows (MPLAB), che possono essere ottenuti dai distributori, oppure entrando direttamente nel sito Internet indicato all'inizio del capitolo.

In ogni caso, questi simulatori, con l'uso dei comandi opportuni (che sono chiaramente documentati), consentono di realizzare operazioni quali:

- visualizzare/modificare registri della memoria dei dati;
- visualizzare/modificare le istruzioni della memori di programma;
- recuperare/memorizzare programmi eseguibili su file;
- esecuzione continua di un programma;
- esecuzione passo per passo e/o con punti di interruzione (Break points);
- stabilire la frequenza di lavoro per la simulazione;
- attivare/disattivare il Watchdog;
- simulare stimoli per i diversi segnali di entrata.

Nella figura 5-4 è mostrata la finestra di lavoro del MPSIM, che sarà descritta di seguito e nella quale si può apprezzare la serie di comandi utilizzati per verificare il funzionamento del programma ESEMPIO1.HEX.

Nella parte superiore compare la linea di stato. In essa vengono indicati il nome del programma oggetto della simulazione (ESEMPIO1), la base di numerazione utilizzata per default (RADIX=X - esadecimale), la versione del simulatore (MPSIM 5.11), il modello di microcontrolier che

- LO ESEMPIO1.HEX: Carica nella memoria del simulatore il codice eseguibile ottenuto durante il processo di assemblaggio e oggetto della simulazione. Carica inoltre il file .HEX, quello di elenco e quello dei simboli, anch'essi generati dall'assembler. Il messaggio che mostra questo comando è:

Hex Code loaded
Listing file loaded
Symbol table loaded

- AD W,B,8: Aggiunge sul visore il registro W in modo che il suo contenuto sia visualizzato in binario (B) e con 8 cifre (8).

- AD PORTB,B,8: Aggiunge sul visore il registro PORTB in modo che il suo contenuto sia visualizzato in binario (B) e con 8 cifre (8).

- DW D: Disattiva l'opzione che simula il funzionamento del Watchdog. Questa opzione dovrà essere realmente scelta nel processo di registrazione del chip, mediante il software di registrazione PICME-TR. La risposta del comando è:

Watch Dog Timer disabled

- SC 1: Definisce la base dei tempi con cui verrà simulata l'esecuzione del programma. Il valore numerico che è associato al comando esprime, in microsecondi, la durata di un ciclo di istruzione (1). Dato che un ciclo di istruzione necessita a sua volta di 4 impulsi del clock principale, in questo esempio si sta simulando un PIC che lavora a 4MHz.

- RS: Questo comando simula un RESET del microcontroller. Ciò determina, tra le altre cose, che si inizia a eseguire dall'indirizzo del vettore di RESET (0000) e che le porte di E/U restano configurate come entrate in alta impedenza. Mostra il messaggio:

Processor Reset

- B 000B: Definisce un breakpoint o punto di interruzione nell'indirizzo esadecimale 000B, che corrisponde all'indirizzo finale del programma ESEMPIO1 soggetto ad analisi.

- E 0000: Esegue il programma dall'indirizzo esadecimale 0000 sino a raggiungere il primo breakpoint (nell'esempio l'indirizzo 000B). Il messaggio di uscita indica l'indirizzo di interruzione e le istruzioni che si trovano in questo punto:

```
Break at address B
    000B    3FFF    ADDLW FF
```

5.4.6 Registrazione del Microcontroller

Una volta ripulito e messo a punto il software mediante l'utilizzo di emulatori e/o simulatori, la fase successiva è quella più attesa. Si deve registrare il programma eseguibile *.HEX nella memoria EPROM del microcontroller. Poi, sarà possibile verificare il funzionamento del programma.

Per poter scrivere e provare i diversi esempi presentati in questo capitolo, sarà necessario almeno un PIC cancellabile. Il PIC 16C84 è basato sulla memoria EEPROM (Electrically Eraseable Read Only Memory) e può essere cancellato rapidamente e facilmente e programmato "in circuito". Il PIC 16F84 è analogo, ma con memoria FLASH.

In questo caso, verrà usato il PIC 16C84. Una volta assemblato il programma ESEMPIO1, si deve connettere il sistema µPIC Trainer al canale parallelo del personal computer. Considerando la capacità di registrazione in circuito di tale sistema, è il momento di inserire il PIC da registrare nello zoccolo a 18 piedini. Connettere l'alimentazione, utilizzando il trasformatore da 15 VAC o le due batterie da 9 VDC.

Bisogna ricordare che nei modelli di gamma media, la registrazione dei PIC si effettua attraverso i piedini RB6 e RB7. Considerando che questi piedini sono a loro volta connessi con le periferiche del µPIC Trainer, è necessario disabilitarli mediante i corrispondenti jumpers J5 (Display), J6 (LCD) e J7 (LEDs), in modo che non agiscano come carico nel momento della registrazione, impedendola.

È possibile che connettendo tra loro il personal computer e il sistema µPIC Trainer si accenda il LED rosso (D5) di quest'ultimo. Non c'è da preoccuparsi: si spegnerà quando il software del registratore ha caricato il controllo del canale parallelo del computer. Si accende ogni volta che si effettua una operazione che implichi la lettura o la scrittura della memoria del PIC.

5.4.7 Memorizzazione del Codice Eseguibile

Dal prompt del DOS bisogna attivare il software del registratore PICME-TR. Si suppone che il software sia stato già installato nella directory di lavoro che è stata utilizzata sino ad ora. Si deve digitare:

```
C:\PIC> PICME-TR  (ENTER)
```

Nel caso non compaia la schermata, bisogna controllare la connessione tra il µPIC Trainer e il canale parallelo e che questo sia quello giusto (LPT1:). Inoltre, dato che il software PICME-TR effettua un controllo dell'hardware del µPIC Trainer, bisogna accertarsi del suo corretto stato.

Sia il µPIC Trainer che il corrispondente software consentono di lavorare con tutti i modelli PIC di gamma media a 18 e 28 piedini. L'utilizzo di tale software e di tutte le sue funzioni è già stato spiegato esaurientemente nel capitolo 3.

Nella finestra di VERIFICA DELLA CANCELLAZIONE, prima di procedere con la registrazione, si può scegliere se si vuole controllare che il PIC sia stato preventivamente cancellato oppure no.

Nella finestra PAROLA DI CONFIGURAZIONE vengono selezionati una serie di parametri.

- Protezione del codice (Codice P). In caso affermativo il PIC non potrà essere letto una volta registrato.
- Attivazione o meno del Watchdog Timer (Watchdog T).
- Scoperta o meno di mancanze nell'alimentazione (Brown En). Questa opzione è disponibile solo nei modelli 16C62X.
- Attivazione o meno del temporizzatore che ritarda l'inizio del funzionamento di fronte alla presenza di tensione di alimentazione (Power Timer).
- Tipo di oscillatore con cui lavorerà il PIC (Oscillatore): economico (RC), alta velocità (HS), standard (XT) e basso consumo (LP).

Nella finestrella ID si può inserire un numero di identificazione a 4 cifre. Questo numero può essere un numero di serie, un codice personale, un checksum, ecc.

Nella parte sinistra della schermata compare una finestra che rappresenta il BUFFER DI MEMORIA. Tale buffer contiene sia le posizioni di memoria che le istruzioni in codice macchina del programma. Il buffer si può riempire con il file eseguibile .HEX che verrà registrato nella EPROM, oppure con il risultato della lettura della EPROM di un PIC già registrato (se non è protetto).

I pulsanti che compaiono in fondo alla finestra del buffer di memoria consentono di effettuare le classiche operazioni di lettura/registrazione.

- Apri File: apre una nuova finestra dove vengono mostrate le diverse directory e i file eseguibili .HEX contenuti. Da qui viene scelto il file da registrare, viene letto e, tutto il contenuto, viene riversato nel buffer di memoria.

- Programma Tutto: trasferisce il contenuto del buffer alla memoria EPROM del microcontroller. È un'operazione che può ritardare un po' di tempo, in quanto dipende dal volume delle informazioni da registrare. In seguito, si registra la parola di configurazione e il numero ID di identificazione stabilito.

- Programma Parola: programma unicamente la parola di configurazione e il numero ID di identificazione.

- Verifica: legge il contenuto della EPROM di un PIC e lo compara con il contenuto del buffer, con l'obiettivo di verificare che entrambi siano uguali. Si usa per verificare la corretta registrazione di un programma.

- Leggi PIC: questo pulsante consente di leggere il contenuto della memoria di programma di un PIC (se non è protetto) e di depositarlo nel buffer di memoria.

- Cancella: questa opzione cancella la memoria di programma del PIC. È disponibile solo se si lavora con il modello 16C84, l'unico con memoria

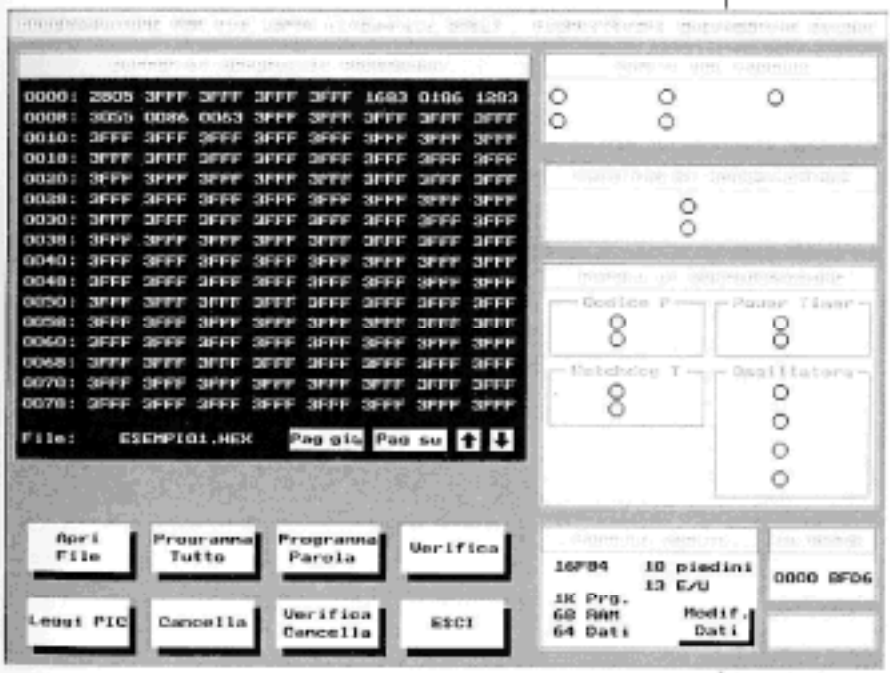


Figura 5-5. Contenuto del buffer quando si carica il programma ESEMPIO1

di tipo EEPROM. Gli altri modelli devono essere cancellati con la luce ultravioletta.

- Verifica cancellazione: effettua la lettura della memoria di un PIC con l'obiettivo di verificare se è cancellato oppure no. Deve essere realizzata prima di procedere con la registrazione.

- ESCI: chiude la sessione con il software PICME-TR di registrazione e riporta il controllo al sistema operativo.

Selezionare l'opzione Apri File: apparirà una schermata che mostra tutte le subdirectory e i file .HEX disponibili. Bisogna scegliere il file ESEMPIO1.HEX e il suo contenuto verrà caricato nel buffer, come si può osservare nella figura 5-5.

Il file selezionato verrà letto dal disco e il suo contenuto caricato nel buffer di memoria.

Selezionare ora il pulsante di programmazione: verranno lette tutte le posizioni di memoria per accertarsi che il microcontroller sia in bianco e non contenga alcuna istruzione. In seguito, si passa all'operazione vera e propria di registrazione. Nel caso il microcontroller non sia cancellato, comparirà un avviso in cui si chiede conferma per continuare.

Dopo alcuni secondi di attesa comparirà il messaggio "Programmazione Completata".

Durante il processo di registrazione avverrà la verifica del contenuto di ogni posizione di memoria, mediante la lettura e la comparazione con il contenuto del buffer. Se esiste qualche posizione che non è correttamente registrata, comparirà un messaggio di avvertimento dell'errore.

5.4.8 Esecuzione del Programma

Dato che il microcontroller è stato registrato in circuito sul sistema µPIC Trainer, il programma verrà eseguito immediatamente, cioè non appena il registratore ha terminato di lavorare. È necessario abilitare le periferiche di uscita che avevano dovuto essere sconnesse mediante i jumpers J5 (Display), J6 (LCD) e J7 (LEDs) durante la registrazione, per poter vedere l'effetto prodotto dall'esecuzione di tale programma su questi elementi. Nell'esempio devono essere abilitati i LEDs (J7), dove comparirà il numero binario 01010101.

Se i LEDs non sono accesi correttamente, premere il pulsante di Reset del sistema µPIC Trainer, che si trova nell'angolo inferiore destro. Ciò dovrebbe reinizializzare il microcontroller per eseguire nuovamente il programma appena registrato.

Se tutto quanto appena spiegato risulta corretto, congratulazioni per aver scritto, assemblato, ripulito e programmato il vostro primo microcontroller PIC!

Proseguendo nella lettura del testo, sarà necessario riprogrammare il microcontroller con i diversi esempi proposti. Nel caso si utilizzi il PIC 16C84, non sarà necessario toglierlo dallo zoccolo della scheda µPIC Trainer, per poterlo riprogrammare tutte le volte che si vuole.

Chiaramente, nel caso di qualsiasi altro modello di PIC con memoria EPROM e finestra, bisognerà toglierlo dallo zoccolo e esporlo ai raggi UV per cancellare la EPROM. Quando si usa un PIC con queste caratteristiche è opportuno mantenere coperta la finestra una volta registrata la EPROM, dato che il silicio è fotosensibile e la luce che la attraversa può influenzare il suo funzionamento. Un buon sistema per evitarlo è utilizzare un pezzo di nastro isolante nero.

5.5 SECONDO ESEMPIO

In questo nuovo esempio dimostrativo si utilizza l'interrupt provocata dal TMR0 per fare in modo che il LED connesso all'uscita RB7 realizzi una intermittenza di 250mS. Contemporaneamente viene letto lo stato delle entrate (interruttori) connesse a RA0 e RA1 per mostrare il loro livello logico nei LEDs di uscita connessi rispettivamente a RB0 e RB1.

Si usa il PIC 16C84, con una frequenza di lavoro di 4MHz.

5.5.1 PROGRAMMA PRINCIPALE

è quello presentato di seguito. Deve essere scritto con un elaboratore di testi che salvi le informazioni su file in formato ASCII.

ESEMPIO 2
Prova di interrupt

Si tratta di verificare l'interrupt provocata dal TRM0. Il programma legge lo stato dei pulsanti connessi a RA0 e RA1, per mostrarlo sui LEDs connessi rispettivamente a RB0 e RB1. Contemporaneamente, il TRM0 genererà un'interrupt ogni 0,25 secondi (250mS), con l'obiettivo di creare un'intermittenza sul LED connesso a RB7. Si usa il modello PIC 16C84 a una frequenza di 4MHz, il cui ciclo di clock è di 250mS e il ciclo di istruzioni di 1mS.

```
            list    p=16c84     ;Microprocessore PIC16c84
            list    c=132       ;Elenco a 132 caratteri
TIMER0      equ     01          ;Registro del TIMER0
OPTION      equ     01          ;Registro delle opzioni
STATUS      equ     03          ;Registro di stato
RA          equ     05          ;Porta A
RB          equ     06          ;Porta B
TRISA       equ     05          ;Registro di programmazione della porta A
TRISB       equ     06          ;Registro di programmazione della porta B
INTCON      equ     0xb         ;Registro di controllo degli interrupt
COMPT       equ     0xc         ;Contatore di ripetizione
            org     0           ;Vettore di RESET
            goto    INIZIO
            org     4           ;Vettore di interrupt
            goto    INTER
            org     5
INIZIO      bsf     STATUS,5    ;Seleziona il banco 1 dei dati
            clrf    TRISB       ;RB è programmato come uscita
            movlw   b'00000011'
            movwf   TRISA       ;RA<1:0> sono programmati come entrate
            movlw   b'00000011'
            movwf   OPTION      ;Assegna il prescaler al TMR0
            bcf     STATUS,5    ;Seleziona il banco 0 dei dati
```

Il TIMER0 si incrementa ogni 4 impulsi di clock (250mS*4=1mS a 4MHz). Il prescaler scelto è di 256 e gli incrementi si verificano ogni 256mS. Se si carica con il valore 12, mancheranno 244 incrementi per superare il tempo e provocare l'interrupt. Ciò succede a 62.464mS e, se si ripete 4 volte, la temporizzazione totale sarà di 249.856mS.

```
            movlw   b'10100000'
            movwf   INTCON      ;Attiva l'interrupt del TMR0
            movlw   4
            movwf   COMPT       ;Valore da ripetere ogni interrupt
            movlw   12
            movwf   TIMER0      ;Carica il TIMER0
            clrf    RB          ;Disattiva i LEDs della porta B
START       btfsc   RA,0        ;Testa lo stato di RA0
            goto    RA0_EST_1
            bcf     RB,0        ;Disattiva RB0
            goto    TEST_RB1
RA0_EST_1   bsf     RB,0        ;Attiva RB0
TEST_RB1    btfsc   RA,1        ;Testa lo stato di RA1
            goto    RA1_EST_1
            bcf     RB,1        ;Disattiva RB1
            goto    LOOP
RA1_EST_1   bsf     RB,1        ;Attiva RB1
LOOP        goto    START
INTER       decfsz  COMPT,1     ;Decrementa CONTA
            goto    CONTINUER
COMPT_SI_0  movlw   4
            movwf   COMPT       ;Mostra nuovamente il contatore
            btfsc   RB,7        ;Testa l'ultimo stato di RB7
            goto    RB7_EST_1
            bsf     RB,7        ;Attiva RB7
            goto    CONTINUER
RB7_EST_1   bcf     B,7         ;Disattiva RB7
CONTINUER   movlw   b'10100000'
            movwf   INTCON      ;Mostra il registro degli interrupt
            movlw   12
            movwf   TIMER0      ;Mostra il TIMER0
            return
            end
```

5.5.2 Fasi da Seguire

1. Scrivere con l'aiuto di un elaboratore di testi il programma principale presentato in precedenza. Si può usare l'EDIT del MS-DOS.

2. Mediante il programma assembler MPASM, assemblarlo per ottenere il file eseguibile (*.HEX), quello di elenco (*.LST), quello degli errori (*.ERR), ecc. Se comparissero degli errori di scrittura o degli avvisi (Warnings), consultare il file degli errori (*.ERR) per verificare il tipo di errore e in che punto del programma principale si è verificato. Correggerlo e ricominciare a inserire i dati.

3. Anche se il programma elencato in precedenza funziona correttamente, è comunque interessante per l'utente utilizzare il simulatore MPSIM per assicurarsi che ciò sia vero e per fare pratica con questo simulatore.

4. Connettere, se ancora non lo fosse, il sistema di valutazione µPIC Trainer al canale parallelo del computer, inserire il PIC nel corrispondente zoccolo e lanciare il programma PICME-TR, per procedere con la registrazione dell'esempio proposto. Ricordarsi che durante la registrazione è necessario che le periferiche connesse alla porta B siano disabilitate mediante i rispettivi jumpers.

5. Verificare il funzionamento dell'applicazione. Sarà necessario riconnettere i jumpers che attivano le periferiche della porta B e, se è il caso, effettuare il RESET mediante il relativo pulsante.

5.6 TERZO ESEMPIO

In questo caso si usa il PIC 16C71, per poter utilizzare il convertitore A/D di cui è dotato. Si tratta di mostrare nei LEDs di uscita connessi alla porta B il valore binario che proviene dai canali analogici AN0 e AN1.

Nel sistema di valutazione µPIC Trainer, essi sono connessi mediante potenziometri ai segnali RA0 e RA1 del PIC 16C71. Il valore analogico introdotto per questi segnali sarà visualizzato in binario, dopo la conversione, nei LEDs di uscita.

Il segnale di entrata RA2 funziona come entrata digitale. È collegato a un interruttore di entrata che, a seconda valga un livello "0" o "1", visualizzerà nei LEDs il valore analogico del canale 0 o del canale 1.

5.6.1 Programma Principale

è quello presentato di seguito.

```
**********************************************************
                      ESEMPIO 3
                  I convertitori A/D
**********************************************************
```

L'esempio che segue è un'applicazione con il microprocessore PIC 16C71 della Microchip.L'applicazione consiste nel rendere operativi i convertitori A/D (RA0-AN0 e RA1-AN1), le cui uscite saranno visualizzate in RB<0-7> (LEDs) selezionabili mediante un interruttore in RA2.

La frequenza di lavoro è di 4MHz, con un periodo di 250mS.

```
**********************************************************
        list P =16C71                ;µC PIC16C71
        list C =132                  ;Elenco a 132 caratteri

OPTION  equ     01h                  ;Registro delle opzioni
STATUS  equ     03h                  ;Registro di stato
PORT_A  equ     05h                  ;Porta A
PORT_B  equ     06h                  ;Porta B
TRIS_A  equ     05h                  ;Reg. di programmazione di PA
TRIS_B  equ     06h                  ;Reg. di programmazione di PB
ADCON0  equ     08h                  ;Reg. di programmazione e controllo del ADC
ADCON1  equ     08h                  ;Reg. di programmazione del ADC
ADDRES  equ     09h                  ;Reg. di memorizzazione della con
                                     ;versione
INTCON  equ     0Bh                  ;Reg. di controllo degli interrupt
COMPT   equ     0Ch                  ;Variabile di temporizzazione
**********************************************************
0       org     0                    ;Vettore di Reset.
        goto    Inizio               ;Salta all'inizio

        org     5                    ;Indirizzo di inizio
********** Programma principale **********
INIZIO  bsf     STATUS,5             ;Seleziona il banco 1
        clrf    TRIS_B               ;RB<7-0> uscita del convertitore A/D.

        movlw   B'00000111'          ;RA0,RA1 e RA2 come entrate
        movwf   TRIS_A               ;dell'interruttore e del convertitore A/D

        movlw   B'00000010'          ;Configurazione dei piedini del ADCON1 come
        movwf   ADCON1               ;entrate. RA0 e RA1 = Analogiche, RA2 = Digitale
                                     ;et Vréf=Vdd.
        movlw   B'10000000'          ;Programmazione del registro delle opzioni
        movwf   OPTION

        movlw   B'00000000'          ;Disattiva gli interrupt
        movwf   INTCON

        bcf     STATUS,5             ;Seleziona il banco 0
**********Routine di scelta del convertitore**********
QUIEN_ES  btfss   PORT_A,2           ;è il bit RA2 = 1 (E digitale)
          goto    EST_AN_0           ;NO
********** Routine di configurazione del convertitore**********
          movlw   B'10001001'        ;SI. Attiva il convertitore, cancella ADIF,GO
          movwf   ADCON0             ;(ADCON0<2>)1 canale RA1/AN1 e Fosc/32
          goto    NON_FIN
EST_AN_0  movlw   B'10000001'        ;Attiva convertitore, cancella ADIF,GO
          movwf   ADCON0             ;(ADCON0<2>)1 canale RA0/AN0 e Fosc/32

**********Routine di attesa di fine conversione**********
NO_FIN    movlw   B'00001010'
          movwf   CONFTA             ;Carica il contatore
DELAY     decfsz  CONFTA,1
          goto    DELAY
```

```
            bsf     ADCON0,2        ;Attiva convertitore (GO)
NO_FIN_1    btfsc   ADCON0,2        ;¿ DONE(ADCON0<2>)=0 (Fine della conversione)?
            goto    NON_FIN_1       ;NO. Continua ad aspettare la fine della conversione

********** Routine di campionamento in PB **********

            movf    ADDRES,0        ;ADDRES==>W. Raccoglie il risultato della
            movwf   PORT_B          ;conversione e lo porta a RB<0-7>
            bcf     ADCON0,1        ;Abbassa il flag ADIF
            goto    QUIEN_ES        ;Ricomincia il ciclo

            end
```

5.6.2 Fasi da Seguire

1.Scrivere con l'aiuto di un elaboratore di testi il programma principale presentato in precedenza. Si può usare l'EDIT del MS-DOS.

2.Mediante il programma assembler MPASM, assemblarlo per ottenere il file eseguibile (*.HEX), quello di elenco (*.LST), quello degli errori (*.ERR), ecc. Se comparissero degli errori di scrittura o degli avvisi (Warnings), consultare il file degli errori (*.ERR) per verificare il tipo di errore e in che punto del programma principale si è verificato. Correggerlo e ricominciare a inserire i dati.

3. Anche se il programma elencato in precedenza funziona correttamente, è comunque interessante per l'utente utilizzare il simulatore MPSIM per assicurarsi che ciò sia vero e per fare pratica con questo simulatore.

4. Connettere, se ancora non lo fosse, il sistema di valutazione µPIC Trainer al canale parallelo del computer, inserire il PIC nel corrispondente zoccolo e lanciare il programma PICME-TR, per procedere con la registrazione dell'esempio proposto. Ricordarsi che durante la registrazione è necessario che le periferiche connesse alla porta B siano disabilitate mediante i rispettivi jumpers.

5. Verificare il funzionamento dell'applicazione. Connettere i canali analogici AN0 e AN1, mettendo i jumpers J8(0) e J8(1) nella posizione analogica e J8(2) nella posizione digitale, in modo che SW4 funzioni come entrata digitale RA2.

6. Chiudere il jumper J7 per abilitare i LEDs di uscita connessi alla porta B.

5.7 QUARTO ESEMPIO

In questo caso, si tratta di gestire il modulo di visualizzazione LCD. Si tratta di un modulo che permette di rappresentare 2 righe di 16 caratteri alfanumerici ognuna.

Come già stato spiegato in precedenza nel capitolo 4, l'interfaccia tra il modulo LCD e il sistema di valutazione µPIC Trainer viene realizzato utilizzando la porta B per inserire i diversi caratteri, i codici di controllo e le linee RA0, RA1 e RA2 della porta A, per generare i segnali di controllo RS, R/W e E.

Il programma consiste nella presentazione sul modulo LCD di una serie di messaggi relativi alle caratteristiche del µPIC Trainer, con differenti effetti di visualizzazione.

5.7.1 Programma Principale

Dato che l'elenco è piuttosto lungo, il programma principale viene fornito, come gli esempi precedenti, nel dischetto dato con il sistema µPIC Trainer.

5.7.2 Fasi da Seguire

1. Scrivere con l'aiuto di un elaboratore i testi il programma principale ESEMPI04.ASM, che si trova nel dischetto. Si può usare l'EDIT del MS-DOS.

2. Mediante il programma assembler MPASM, assemblarlo per ottenere il file eseguibile (*.HEX), quello di elenco (*.LST), quello degli errori (*.ERR), ecc. Se comparissero degli errori di scrittura o degli avvisi (Warnings), consultare il file degli errori (*.ERR) per verificare il tipo di errore e in che punto del programma principale si è verificato. Correggerlo e ricominciare a inserire i dati.

1. Connettere, se ancora non lo fosse, il sistema di valutazione µPIC Trainer al canale parallelo del computer, inserire il PIC nel corrispondente zoccolo e lanciare il programma PICME-TR, per procedere con la registrazione dell'esempio proposto. Ricordarsi che durante la registrazione è necessario che le periferiche connesse alla porta B siano disabilitate mediante i rispettivi jumpers.

BIBLIOGRAFIA

1.- Microcontrollori PIC. Descrizione e preparazione Christian Tavernier, Dunod, 1995
ISBN: 2-10-002647-X

2.-Microcontrollori PIC Disegno pratico di applicazioni. Dodicesima edizioneContiene CD con software, programmmi e esercitazioni del Micro_PIC Trainer, McGraw-Hill, 1999

Prima parte: PIC16F84 ISBN: 84-481-2496-0
Seconda parte: PIC16F87X ISBN: 84-481-2858-3