LA PROGRAMMAZIONE IN ASSEMBLER

Iniziamo da questo numero la pubblicazione mensile di un corso completo di programmazione relativa ai chip della famiglia PIC. Chi meglio di noi, primi in Italia a presentare questi microprocessori, poteva proporre un servizio del genere?

Andrea Sbrana - 1ª parte
su gentile concessione della Microlabs

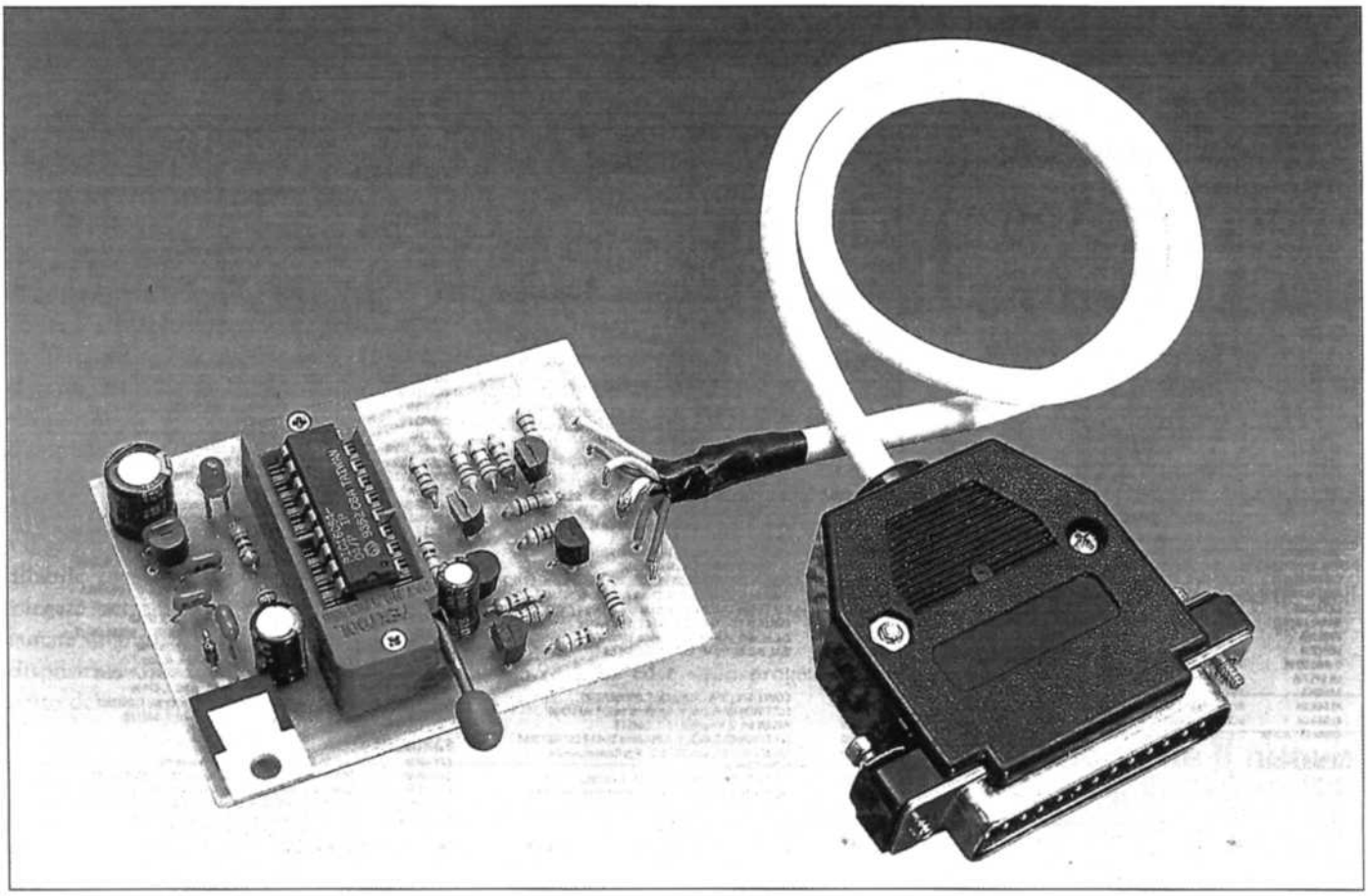
L'evoluzione della tecnologia ha fatto sì che i microcontroller, fino a pochi anni fa completamente sconosciuti, siano oggi abitualmente impiegati non solo in apparati commerciali, ma anche in circuiti dedicati e realizzati da hobbisti e sperimentatori, data la facilità con cui vengono programmati.

Ma la maggior parte degli hobbisti elettronici, tradizionalmente ha sempre avuto una certa difficoltà con il software, indispensabile per poter sfruttare questi chip, quindi abbiamo deciso di dedicare una serie di puntate alla programmazione in assembler che, anche se finalizzata al PIC16C84 della Microchip, risulterà molto utile per l'apprendimento di concetti che sono al di sopra del componente impiegato.

In questa prima puntata, illustreremo un programmatore per PIC16C84 molto semplice, da collegare alla porta parallela di un qualsiasi computer IBM compatibile e di costo estremamente ridotto, dato l'esiguo numero di componenti.

Questo prodotto ci è stato gentilmente concesso dalla ditta Microlabs, Via Circonvallazione, 11 20060 Masate (MI), telefono 02/95762168 oppure 0336/408090 che si è resa pienamente disponibile anche per risolvere eventuali problemi dei nostri lettori relativi a qualsiasi tipo di microcontroller della Microchip.

A tale proposito potrete telefonare direttamente all'Ing. Gordan Rancic, che segue costantemente l'evoluzione della famiglia PIC.



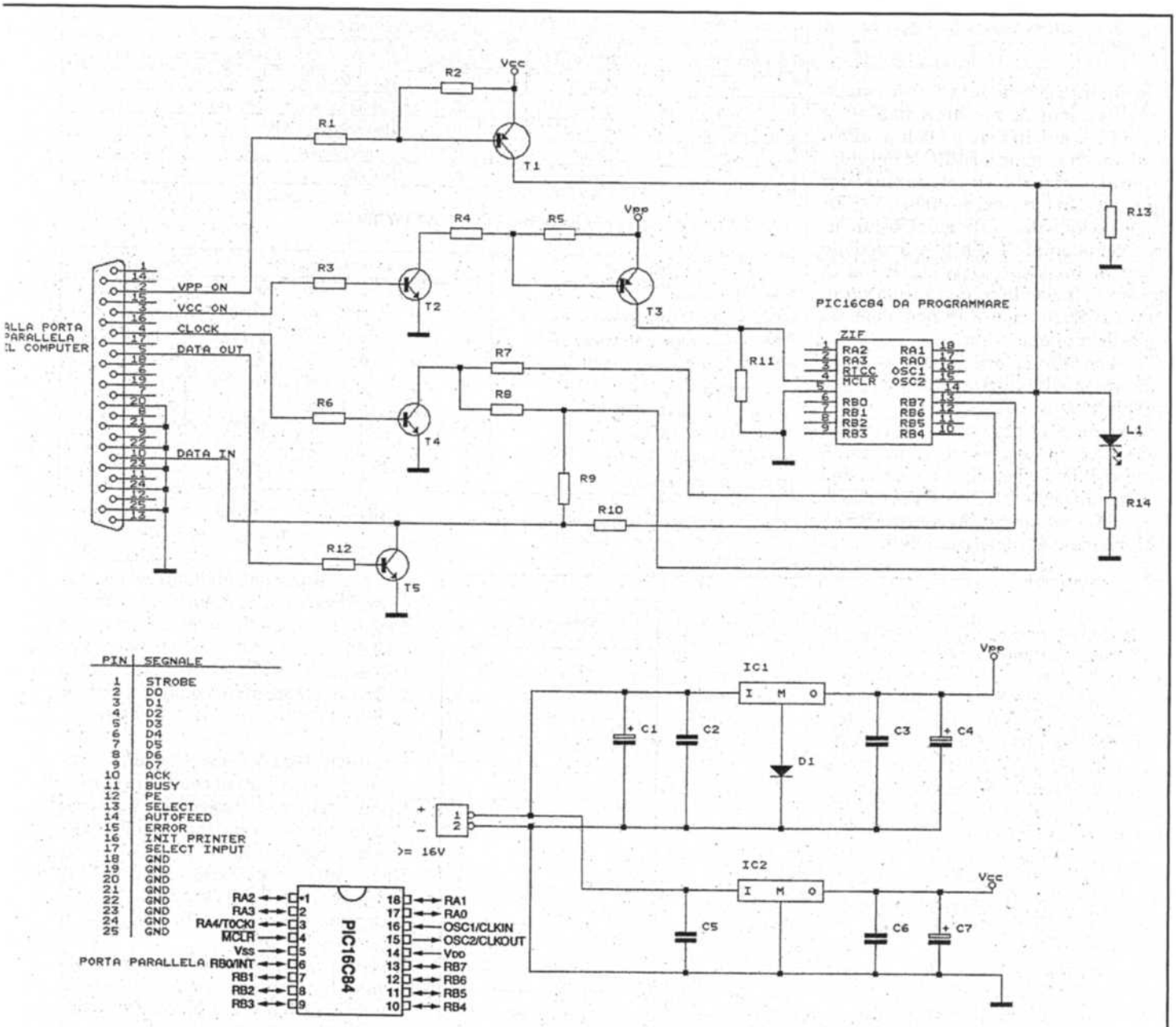


Figura 1. Schema elettrico del programmatore

Funziona così

Passiamo quindi a vedere le caratteristiche del programmatore, iniziando dallo schema elettrico di Figura 1.

Dalla porta parallela del computer, vengono prelevati 4 segnali (D0, D1, D2 e D3) che comandano le funzioni del programmatore, mentre un segnale entra nel computer (ACK) per avere la possibilità di uno scambio dati seriale.

Il segnale D0 viene impiegato per abilitare la tensione di alimentazione Vcc per il 16C84 attraverso il transistor T1.

Quando D0 è a basso livello, sul pin 14 del PIC è presente una tensione di 5 volt.

La tensione di programmazione di 12,8 V invece, viene abilitata con il segnale D1 per mezzo dei due transistor T2 e T3.

I due segnali D3 e D4 simulano rispettivamente il segnale di clock e di dato per il PIC.

I dati che provengono dal PIC, sono invece inviati sulla linea ACK del computer per mezzo di three-state implementato con il transistor T5.

Le due alimentazioni per la programmazione vengono stabilizzate da due regolatori appositi, IC1 e IC2. Si nota che IC1 ha il riferimento di massa elevato con un diodo, per ottenere i 12,8 V necessari alla tensione di programmazione.

Per questo motivo, la tensione di ingresso del programmatore non potrà mai stare al di sotto di 16 V, pena il non corretto funzionamento del circuito.

Sempre in Figura 1 vediamo le connessioni relative al connettore DB25 della porta parallela e quelle relative alla piedinatura del PIC16C84.

La programmazione del PIC

In Figura 2 possiamo vedere la mappa delle locazioni di memoria impiegate nel 16C84: dall'indirizzo 000h al 3FFh troviamo la memoria EPROM del chip, ovvero dove risiederà il programma vero e proprio. Gli indirizzi da 400h a 1FFFh non sono implementati, mentre dall'indirizzo 2000h al 200Fh troviamo delle locazioni in parte scrivibili, in parte riservate, fra cui la configuration word.

Poi, da 2010h a 3FFFh non abbiamo più celle implementate.

Il programmatore dovrà, quindi, lavorare sugli indirizzi 000h-3FFh e 1FFFh-2000h.

Durante la programmazione, i pin del PIC16C84 interessati sono soltanto 5, come visibile in Tabella 1a, e cioè i due di alimentazione, il MCLR, l'RB6 e l'RB7.

L'RB6 accetta il clock, mentre l'RB7 è bidirezionale e trasferisce i dati.

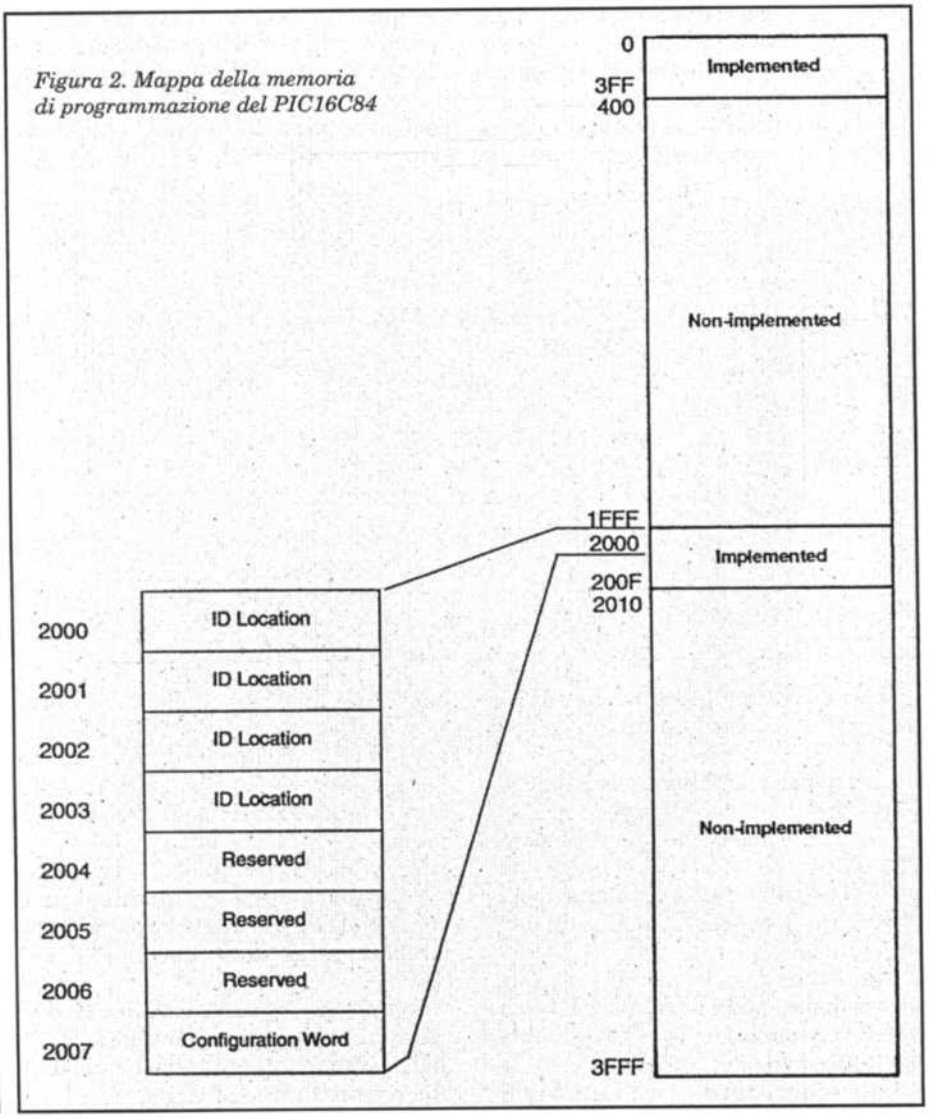
Pin Name	During Programming		
	Pin Name	Pin Type	Pin Description
RB6	CLOCK	I	Clock input
RB7	DATA	I/O	Data input/output
$\overline{MCLR}$	VTEST MODE	P*	Program Mode Select
VDD	VDD	P	Power Supply
VSS	VSS	P	Ground

Tabella 1a. Segnali per la programmazione del PIC16C84

Command	Mapping (msb ... lsb)						Data
Load Configuration	0	0	0	0	0	0	0, data (14), 0
Load Data for Program Memory	0	0	0	0	1	0	0, data (14), 0
Read Data from Program Memory	0	0	0	1	0	0	0, data (14), 0
Increment Address	0	0	0	1	1	0	
Begin Programming	0	0	1	0	0	0	
Load Data for Data Memory	0	0	0	0	1	1	0, data (14), 0
Read Data from Data Memory	0	0	0	1	0	1	0, data (14), 0
Bulk Erase Program Memory	0	0	1	0	0	1	
Bulk Erase Data Memory	0	0	1	0	1	1	

Tabella 1b. Mappa dei comandi implementati nel PIC16C84

Figura 2. Mappa della memoria di programmazione del PIC16C84



Il MCLR permette di entrare in programmazione semplicemente in funzione della tensione che vede in ingresso (5 o 12,8).

I comandi che è possibile inviare a PIC sotto programmazione sono visibili in Tabella 1b.

Load Configuration: permette di caricare il registro di configurazione
Load Data for Program Memory: consente di caricare una cella di memoria del programma
Read Data from Program Memory: legge una cella di memoria
Increment Address: fa avanzare il program counter
Begin Programming: fa iniziare la fase di programmazione di una cella
Load Data for Data Memory: carica una cella della memoria dati
Read Data from Data Memory: legge una cella della memoria dati
Bulk Erase Program Memory: cancella la memoria di programma
Bulk Erase Data Memory: cancella la memoria dati.

Il protocollo di invio e ricezione dati, è visibile rispettivamente in Figura 3a e 3b.

Come si vede chiaramente, tutte le informazioni passano sui due segnali RB6 (clock) e RB7 (data).

In Figura 5 vediamo che cosa dovremo scrivere nella configuration word per settare il modo di funzionamento del PIC: i bit 0 e 1 indicano il tipo di oscillatore scelto, il bit 2 se il watchdog deve

essere abilitato o meno, il bit 3 se vogliamo inserire un ritardo di 65 mS ad ogni partenza del chip (per un reset più sicuro) ed, infine, il bit 4 deve essere impostato per abilitare o meno la protezione contro letture non permesse del programma.

Uno sguardo al software di programmazione

Il programma che va inserito nel computer per poter sfruttare la nostra interfaccia, è free-ware, ovvero può essere copiato da tutti, ma non per essere rivenduto.

A tale proposito, per ottenerlo potrete telefonare alla Microlabs, che potrà inviarvelo gratuitamente (escluso chiaramente il costo del dischetto e delle spese postali) corredato eventualmente del circuito stampato e di alcuni PIC per le vostre prove che farete durante il corso di programmazione.

Ricordiamo che il PIC16C84 è elettricamente cancellabile e riscrivibile, quindi anche se sbaglierete un programma, potrete riprogrammare il chip senza necessità di lampade per EEPROM.

Schermata 1

```
:    PRO84 - Microchip PIC 16C84 programmer Driver                    :
: Copyright (c) MicroLabs 1995. All rights reserved.                  :
:    Written by Gordan RANCIC for PROGETTO ELEKTOR                    :
:    Shareware version 1.0 - 10. September 1995                       :

USAGE: PRO84 <FILENAME> /arguments... /....
<FILENAME> is the name of the DOS file for READ/WRITE in INHX8M format
ARGUMENTS:  /P       = program                              default
            /F       = program fuses only
            /C       = erase device only
            /E       = eeprom program only
            /NOEE    = do not program/read eeprom
            /NOID    = Do not program/verify ID code
            /R       = read device to file
            /NB      = no blank check                       BLANK CHECK
            /B       = blank check only
            /V       = verify only
            /OSC:XT  = oscillator type LP/RC/XT/HS          XT
            /WD:ON   = watch dog ON/OFF                     ON
            /CP:ON   = code protection                      OFF
            /TU:ON   = power up timer OFF/ON                ON
            /LPT2    = paralel port LPT1, LPT2              LPT1
```

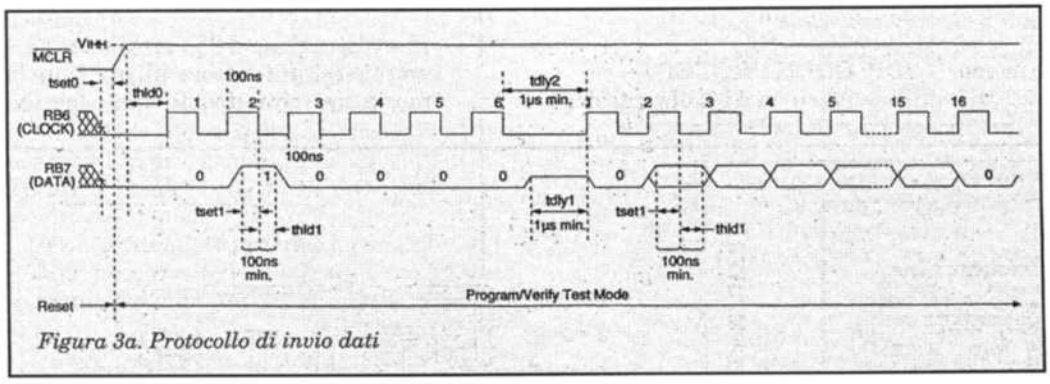


Figura 3a. Protocollo di invio dati

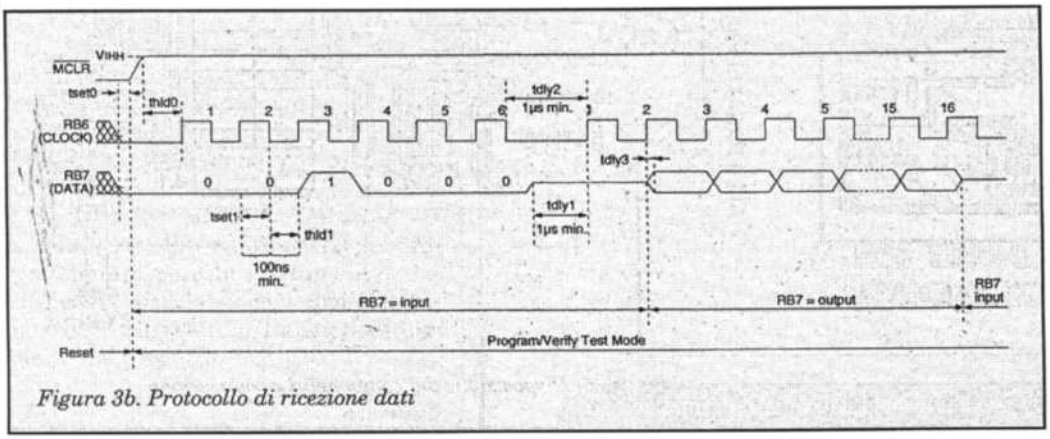


Figura 3b. Protocollo di ricezione dati

Prima di vedere la sintassi di programmazione ricordiamo due cose: il software di programmazione è stato realizzato per tutti i tipi di computer, quindi anche per i più lenti. Se impiegherete 486DX ad elevata velocità oppure pentium, dovrete togliere l'opzione "Turbo".

Inoltre il software per l'assemblaggio del codice sorgente, è anch'esso sharewware e disponibile assieme a questo programma.

La ditta Microlabs inoltre, ha implementato un simulatore grafico che potrete richiedere ad un prezzo molto basso, specificando di essere lettori di Progetto.

Se digitate al prompt il solo comando:

> PRO84

apparirà la **Schermata 1**, visibile in questa stessa pagina.

Tutti i parametri di programmazione possono essere inseriti senza rispettare un ordine particolare se non quello relativo al nome del file, che deve essere inserito per primo.