ASSEMBLER 8086 8088

di Pierluigi Panunzi

Le direttive

Continuiamo in questa puntata a parlare degli elementi costitutivi di un programma assembler 86/88 intendendo con tale termine gli elementi «sintattici» con cui si scrive un programma in assembler.

Riprenderemo il discorso terminato la scorsa puntata esaminando in quale modo possono essere inizializzate le variabili. Infine introdurremo il discorso delle regole sintattiche secondo le quali si possono scrivere le istruzioni e le direttive di un programma in assembler.

Le variabili

Con tale termine si intende un dato posto in una certa locazione di memoria, sul quale eseguiamo le operazioni indicate dal nostro programma.

Dal momento che si parla di oggetti residenti nella memoria, ecco che allora le variabili avranno innanzitutto due attributi fissi, rappresentati dal segment e dall'offset della locazione di memoria dove il dato stesso risiede: già abbiamo sottolineato più volte l'importanza del concetto di segment ed offset di una cella di memoria, concetto sul quale si basa infatti tutta la logica dell'86/88.

Il terzo attributo, questa volta variabile a seconda della scelta del programmatore è il tipo della variabile stessa, che a sua volta indica da quanti byte è formata la variabile nella sua interezza: il tipo può essere a scelta tra quelli indicati nella tabellina 1, in cui viene riportata la lunghezza della variabile in byte e la relativa direttiva dell'assembler.

Per quanto riguarda le direttive, ricordiamo che non si tratta di istruzioni particolari che l'assembler può eseguire, ma viceversa sono indicazioni che si devono fornire all'assembler per poter effettuare subito o in seguito certe determinate operazioni: in particolare con la direttiva D* si specifica appunto l'ampiezza di una certa variabile in termini di occupazione di byte, a seconda della lettera che poniamo al posto dell'asterisco (*).

Altra funzione di queste direttive, a scelta del programmatore, è quella di inizializzare le celle di memoria oltreché a definirle a livello logico.

Già abbiamo visto che se non si vuole inizializzare con un certo valore una variabile allora si usa un «?» subito dopo la direttiva: ad esempio con

```
ALFA DW?
```

indicheremo all'assembler che vogliamo riservare due byte per la variabile ALFA, della quale non ci interessa assegnare un valore specifico all'inizio: in questo modo, bisogna però stare attenti, l'assembler vi carica un valore qualsiasi, casuale, che poi il nostro programma dovrà andare a sovrapporre con il valore desiderato.

Se invece vogliamo solamente definire la variabile, senza che nemmeno venga toccata dall'assembler, allora dobbiamo usare al posto del generico «?» la particolare espressione

```
contatore DUP (?).
```

Ad esempio con

```
ALFA DW 1 DUP(?)
```

istruiremo l'assembler di riservare due byte per la variabile ALFA senza però consentirgli di depositarvi dei valori casuali: questo fatto si rivela importantissimo allorché dobbiamo inserire un certo programma all'interno di un ambiente già esistente, tipicamente quando vogliamo scrivere un programma in assembler da far eseguire senza alterare il contenuto di celle di memoria preesistenti.

Tabella 1

Tipo	num. byte	direttiva
BYTE	1	DB
WORD	2	DW
DOUBLEWORD	4	DD
QUADWORD	8	DQ
TENBYTE	10	DT
STRUCTURE	variabile	STRUC

È il caso ad esempio di un programma che deve sfruttare i dati forniti da un programma precedente e residenti in memoria: scrivendo il programma in assembler, l'unico modo per far si che i dati vengano definiti senza essere alterati dal caricamento del programma stesso è come detto di usare la «DUP(?)», al limite anche con il contatore posto ad 1.

Ecco che i «buffer del disco» (in genere da 512 byte) potranno essere definiti sia con istruzioni del tipo

```
DSK_BUFF   DB   512   DUP(?)
```

sia con istruzioni del tipo

```
DSK_BUFF   DB   512   DUP(0)
```

nel qual caso, viceversa, si assegnerà un ben preciso valore (0) da porre nelle 512 locazioni costituenti la zona di memoria chiamata DSK_BUFF.

Tornando invece all'inizializzazione con valori prefissati dal programmatore, allora al «?» si può sostituire una generica espressione sia numerica che un'«address expression».

Per quanto riguarda le espressioni numeriche, non c'è niente di particolare da dire se non che sono permesse tutte e quattro le operazioni matematiche eventualmente con l'uso di parentesi, gli operatori logici (AND, OR e XOR), gli opertori relazionali (EQ, LT, LE, GT, GE e NE: rispettivamente EQual, Less Then, Less or Equal, Greater Than, Greater or Equal, Not Equal) ed altre funzioni logiche (resto di divisione e shift).

Invece per quanto riguarda le «address expression», ci troviamo ancora una volta di fronte ad una caratteristica dell'assembler che stiamo studiando: in particolare con tale termine si intende un'espressione il cui risultato è un indirizzo secondo lo «stile

86/88» e cioè considerato come accoppiata offset-segment, ottenuto a partire tanto da quantità numeriche quanto da indirizzi.

Va detto subito che le AE (abbreviazione che useremo invece di scrivere «address expression») si usano solo con le direttive DW e DD: ma per spiegarci su questi concetti preferiamo far parlare alcuni esempi.

Supponiamo di avere un certo segmento di dati, chiamato «DATI», contenente un certo numero di variabili definite con le direttive D*: vediamo dunque questo frammento di programma che analizzeremo subito in dettaglio:

```
0000                        DATI    SEGMENT  AT    1000H
0000  00                    UNO     DB       0
0001  1237                  DUE     DW       1234H+25/7
0003  0003                  TRE     DW       TRE
0005  0000                  QUATTRO DW       UNO
0007  0007  1000            CINQUE  DD       CINQUE
000B  0005  1000            SEI     DD       QUATTRO
000F                        DATI    ENDS
                                    END
```

Abbiamo dunque creato sei variabili, di tipo byte, word e doubleword in un segmento di dati del quale abbiamo specificato il segmento di appartenenza (1000H) con la direttiva AT posta subito dopo la direttiva SEGMENT: su tale possibilità ritorneremo senz'altro nei dettagli quando ci occuperemo della sintassi di «SEGMENT».

In questo caso ci basta sapere che di solito il segmento non viene prefissato dal programmatore se non per scopi particolari,mentre viceversa si lascia prima all'assembler e poi al linker il compito di settarlo in modo da poter sempre lavorare con programmi completamente rilocabili che verranno allocati solo all'atto dell'esecuzione.

Cominciamo dunque ad analizzare linea per linea cosa abbiamo indicato con le D*:

UNO DB 0 non fa altro che assegnare alla variabile UNO, di tipo byte, il valore 0: è questa dunque un'assegnazione di una generica espressione ad una variabile.

DUE DW 1234H + 25/7 assegna alla variabile DUE di tipo word il risultato dell'espressione 1234H+25/7, nella quale, come si vede, si possono mischiare quantità in basi differenti.

TRE DW TRE invece, come i successivi esempi, contiene un'address expression in quanto «TRE» è stata definita proprio con una direttiva DW come l'etichetta di una variabile: quando l'assembler, dopo la DW, incontra il simbolo «TRE» la riconosce come un'etichetta (cioè ne conosce l'offset, da tenere sempre bene in mente!!) e dunque come valore da assegnare alla variabile TRE associa proprio il suo offset all'interno del segmento dato: in questo caso, come si può vedere dall'esplosione in assembler posta alla sinistra, l'offset della variabile TRE è 0003H e tale valore viene proprio associato come inizializzazione della variabile.

QUATTRO DW UNO è identico al caso precedente ed associa alla variabile QUATTRO, di tipo word, l'offset della variabile UNO e cioè 0000.

CINQUE DD CINQUE è praticamente simile al caso della definizione della variabile TRE, solo che in questo caso si ha a che fare con una direttiva DD invece che DW: la DD è un'estenzione della DW, nel senso che permette di assegnare ad una variabile non soltanto l'offset di una certa variabile, ma anche il segmento di appartenenza, come dire permette di associare ad una variabile un indirizzo completo. Dal momento che il segment di un certo indirizzo di tipo 86/88 è in un certo senso «più significativo» (più importante) dell'offset, ecco che i progettisti dell'Intel hanno posto la convenzione di allocare in memoria l'offset come «Least Significant Word» ed il segment come «Most Significant Word», come dire che leggendo locazioni successive di memoria troveremo dapprima il valore di un offset e poi il valore del segment.

Anche per ognuno di questi due valori (che ricordiamo essere posti in due byte) vale la consueta regola di porre prima la parte meno significativa (LSB) e poi la più significativa (MSB): in questo caso l'assemblatore però può sembrare fuorviante oppure in errore in quanto mostra, sempre nel lato sinistro dove c'è l'«esplosione» di un'istruzione, le word così come si leggono dal punto di vista «logico», ma non dal punto di vista dell'allocazione effettiva dei byte in memoria.

Infatti vediamo, come esplosione dell'istruzione in esame, i valori 0007 1000, che rappresentano, sì, l'offset ed il segment della variabile CINQUE, ma come detto sono da leggersi così come sono e cioè: l'offset di CINQUE è 0007 ed il segment è 1000; questo accade se usiamo, come nel nostro caso, il programma MASM.EXE che gira sotto ambiente MS-DOS su di un PC IBM (compatibile...). Viceversa usando il programma ASM86 originario dell'Intel, otterremmo come esplosione qualcosa come «07000010», che rappresenta l'effettiva allocazione in memoria dei byte.

Evidentemente l'output del programma MASM è più leggibile dal punto di vista logico, mentre l'ASM86 è corretto dal punto di vista fisico: tutto è farci l'abitudine e non confondersi nei due casi. In particolare segnaliamo il fatto che l'output del MASM contiene parecchi «blank» di separazione tra quantità «logiche», mentre l'output dell'ASM86 non contiene blank, rappresentando uno attaccato all'altro i byte costituenti una certa istruzione.

SEI DD QUATTRO è ancora un esempio identico al precedente, in cui alla variabile SEI viene associato l'indirizzo completo della variabile QUATTRO, composto, non dimentichiamocelo mai, dell'offset e del segment della variabile in esame. A conferma di quanto detto prima, l'ASM86 in questo caso mostrerebbe i quattro byte tutti attaccati «05000010».

Le istruzioni e le direttive

Prima di andare a formalizzare quanto detto finora con un certo numero di regolette, ritorniamo sul concetto di istruzioni e direttive. In particolare le istruzioni da un lato e le direttive dall'altro sono degli statement e cioè delle indicazioni fornite all'assemblatore per fargli compiere certe azioni: ecco che un programma viene inteso come insieme di statement e cioè di operazioni da far compiere al nostro assembler.

In particolare le istruzioni, come già detto, vengono tradotte dall'assemblatore direttamente in opcode, che l'86/88 eseguirà quando il programma oggetto verrà caricato in memoria; viceversa, le direttive non vengono tradotte in opcode, ma indicano all'assembler che deve compiere certe predeterminate operazioni.

Le istruzioni infine sono sia prefissate dall'assembler (corrispondono cioè ai codici mnemonici delle istruzioni in linguaggio macchina), sia definibili dal programmatore, mentre le direttive sono predefinite a livello assemblatore e non possono essere create dal programmatore, a meno di non riscrivere daccapo l'assemblatore stesso...

Con questo terminiamo anche questa puntata, dal momento che preferiamo sempre consentire al lettore una pausa di riflessione.

La prossima puntata sarà dedicata alla formalizzazione delle regole sintattiche alle quali abbiamo accennato in precedenza.

DISK-DRIVE
STAMPANTI
COMPUTER
MONITOR
LA CASA DEL COMPUTER
IMPORTAZIONE DIRETTA

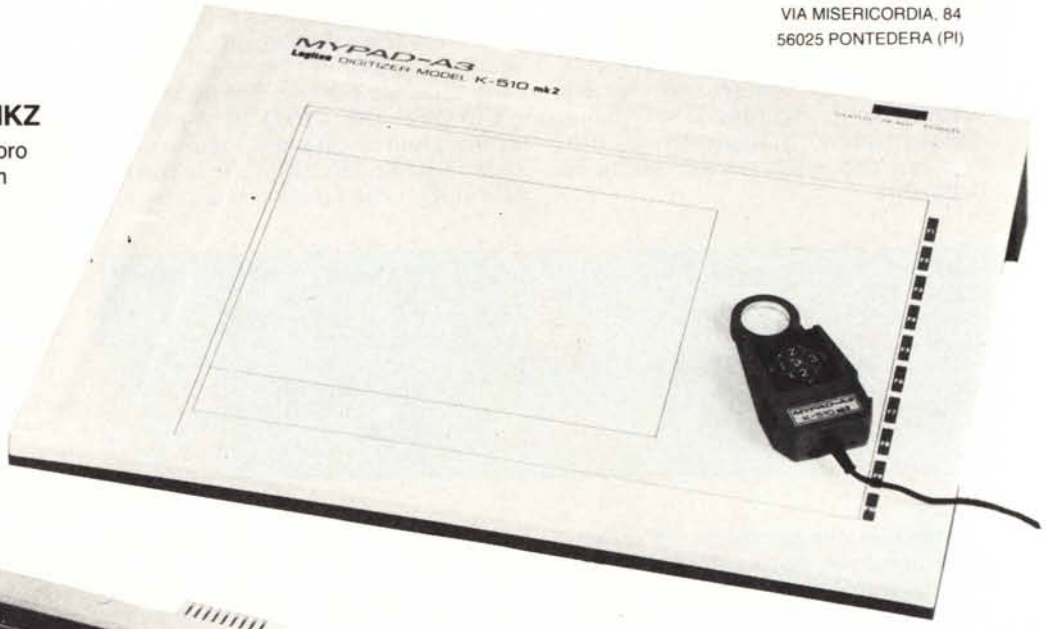
CAD-CAM FACILE??

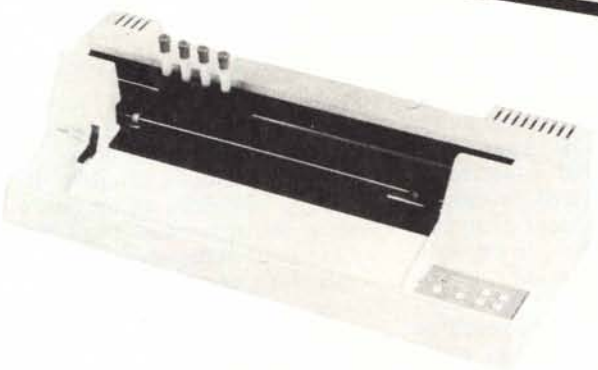
SOLO L'IMBARAZZO DELLA SCELTA!

VIA MISERICORDIA, 84
56025 PONTEDERA (PI)

DIGITIZER K-510 MKZ

Potente strumento di lavoro
con risoluzione di 0,1 mm
ed area digitalizzabile
formato A3
IDEALE PER
APPLICAZIONI
CON AUTOCAD.



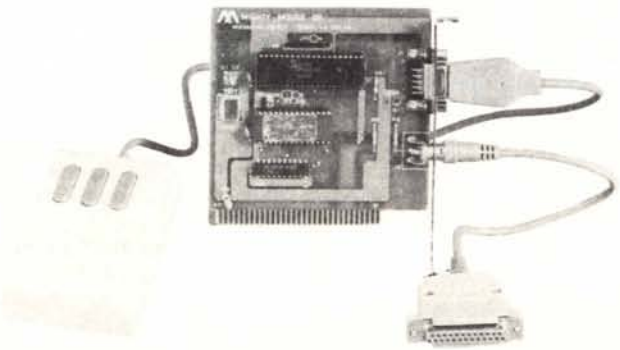


PLOTTER FPL-2000

Veloce e preciso plotter a 4 penne ed interfacie seriale e parallela. Precisione: 0,1 mm con area plottabile formato A3. Viene fornito con procedura X-ON/X-OFF per autocard (compatibile MP-7470).
Accessori optional: pennini ed adattatori per china, penne ed adattatori Ball Pentel R56, pennarelli ed adattatori Ceramicron.

MIGHTY MOUSE

Sistema meccanico mediante sfera gommata. Interfacciabile con RS-232 mediante cavetto e scheda forniti di serie.
Completa compatibilità con tutti i più importanti Pacchetti Software: Autocad, PC-Paint, D.Halo, ecc.



CRYSTAL MOUSE

Sistema a sensore Opto-Electronic (Encoder) di altissima precisione. Viene dotato della apposita tavoletta. Interfacciabile con RS-232. Compatibile con tutti i più importanti Pacchetti Software.

0587
212.312

CONTATTATECI OGGI STESSO PER MAGGIORI DETTAGLI E QUOTAZIONI

SIG.ri RIVENDITORI

PREZZI IVA ESCLUSA

DISK-DRIVE · STAMPANTI · COMPUTER · MONITOR

LA CASA DEL COMPUTER

IMPORTAZIONE DIRETTA

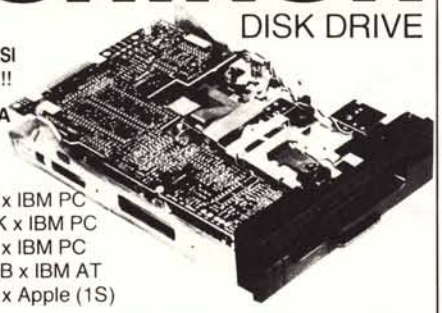
0587 212.312

VIA MISERICORDIA, 84 - 56025 PONTEDERA (PI)

CHINON DISK DRIVE

10 VOLTE PIÙ SILENZIOSI DEGLI ALTRI!!!

GARANZIA 1 ANNO



TIPI:
F-502 360K x IBM PC
F-502L 360K x IBM PC
F-561 1 MB x IBM PC
F-506 1,6 MB x IBM AT
F-051 180K x Apple (1S)

DISPONIBILI ORA I NUOVI MODELLI CON CHIAVETTA

PREZZI: DA LIT. 270.000

CHI VI DA UN ASSORTIMENTO COSÌ COMPLETO CON PREZZI SUPER COMPETITIVI???

Basta una telefonata ed in 48 ore riceverete quanto ordinato con garanzia 6 mesi od 1 anno e, se non sarete soddisfatti, vi sostituiremo l'articolo con lo stesso modello o con altro materiale a patto he il reso ci pervenga non manomesso, in porto franco, con gli imballi originali entro 18 gg. dalla data di spedizione

AT COMPATIBILE

Versione Base: Main Board OK espandibile d 1 M.B., alimentatore 200 W. Cabinet in metallo, tastiera **L. 2.600.000**

PC/XT TURBO

L. 1.475.000
Clock 8-4,77 MHz
Main Board Esp. 640K

NOVITÀ

N. 1 Drive DS/DD 360K, controller.
Main Board OK espandibile A 640K,
Alimentatore 130 W, Tastiera K5 S

PC/XT STANDARD (4,77 MHz)

L. 1.299.000
Configurazione come sopra ma con Main Board 256K espandibile a 640 K.

* * * Per le interfacce video vedere listino

Monitor Caègi Philips Monocr. x IBM	**L. 227.000**
Monitor Ciregi sonoro	**L. 148.000**
Monitor Philips HR Colori x IBM	**L. 690.000**

INTERFACCE PER APPLE

Controller Drive App.	60.000
16K Ram Card	83.000
80 Colonne Soft/SW.	108.000
8088 Card	592.000
Eprom Writer (16-64)	98.000
Prom Writer	434.000
Z/80 Card	59.000
RS-232 con cavo	100.000
Epson Printer e cavo	88.000
Grappler Pr. e cavo	98.000
AD-DA 12B./16 Canali	504.000
AD Card	177.000
AD-Da 8 Bit/19 Canali	336.000
IEEE-488 con cavo.	238.000
6809 Card	322.000
Communication Card	110.000
Super Serial Card	129.000
Pal Color Card	83.000
RGB Card (8 color).	124.000
RGB II (16 color)	194.000
Stereo Music Card	138.000
Scheda parlante	78.000
Wild Card	78.000
Scheda orologio	87.000
6522 Card	93.000
Forth Card	131.000
I.C. Test Card	198.000
80 Colonne + 64K IIE	54.000
80 Colonne x IIE	26.000
Adattatore Drive IIC	20.000
Adatt. Joystik IIC	14.000
Sch. orologio Prodos	120.000
Apple-IBM Conn. Card	590.000
512K Ram (ok) Esp. 1M	532.000
Esp. ulteriori 512K	240.000
Kit 8 Ram 4164 (64K)	34.800
Kit 8 Ram 256 (256K)	102.000

INTERFACCE PC/XT IBM

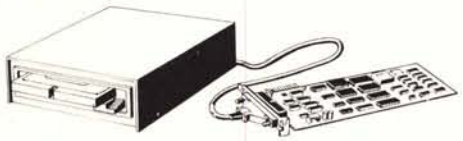
H.D. Controller 6210	330.000
Controller + cavo	120.000
Printer Card IBM	72.000
Color Graph. 2/L IBM	190.000
Mono/Col/Gr/Prin CR	340.000
Mono/Cr/Print Herc. 2	240.000
Multif. 256K Oran IBM	220.000
Multif. 384K Oram IBM	270.000
AD-DA Card IBM	435.000
Kit Ram 64K (9 Chip)	39.150
RS-232 Card IBM	108.000
Game I/O Card IBM	72.000
I/O Plus Card IBM	200.000
Eprom Writer 16/128	345.000
8255 Card IBM	270.000
IEEE-488 Card IBM.	570.000
Espansione 384K Ok	148.000
Espansione 512K (Ok)	138.000
Rete loc. I-Net + cavo	980.000
Rete loc. RPTI TR/Net	1.320.000
8087 Coprocessore PC	390.000
Mon/Col/Gr/Pr Amdek	490.000
Mono/Col/Gr Alta Ris	400.000
E.G.A. Color/Gr H.R.	980.000

INTERFACCE AT IBM

AT Controller X 2FDD	278.000
AT Parall/Serial C.	224.000
AT Multi Serial (4S).	392.000
At Espans. 2,5 MB Ok	376.000
AT Espans. 3,5 MB Ok	520.000
AT Multifunc. 2,5 MB	490.000
AT Multifunc. 3,5 MB	590.000
Kit Ram 256K	114.750
Controller HDD + 2FDD	1.024.000

STREAMER 20 M.B.

TEAC MT-25T - Sofisticato sistema corredato di interfaccia e soft di gestione. Da collocarsi internamente al PC/XT/AT. La copia di 23 MB viene eseguita in 9 minuti circa su cassette tipo «COMPACT» da 500/600 FT.



SUPER 5 - Versatile unità di back-up per PC/XT/AT corredato di interfaccia e soft di gestione. Di semplice e veloce uso in quanto provvede ad eseguire la copia di 20 MB in soli 5 minuti. Usa cassette da 600 FT tipo «COMPACT». È dotato di cabinet metallico e cavo di collegamento all'interfaccia. Consigliato per installazioni esterne al sistema.

HARD DISK

Delle migliori marche come i nuovissimi Epson con ricovero automatico delle testine nella «Shipping zone» al momento dello spegnimento del sistema.

Epson HD-8140 10 MB senza/contr.	**L. 1.090.000**
Seagate ST-225 20 MB senza/contr.	**L. 1.190.000**
Seagate ST-4051 40 MB senza/contr.	**L. 2.430.000**

DATAFLEX

PROFESSIONAL 5¼

100% CERTIFICATI ERROR FREE

CONFEZIONE BULK 250/PZ. **L. 1.760** DOPPIA/DOPPIA

CON BOX IN PLASTICA OMAGGIO!!!
SCONTI PER QUANTITÀ

SINGOLA F. - DOPPIA D.		DOPPIA F. - DOPPIA D.	
200 Pezzi	**L. 1990**	200 Pezzi	**L. 2650**
100 Pezzi	**L. 2100**	100 Pezzi	**L. 2800**
30 Pezzi	**L. 2350**	30 Pezzi	**L. 3150**
ALTA DENSITÀ PER AT. **L. 7.800**			

- I dischetti dataflex sono prodotti da uno dei più grossi fabbricanti americani che garantisce l'altissima qualità ed affidabilità.
- Uno speciale ed esclusivo strato «Multicot» protegge la superficie dall'usura del contatto con le testine garantendo minimo ben
- 10.000.000 di passaggi!!!
  La sicurezza dei Vs. dati è assicurata dall'ineccepibile supporto magnetico di primissima qualità.

DATO L'INSTABILE MERCATO DEI CAMBI PREGASI TELEFONARE PER CONFERMA PREZZI E DISPONIBILITÀ — RICHIEDETECI IL CATALOGO — SCONTI AI SIG. RIVENDITORI