ASSEMBLER 8086 8088

di Pierluigi Panunzi

Istruzioni e direttive

In questo numero parleremo delle regole sintattiche secondo le quali devono essere scritte le istruzioni e le direttive di un programma Assembler.

Abbiamo già detto che le «istruzioni» di un programma in Assembler corrispondono alle istruzioni in codice macchina eseguite dal micro-processore, mentre viceversa le «direttive» non sono altro che delle indicazioni, dei comandi impartiti all'assemblatore, per ottenere certe predeterminate prestazioni ed operazioni particolari.

La sintassi delle istruzioni

Sappiamo già da altri linguaggi specie ad alto livello, che un programma è formato da una sequenza di istruzioni che il micro-processore va ad eseguire una dopo l'altra a meno che non ne incontri una particolare, un «salato»; dal momento che oltre alle istruzioni abbiamo a disposizione le direttive, ecco che un programma in Assembler per 8086/88 sarà costituito da un'opportuna serie di istruzioni e di direttive, che vengono definite con il nome generico di «statement».

Si può così dire che un programma è costituito da una sequenza di «statement», posti in generale (per motivi di leggibilità) ognuno su di una linea di stampa, che risulta separata dalle altre da un cosiddetto «carattere terminatore» (di solito il «carriage return» oppure il «line feed» oppure la coppia «carriage return/line feed»). Dal momento che uno di questi caratteri terminatori ha l'effetto di mandare a capo il cursore nello schermo video oppure il carrello di stampa nella stampante, ecco che così si ottengono statement posti su linee differenti.

Tuttavia l'Assembler prevede anche il caso in cui si desidera allargare uno statement a più di una riga dello schermo o di stampa: ciò si ottiene ponendo come primo carattere della linea «continuazione» della linea precedente il carattere «&», avendo però l'accortezza di non troncare i simboli costituenti la linea, ma di passare a linea successiva in genere in corrispondenza al «blank» di separazione tra due parole oppure in corrispondenza di «virgole».

Veniamo dunque alla sintassi di un'istruzione: sappiamo che con tale termine si intendono le regole da seguire nello scrivere una generica istruzione.

La sintassi prevede per una qualsiasi istruzione Assembler un unico «formato» da seguire, costituito da un insieme di cinque «campi», che sono i seguenti:

LABEL PREFIX OPCODE OPERAND(S) COMMENT

In una generica istruzione potranno dunque comparire da uno a cinque capi differenti, secondo delle semplici regolette, del tutto simili a quelle che si trovano in Assembler di altri micro-processori.

Analizziamo dunque uno per uno i campi.

Il campo «label»

Il campo «label» o campo «etichetta» è formato da un simbolo seguito dai «due punti» («:») e definisce appunto una Label posta all'indirizzo dato dal valore attuale dell'«Instruction Pointer» (IP) ed all'interno del "segment" corrente.

È un campo che può anche mancare (ad esempio nella maggioranza delle istruzioni che devono essere eseguite in sequenza) e può essere usato opzionalmente laddove le esigenze di programmazione lo richiedano (ad esempio l'entry point di una routine o il punto di arrivo di un salto, condizionato o meno).

Il campo «prefix»

Il campo «prefix» è un campo particolare sfruttato solo in caso si usino delle istruzioni speciali ed in genere si riferisce appunto ad un prefisso dell'istruzione speciale, prefisso che ne modifica il comportamento.

Come per il campo precedente così anche in questo caso il campo è opzionale ed infatti verrà usato solo per particolari istruzioni di gestione di coprocessori e di arbitrariaggio del bus, nonché per ottenere la ripetizione di istruzioni elementari (primitive) su stringhe di caratteri.

Il campo «opcode»

Il campo «opcode» è quello più importante in quanto in esso compariranno tutte le istruzioni Assembler, costituite da un simbolo compreso tra quelli previsti per le istruzioni dall'Assembler, corrispondente mnemonico di una ben determinata istruzione in linguaggio macchina (esadecimale) eseguibile dal micro-processore.

Anche questo campo può essere opzionale e cioè può mancare, ma in questo caso non può esistere il capo «operand(s)» di cui parleremo tra breve.

L'insieme dei vari vocaboli mnemonici riconosciuti dall'assemblatore è quello che viene indicato molto spesso come «set di istruzioni» del micro-processore stesso.

Il campo «operand(s)»

Il campo «operand(s)» è un campo

strettamente legato al precedente in quanto sarà presente laddove l'istruzione preveda la presenza di uno o due operandi (ecco il perché della «s» tra parentesi) sui quali l'operazione verrà effettuata.

Come regola generale,se è presente più di un operando, allora il o i successivi devono essere separati da una virgola («,»).

Come vedremo molto più avanti, quando parleremo delle già citate «CODEMACRO», sarà possibile, su scelta del programmatore, inserire anche più di due opernadi «a valle» di una certa istruzione.

Il campo «comment»

Il campo «comment» è un campo molto utile per conferire al nostro programma un certo grado di leggibilità e di documentazione e deve essere iniziato da un carattere «punto e virgola» («;»): tutto quello che appare nella linea di programma tra il «;» ed il carattere terminatore verrà considerato come commento e perciò del tutto escluso dall'analisi dell'assemblatore.

Più che un campo obbligatorio o viceversa opzionale, è invece un campo vivamente utile e consigliato in quanto consente una migliore comprensione del programma sia da parte di altri programmatori, sia del «creatore» del programma specie a distanza di tempo dalla «creazione».

Alcuni esempi

Vediamo ora alcuni esempi di istruzioni (sul cui significato intrinseco torneremo a tempo debito), nei quali compariranno a volte alcuni ed a volte altri dei cinque campi previsti.

```
1) ETICHETTA: SUM BX ;
   somma di due registri
2) ; programma scritto il 23/9/1985
3) CALL PIPPO
4) REP STOSW
5) CLI
```

Iniziamo dall'esempio numero 1: in questo caso abbiamo il campo «label» dal momento che troviamo un simbolo seguito dai due punti (ETICHETTA), il campo «opcode» in quanto SUM è un codice mnemonico di un'istruzione di somma, il campo «operand(s)» in quanto l'istruzione SUM prevede due operandi separati dalla virgola ed infine il campo «comment» che ci permette di spiegare il significato dell'istruzione con parole precedute dal punto e virgola.

Il secondo esempio prevede solo la presenza di un commento ed è utile per dare delle informazioni generali, o anche per separare parti logiche del nostro programma: un comune esempio di commento separatore è quello formato da una linea di commento vuota, una linea di commento fatta di asterischi ed un'altra linea di commento vuota:

```
;
;**************************
;
```

Il terzo esempio prevede l'esistenza del campo «opcode» rappresentato dall'istruzione CALL, seguito dal campo «operand(s)», costituito dal simbolo PIPPO (poteva mancare?!), presumibilmente l'etichetta di una certa routine oppure, come vedremo, il contenuto di una locazione di memoria (CALL indiretta): mancano in questo caso gli altri tre campi ed in generale è questa la forma di un'istruzione generale. Il quarto esempio è particolare in quanto prevede il campo «prefics» ed il campo «opcode»: che il simbolo REP sia un prefisso e non un'etichetta si capisce innanzitutto dal fatto che mancano i due punti di identificazione e poi dal fatto che REP è un'istruzione, sulla quale ritorneremo in dettaglio insieme alla descrizione delle istruzioni di gestione delle stringhe quali appunto la STOSW.

L'ultimo esempio invece comprende un unico campo, quello degli «opcode», dal momento che l'istruzione CLI non prevede operandi né prefissi e dato che non abbiamo voluto inserire un commento e le esigenze di programmazione non hanno richiesto la presenza di un'etichetta.

La sintassi delle direttive

Nel caso delle direttive, una linea di programma può essere formata al massimo da quattro campi e perciò avrà l'aspetto seguente:

```
NAME DIRECTIVE OPERAND(S) COMMENT
```

Analizziamo ora uno per uno i campi, ma già diciamo da adesso che il significato di alcuni concetti apparirà più chiaro nel seguito quando torneremo in dettaglio sugli argomenti specifici.

Il campo «name»

Il campo «name» è vagamente simile al campo «label», vagamente nel senso che non rappresenta in realtà l'etichetta della direttiva, ma ne rappresenta una parte integrante: quando analizzeremo in dettaglio le varie direttive apparirà più chiara quest'ultima affermazione.

Comunque altro indizio che non si tratta di un campo rappresentante un'etichetta è dato dal fatto che il «name» non deve mai essere seguito dai «due punti», come una normale label: in caso contrario l'assemblatore segnalerà un perentorio messaggio d'errore.

Il campo «directive»

Questa campo è tutto sommato simile al campo «opcode» delle istruzioni: nel campo «directive» infatti troverà posto una delle tante direttive previste dall'Assembler, ognuna delle quali riferita ad una particolare operazione logica da effettuarsi all'atto dell'assemblaggio («assembly-time»), fra tutte le quali la ben nota definizione delle zone di memoria e dei segmenti.

Il campo «operand(s)»

Tale campo è analogo a quello visto nel caso delle istruzioni ed è in un certo senso più generalizzato: nel campo «operand(s)» infatti troveranno posto uno o più operandi, a seconda delle esigenze della direttiva oppure una particolare parola chiave o espressione che in un certo senso non sono dei veri e propri «operandi», ma più degli «attributi» della direttiva in esame.

Il campo «comment»

Quest'ultimo campo è perfettamente analogo a quello delle istruzioni e, ci sia consentito il gioco di parole, oramai «si commenta da solo»

Esempi di direttive

Nel proporre alcuni esempi di direttive ci atterremo a quelle direttive che già abbiamo conosciuto:

```
1) PROGR SEGMENT
2) VALUE EQU 1000
3) ALFA DB 1,2,3,4
```

Nel primo esempio abbiamo riportato la direttiva che definisce un «segment», chiamato in questo caso PROGR: ribadiamo il concetto che «PROGR» in questo caso non è l'etichetta della direttiva SEGMENT, ma è proprio il nome assegnato al segmento.

Il secondo esempio, altrettanto ben noto, è quello che consente di associare al simbolo «VALUE» un valore ben definito, che potrà essere utilizzato nel corso del programma tutte le volte che altrimenti si dovrebbe usare il valore 1000.

L'ultimo esempio è ancora una volta ben noto e si riferisce alla creazione di quattro celle di memoria (4 byte), la prima delle quali chiamata ALFA (che in questo caso è un'etichetta...) e ad ognuna delle quali abbiamo assegnato un valore iniziale.

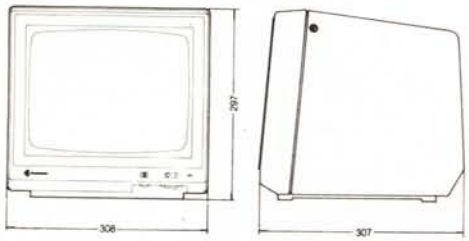
Con questo abbiamo terminato: la prossima puntata andremo a conoscere in dettaglio tutte le direttive dell'assemblatore, formalizzando secondo alcune semplici regolette quando già conosciamo. MC

CRYSTAL

MONITORS

CASELLA POSTALE 142
56025 PONTEDERA (PI)
VIA MISERICORDIA, 84
TEL. 0587 - 212.312

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

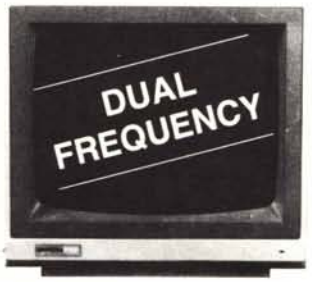


La differenza c'è!! E si vede.
Certo!, non tutti i monitor 12'' sono uguali e per questo ti chiediamo di fare una prova confrontando la risoluzione al centro ed ai bordi di un CRYSTAL contro un altro qualsiasi monitor TTL per IBM PC.

FOSFORI VERDI (P39)
Freq. orizzontale 18,4 KHz
Banda passante 16 MHz
Ingresso: TTL
Risoluzione: 1000 linee
Uso: per schede IBM tipo Hercules

FOSFORI AMBRA (PLA)
Freq. orizzontale 18 KHz
Banda passante 16 MHz
Ingresso: TTL
Risoluzione 1000 linee
Uso: per schede IBM tipo Hercules

FOSFORI BIANCHI (WD)
Freq. orizzontale 18,4 KHz
Banda passante 16 MHz
Ingresso: TTL
Risoluzione: 1000 linee
Uso: per schede IBM tipo Hercules



FOSFORI P42
Doppia frequenza orizzontale 15,7-18,4 KHz
Ingressi: TTL e Composito
Risoluzione: 800 linee
Uso: per schede tipo Hercules e Color, per Apple computer ecc.

SUPER E.G.A. CARD

Questa scheda è frutto della CMOS-VLSI ed ha una tecnologia di larga applicazione in quanto, oltre ad emulare la Enhanced Graphics Adapter, emula anche la Color Graphics Adapter e la Hercules.

MONITOR

VENITE A TROVARCI ALLO
+X= smau
MILANO 17-22 SETTEMBRE
PAD.15, SAL.2, STAND F6-G1

Monitor ad alta risoluzione (DOT PITCH: 0,31 mm) consigliato con IBM PC/XT/AT per l'uso della ENHANCED GRAPHICS ADAPTER o della COLOR GRAPHICS ADAPTER.
Scanning automatico della frequenza orizzontale (15,75-21,85 KHz) per entrambi i modi grafici 640 x 350 e 640 x 200.
Nel modo 640 x 350 possono essere usati più di 64 colori per la grafica (16 alla volta).
Smagnetizzazione del tubo automatica, ventilatore interno.
Possibilità di usare su tutto lo schermo solo i colori verde ed arancio (particolarmente richiesto per elaborazione di testi).

Graphics Modes:

N° dei colori	Pixels (H x V)	Pagine (max)	Tipo di monitor*	Adattatori emulati**
16	320 x 200	8	CD/ECD	EGA
16	640 x 200	4	CD/ECD	EGA
2	640 x 350	1	MD	EGA
2	640 x 350	2	MD	EGA
4/64	640 x 350	1	ECD	EGA
16/64	640 x 350	2	ECD	EGA
4	320 x 200	1	CD/ECD	CGA, EGA
2	320 x 200	1	CD/ECD	CGA, EGA
2	640 x 200	1	CD/ECD	CGA, EGA
2	720 x 348	2	MD	HGC

Alphanumeric Modes:

N° dei colori	Caratteri x linee	Pagine (max)	Tipo di monitor*	Adattatori emulati**
16/64	80 x 25	2	ECD	EGA
16/64	80 x 43	8	ECD	EGA
2	80 x 43	4	MD	MDA/EGA
2	40 x 25	8	CD/ECD	CGA, EGA
16	40 x 25	8	CD/ECD	CGA, EGA
2	80 x 25	8	CD/ECD	CGA, EGA
16	80 x 25	8	CD/ECD	CGA, EGA
2	80 x 25	8	MD	MDA,HGC,EGA

* ECD = Enhanced Color Display, CD = Color Display, MD = Monochrome Display.
** EGA = Enhanced Graphics Adapter, CGA = Color Graphics Adapter MDA = Monochrome Display Adapter, HGC = Hercules Graphics Card

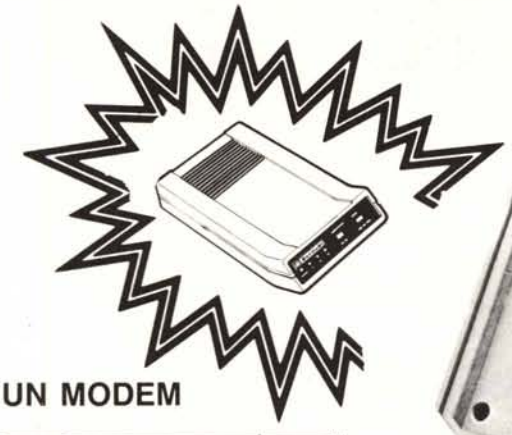
Bondwell™

DISK-DRIVE
STAMPANTI
COMPUTER
MONITOR

LA CASA DEL COMPUTER

IMPORTAZIONE DIRETTA

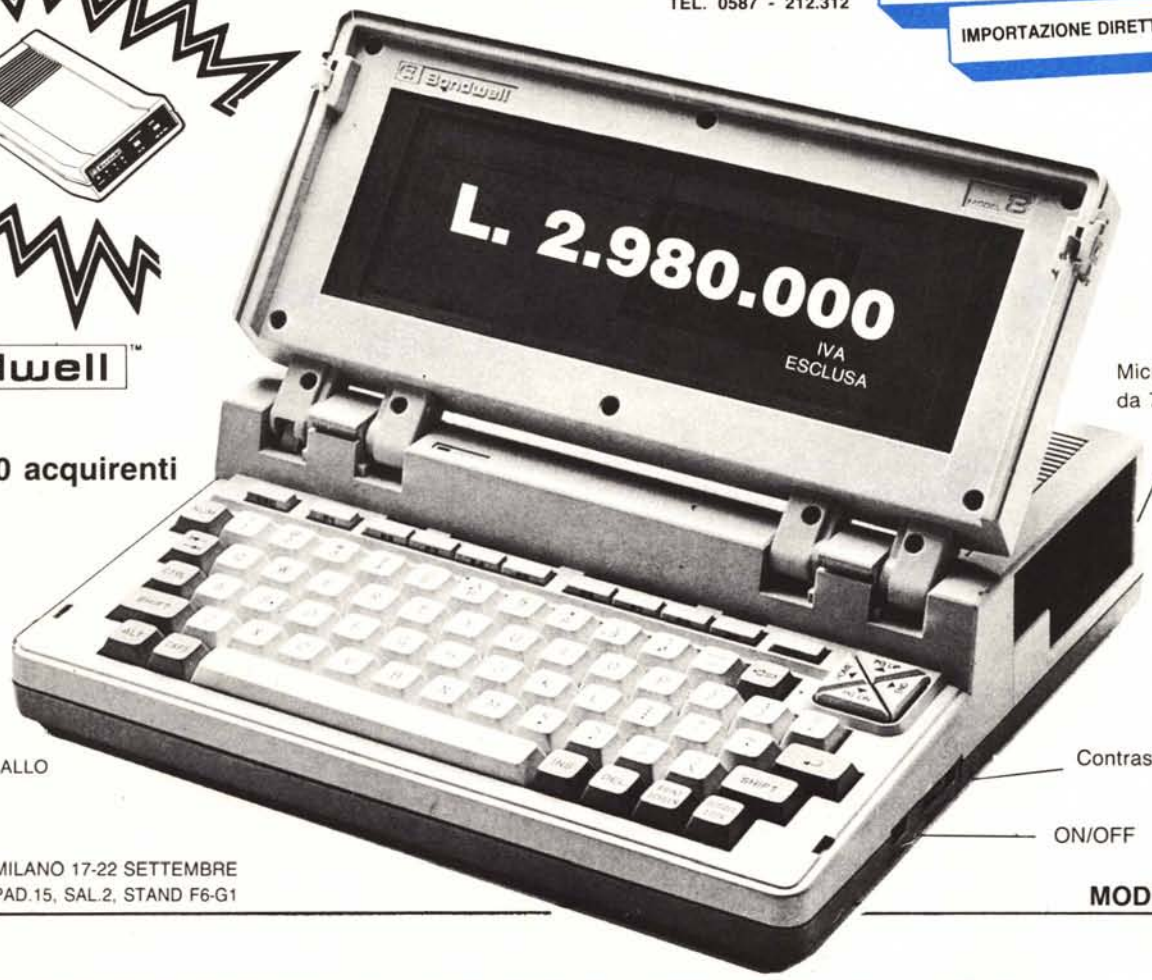
CASELLA POSTALE 142
56025 PONTEDERA (PI)
VIA MISERICORDIA, 84
TEL. 0587 - 212.312



UN MODEM

Bondwell™

IN REGALO
ai primi 1.000 acquirenti



VENITE A TROVARCI ALLO

smau

MILANO 17-22 SETTEMBRE
PAD.15, SAL.2, STAND F6-G1

MODEL 8

IL PIÙ PICCOLO E POTENTE PC PORTABILE SI CHIAMA Bondwell™ 8

- Facilmente trasportabile
- Peso: Kg. 4,500
- Dimensioni: cm. 28 × 31 × 78
- IBM-PC compatibile (DOS 2.11 su licenza Microsoft)
- Dischetto con MS/DOS 2.11, GW Basic 2.0 e manuali inclusi
- Basso consumo ottenuto con l'impiego di componenti CMOS
- Microprocessore: 80C88, 4.77 MHz
- Memoria RAM: 512K
- Schermo a cristalli liquidi ad alto contrasto, illuminabile, e con risoluzione 640 × 200 (grafica), 80 × 25 (testo)
- Floppy disk interno da 3'' 1/2 doppia faccia/doppia densità da 720K formattati
- Orologio/Calendario mantenuto da batterie al nichel-cadmio ricaricabili
- Batterie ricaricabili 12V-3A
- Tastiera con 76 chiavi e basso profilo, compatibile con lo standard PC/XT, dotata di funzioni del PAD numerico, 10 tasti funzione ecc. ecc.
- Porta seriale standard R-232C
- Porta parallela per stampanti
- Porta per la connessione del 2° Drive (5'' 1/4 oppure 3'' 1/2)
- Uscite per video RGB/TTL e video-composito
- Led segnalatore intermittente di fine carica
- Alimentatore/Caricabatterie AC/DC
- Hard e Soft realizzati per ottenere il massimo della compatibilità IBM-PC. Possono essere eseguiti i più popolari pacchetti software come: Lotus 1-2-3, Symphony, D Base II e III, Wordstar, Flight Simulator, Frame work, Jem, Sidekick, PFS serie, ...