di Pierluigi Panunzi

I modi di indirizzamento

Nelle prime due puntate di questa rubrica, abbiamo iniziato a conoscere il microprocessore 80286 dell'Intel, che come ben sappiamo è usato nei modelli IBM AT e compatibili, nonché in alcuni nuovi modelli della serie PS/2, sempre dell'IBM, mentre in questa puntata entreremo un po' più nei dettagli, diciamo così, operativi, parlando dei modi di indirizzamento e dei tipi di dati caratteristici di questo microprocessore.
Si tratta in particolare di caratteristiche che prescindono dallo stato in cui si trova il micro e cioè se in «Real Mode» oppure in «Protected Mode» e che in questo caso riguardano più da vicino la programmazione del componente

Analogamente a quanto accadeva nel caso del microprocessore 8086 (e 8088, ovviamente), un'istruzione generica può o meno contenere uno o due operandi, i quali, se presenti in coppia, in generale possono far riferimento ad una coppia di registri, ad un registro o memoria con un valore immediato, ad un registro ed una locazione di memoria, mentre è generalmente illecito avere come operandi direttamente due locazioni di memoria.

Mentre nel caso di operandi relativi a registri ed a valori immediati non c'è nulla da aggiungere a quanto già sappiamo (non è il caso di ricordare ancora una volta i registri interni, che sono sempre gli stessi), viceversa il caso delle locazioni di memoria richiede semmai un piccolo «refresh» in quanto solo conoscendo alla perfezione tali meccanismi si può avere una completa padronanza della memoria stessa, privilegi permettendo...

In particolare abbiamo a disposizione sei tipi di indirizzamento, che andiamo ad analizzare singolarmente:
1) L'«indirizzamento diretto» consiste nel fatto che nell'istruzione in esame, in corrispondenza dell'operando cui si riferisce tale indirizzamento, appare un valore a sedici bit che rappresenta l'offset di una locazione di memoria, che potrà essere così direttamente raggiunta a partire dal segmento che in generale o è implicito, oppure è esplicitamente citato nell'istruzione stessa (come «override»).

Ricordiamo a tal proposito, per inciso, che riferimenti a locazioni di memoria del Data Segment fanno implicitamente capo al registro DS, mentre riferimenti al Code Segment fanno capo al CS e viceversa riferimenti allo Stack Segment ed al registro base BP fanno capo implicitamente al registro SS.

In ogni caso, salvo rarissime eccezioni, questo infernale meccanismo automatico può essere completamente scavalcato («override») a nostro piacimento indicando nell'istruzione stessa il registro di segmento desiderato.
2) L'«indiretto attraverso un registro» invece sfrutta un livello di «indirettezza» dato dal contenuto di un registro, che contiene dunque lui l'offset della locazione di memoria che vogliamo individuare: anche in questo caso gioca pesantemente il suo ruolo il meccanismo visto, che associa ad ogni tipo di istruzione con riferimento alla memoria uno dei registri di segmento, con la solita possibilità di «override». Dal momento che questo vale per tutti i tipi di indirizzamento che verranno, non lo diremo più...
3) L'«indirizzamento basato con un registro base» prevede l'uso di due particolari registri «base» BX e BP (quest'ultimo detto appunto «Base Pointer») i quali contengono l'offset della locazione di memoria appartenente rispettivamente al Data Segment ed allo Stack Segment.
4) L'«indirizzamento indicizzato con un registro indice» si ha allorché si usa uno dei due registri «indice» SI e DI, in generale contenenti o l'offset della locazione da indirizzare oppure un «displacement» all'interno di un vettore in memoria.

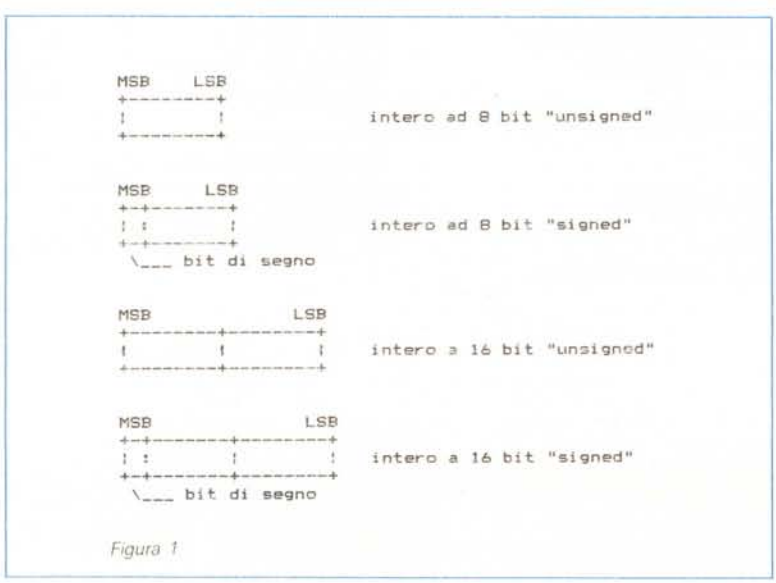


Figura 1

Figura 2

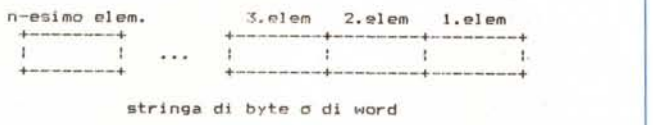


Figura 3

5) L'«indirizzamento basato-indicizzato» opera (come è facile prevedere dal suo nome) per mezzo di un registro «base» e di un registro «indice» dove stavolta la somma dei contenuti dei quali darà l'offset della locazione di memoria desiderata.
6) L'«indirizzamento basato-indicizzato e con displacement» infine è ovviamente analogo al precedente, ma in più consente l'aggiunta di un valore immediato, che può rappresentare sia l'offset iniziale di un vettore o una matrice, oppure un valore fisso in genere uno spostamento all'interno della struttura di dati: è questo il tipo di indirizzamento più complesso — ma poi mica tanto... —, che consente una facile gestione di strutture di dati quali matrici e vettori, senza troppe complicazioni di programma.

Tra parentesi diciamo che nell'80386 sono possibili ulteriori tipi di indirizzamento, che in un certo senso migliorano e completano quelli già visti e che ancor di più sono orientati all'implementazione di strutture di dati complesse dei linguaggi ad alto livello.

I tipi di dati

Nel caso del microprocessore 80286, analogamente a quanto avveniva già nell'8086, si hanno a disposizione parecchi tipi di dati «primitivi» sui quali poter operare: per completezza in questa sede parleremo anche dei dati che vengono utilizzati dal coprocessore matematico 80287, che molti possessori di AT (o compatibili o ecc., ecc.) hanno acquistato per il proprio personal computer.

In particolare (a differenza di quanto avverrà per l'80386, e non diciamo altro per creare una certa suspense...) tutti i tipi di dati su cui si opera non sono altro che opportuni multipli della quantità base data dal byte: cominciamo dunque la carrellata.

Abbiamo a disposizione i seguenti tipi di dati:

— *quantità intere con e senza segno*
— *puntatori*
— *stringhe di caratteri alfanumerici in particolare e di caratteri ASCII in generale*
— *numeri rappresentati in BCD esteso o BCD impaccato («packed BCD»)*
— *numeri espressi in «floating point»*

Analizziamoli uno per uno

— I numeri interi con e senza segno possono essere espressi con quantità ad 8 o a 16 bit, il cui rispettivo bit più significativo rappresenta il bit di segno, come è universalmente adottato e nel caso che si voglia esprimere la quantità in complemento a 2: c'è da dire al solito che la differenza tra una quantità espressa con il segno oppure «unsigned» è solo apparente in quanto dipende da come noi vogliamo interpretare la quantità.

Ad esempio un byte pari ad FFH vale sia «−1» se considerato in complemento a 2, sia «255» se in logica senza segno: operazioni di addizione e sottrazione considereranno queste quantità allo stesso modo, mentre viceversa abbiamo (come per l'8086) due tipi di moltiplicazioni e di divisione rispettivamente tra quantità dotate di segno oppure «unsigned». In questo caso possiamo rappresentare le quantità intere con o senza segno ad 8 o a 16 bit con lo schema di figura 1.

— I puntatori sono quantità a 32 bit, formate perciò da due word, rappresentanti, la word più significativa, il cosiddetto «selector» (che come già sappiamo è una generalizzazione del «segment register») e, la word meno significativa, l'offset di una locazione di memoria: si può in genere trattare di una coppia di registri (il primo un «segment register» ed il secondo un qualsiasi registro della CPU) oppure una coppia di locazioni consecutive di memoria, considerate singolarmente come word (ad esempio nel caso di «Intersegment jump» dove l'indirizzo di un altro segmento a cui saltare è espresso completamente come contenuto di una coppia di word in memoria, il tutto, lo ricordiamo e lo ricorderemo tutte le volte che ci capiterà, sempre se se ne ha il privilegio, nel caso di «Protected Mode»).

Schematicamente abbiamo una situazione del genere, in cui ovviamente in nessuna delle due word ha senso parlare di bit di segno (vedi figura 2).

— Le stringhe possono essere, come visto, formate da caratteri alfanumerici in particolare e di caratteri ASCII in generale e saranno per lo più preposte alla memorizzazione di messaggi, intestazioni, ecc., come pure, nel caso più generale, byte o word aventi un qualsiasi valore: non si tratta anche in questo caso di un tipo di dati particolare,

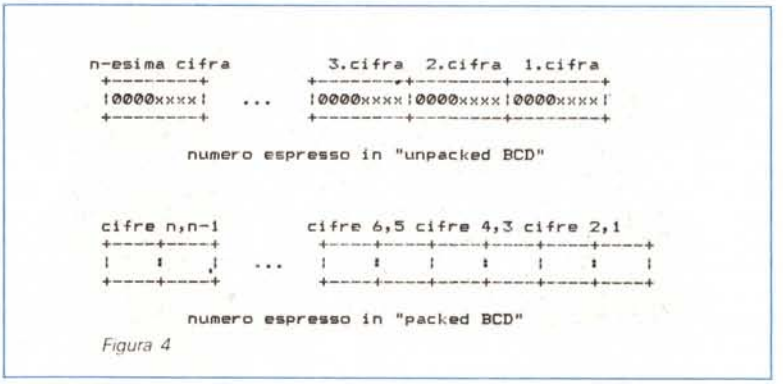


Figura 4

Figura 5

ma, come visto, un semplice susseguirsi di valori e/o codici alfanumerici, sui quali però sono possibili operazioni particolari, appunto quelle «di stringa», comprendenti, come nel caso dell'8086, operazioni «primitive» quali lo spostamento in memoria, la scansione, l'inizializzazione e la comparazione e che come sappiamo possono essere semplicemente ripetute per un certo numero di volte o sotto opportune condizioni.

Sappiamo inoltre che la lunghezza di una stringa può variare da 1 a 64k byte (mica è un linguaggio ad alto livello! Stiamo lavorando in Assembler ed un dato può occupare anche tutti i 64k byte di un segmento...), non essendo limitato se non appunto dalla dimensione di un segmento.

Abbiamo detto che il dato «elementare» che forma una stringa può essere un byte o una word; avremmo perciò due differenti rappresentazioni schematiche, che però riuniamo in una sola, laddove il «rettangolino» può essere sia un byte che una word (vedi figura 3).

— Per quanto riguarda i numeri rappresentati in BCD (esteso oppure «packed»), ci sono da fare alcune considerazioni, che tutto sommato valevano anche per l'8086, ma che non sono state poste in risalto nell'apposita rubrica.

Comunque, iniziamo l'analisi dalle quantità numeriche espresse in «unpacked BCD» (diciamo così, «BCD spacchettat»): sono rappresentate in memoria come una sequenza di byte senza segno, ognuno dei quali contiene un valore esadecimale compreso tra 0 a 9, come dire che ogni cifra del numero desiderato, espresso in decimale, viene memorizzata in un singolo byte.

Va da sé che le operazioni che si effettueranno su queste quantità devono essere effettuate cifra per cifra singolarmente: mentre le addizioni e le sottrazioni (che sono quelle standard...) non considerano il nibble alto dei byte su cui operano, nel caso della moltiplicazione e della divisione tale nibble deve essere necessariamente nullo.

In particolare nell'effettuare queste operazioni si otterrà in generale un risultato intermedio al quale si deve applicare una delle istruzione «di adjust» (tipo AAA, AAM, ecc.), in modo da ottenere in ogni caso un risultato ovviamente corretto.

Inoltre c'è da dire che le quattro operazioni (ed in particolare la moltiplicazione e la divisione) sono considerate del tipo «unsigned», in quanto il segno di una quantità espressa in BCD deve essere posto in un byte a parte.

Per quanto riguarda i «packed BCD», si ha che invece di porre una cifra per byte (sprecando inutilmente un nibble), se ne mettono due, la più significativa ovviamente nel nibble più significativo.

Bisogna badare che il range ammesso per ogni singolo nibble è sempre da 0 a 9 e perciò con un byte (due nibble...) si possono esprimere valori compresi tra 0 e 99.

Purtroppo però l'addizione e la sottrazione sono consentite (e sono sempre le stesse... non cambia niente!), a parte una correzione del risultato intermedio per ottenere un risultato finale corretto, manca totalmente la possibilità di effettuare la moltiplicazione e la divisione tra due «packed», essendo dunque uno stimolante problema di programmazione la ricerca di un opportuno algoritmo sostitutivo.

Analogamente ai casi precedenti, diamo in figura 4 una rappresentazione schematica di quanto detto finora.

— Infine per quantità reali e cioè espresse in virgola mobile, abbiamo (come detto solo con l'80287) la rappresentazione in «floating point», che consente di esprimere un enorme range di quantità numeriche e soprattutto di effettuare calcoli su di esse senza grossi problemi.

Esistono per grandi linee tre tipi differenti di formati in «floating point» (tutti rispondenti alle norme IEEE), in cui il «formato base» è il medesimo, mentre quello che cambia è il numero di byte (o meglio bit) usati: abbiamo perciò la possibilità di rappresentare quantità in virgola mobile con 4, 8 oppure 10 byte e perciò rispettivamente avendo a disposizione 32, 64 o la bellezza di 80 bit.

In generale un numero in tale formato presenta un bit di segno (è il bit più significativo del byte più significativo dei 4, 8 o 10 byte), un certo numero di bit di «esponente» ed i rimanenti come bit significativi.

Senza scendere troppo nei particolari, vediamo ad esempio il tipo di formato «più eclatante», che è formato da ben 10 byte, per un totale di 80 bit: di questi, il primo è il segno, 23 rappresentano l'esponente (e con ciò arriviamo a 3 byte), mentre i restanti 56 bit (gli ultimi 7 byte) sono tutte cifre significative (ovviamente zeri ed uni...).

Possiamo rappresentare graficamente in figura 5 questa situazione, ma ora il singolo «rettangolino» rappresenta un byte.

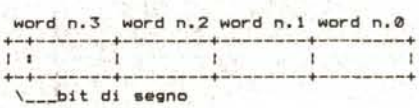
Visto che ci siamo accenniamo ad altri due tipi di quantità che possono essere gestite in presenza del coprocessore matematico 80287:

— quantità espresse come «signed double-word», sono evidentemente un'estensione delle «signed word» e rappresentano quantità espresse su 4 byte (una double-word) ed in più dotate di segno, posto al solito nel bit più significativo della word più significativa.

Eccone dunque la rappresentazione grafica:

— L'ultimo tipo di formato che analizziamo è la cosiddetta «signed quad-word», che è un'ulteriore estensione del caso precedente, relativo a quantità dotate di segno e che si estendono su di una «quad-word», formata, come è facile capire, da ben 4 word.

Dopo aver detto ancora una volta che il bit di segno è rappresentato dal bit più significativo in assoluto, vediamo la rappresentazione grafica di questo ulteriore formato:



Con questo terminiamo questa puntata e diamo appuntamento alla prossima, dove ci addentreremo ancora di più nel mondo alquanto complicato dell'80286

MC

RICORDI presenta:

Archimedes

La potenza del RISC nel personal computer più veloce del mondo

▷ Dalla Acorn di Cambridge, U.K., una nuova rivoluzione nell'informatica personale ▷ Archimedes, un computer (o meglio, un'intera serie) dalle altissime prestazioni ▷ Basato su un'unità centrale RISC (Reduced Instruction Set Computer) a 32 bit, Archimedes mette a vostra disposizione una potenza di calcolo finora sconosciuta nel campo dei personal computer ▷ Potenza per eseguire programmi in BBC BASIC a una velocità superiore a quella del linguaggio macchina di molti microcomputer tradizionali ▷ Potenza per accedere a diversi sistemi operativi, dall'ADFS all'MS-DOS* ad altri ancora ▷ Potenza per supportare linguaggi ad alto livello come C, FORTRAN, LISP, PROLOG, PASCAL (oltre a un BASIC formidabile) ▷ Potenza per generare un suono stereofonico di qualità digitale, e una grafica ad altissima definizione con migliaia di colori ▷ Potenza per collegare le più varie periferiche: digitalizzatori, interfacce MIDI, modem, eccetera ▷ Vincitore del Microcomputer Of The Year Award 1987 ▷ Archimedes, il personal computer più veloce del mondo, a un prezzo eccezionale: presso il vostro rivenditore o nei negozi RICORDI.

*MS-DOS è un marchio della Microsoft Corp.

Distributore esclusivo: G. RICORDI & C.
Settore Informatico
Via Salomone, 77
20138 MILANO
tel. 02/5082-315

DOPPIOUNI

Acorn

The choice of experience.

Un'azienda del gruppo Olivetti

Per maggiori informazioni, inviate questo coupon a G. RICORDI & C.
Settore Informatico, Via Salomone, 77, 20138 MILANO

Desidero avere maggiori informazioni su Archimedes

Nome: _____

Cognome: _____

Qualifica professionale: _____

Ditta, Ente o Scuola: _____

Indirizzo: _____