i trucchi del CP/M

a cura di Pierluigi Panunzi

Le funzioni del BDOS

Eccoci dunque giunti all'ultima puntata della serie riguardante le funzioni del BDOS, quelle funzioni cioè a disposizione dell'utente del sistema operativo CP/M, utilissime per svolgere tutte quelle operazioni elementari di gestione dei file, dell'I/O, ecc.

Le ultime funzioni che analizzeremo riguarderanno prevalentemente la gestione dei file, mentre una di esse si riferisce all'unità a dischi.

Funzione 35: Get File Size

Analogamente alle altre funzioni del BDOS che si riferiscono ai file, questa funzione richiede, all'atto della chiamata, anche il caricamento della coppia di registri DE con l'indirizzo della zona di memoria dove è stato posto il File Control Block (l'ormai ben noto FCB) relativo al file in esame: in particolare la chiamata avverrà nel seguente modo

```
LD C,35
LD DE,FCB
CALL 0005
```

Dopo aver visto come si ottiene l'esecuzione di tale funzione, vediamo ora quali sono le operazioni che essa compie.

Come dice il suo nome (anche se ciò è ingannevole, come vedremo), la funzione in esame consente di calcolare l'ampiezza «virtuale» del file in esame, che solo nel caso dei file sequenziali corrisponde all'ampiezza «fisica», effettiva, del file.

In particolare, a differenza delle altre funzioni del BDOS, fornisce il risultato nei due byte posti rispettivamente come 34° e 35° dell'FCB, riservati, come si può vedere analizzando l'FCB stesso, al «numero del record random»: ciò può sembrare un controsenso nel caso dei file sequenziali, mentre in realtà in tal caso si ha un'informazione esatta.

Per spiegarci meglio, diciamo che la funzione in esame pone nei due byte sopracitati il numero massimo di record da 128 byte presenti in un file sequenziale, pari cioè al numero che avrebbe il record successivo all'«end-of-file»: è proprio in questo caso la ampiezza del file sequenziale...

Nel caso invece dei file random, come già avevamo detto, la funzione non ci viene molto in aiuto in quanto la struttura stessa di un file random comporta enormi differenze concettuali.

Supponiamo infatti di creare un file random e di scrivere soltanto il record numero 200: già sappiamo che il CP/M creerà un opportuno extent in cui allocherà i 128 byte del record n° 200.

Usando in questo caso la funzione in esame, otterremo come risultato il valore 200 nei due byte 34-35 dell'FCB, che in prima analisi indicherebbe un'ampiezza del file di ben 200 record mentre viceversa sappiamo che di record ve n'è uno solo!

Tutto ciò perché, come ben sappiamo, il CP/M crea per un file random solo gli extent necessari: ricordiamo che un file random può infatti essere «sparso» e cioè con record separati da spazi vuoti.

Invece è altrettanto ben noto che un file sequenziale è per sua natura un file «pieno» e perciò il numero del «primo record non scritto» rappresenta proprio l'ampiezza del file, espressa come numero di record da 128 byte: evidentemente tale numero moltiplicato per 128 fornisce il numero di byte occupati globalmente dal file sequenziale, ad esempio sul disco.

Inoltre nel caso dei file sequenziali, per il fatto che dà come output il numero del primo record dopo l'end-of-file, ecco che la funzione in esame ci può tornare ancora utile quando vogliamo aggiungere ulteriori dati alla fine di un file, realizzando quella che si chiama genericamente la funzione di Append.

In particolare il chiamare dapprima questa funzione consente ad una successiva chiamata alla «Write Random» di aggiungere record al nostro file, che però non era random...

Ancora una volta perciò ritroviamo quanto avevamo accennato nelle primissime puntate riguardanti l'argomento del BDOS e cioè che il CP/M non crea assolutamente una barriera inviolabile di separazione tra i file random e quelli sequenziali, anzi, senza ovviamente travisarne la natura nettamente diversa, permette l'uso combinato di funzioni relative all'uno o all'altro tipo di file, senza limitazioni: ovviamente il programmatore deve sapere bene sia quali sono i propri intendimenti, sia quali sono gli effettivi risultati forniti da una certa funzione.

Abbiamo visto infatti che questa funzione applicata ai file random comporta risultati in alcuni casi completamente errati, non certo per colpa del CP/M....

Funzione 36: Set Random Record Number

A continuazione del discorso di poc'anzi, ecco subito una funzione dalle caratteristiche abbastanza particolari, che realizza una specie di «trait d'union» tra i due «mondi», quello random e quello sequenziale, e perciò a completa conferma di quanto affermato in precedenza.

Analogamente alla funzione precedente, la «Set Random Record Number» viene attivata con la stessa chiamata (a parte il valore C che in questo caso vale ovviamente 36) ed inoltre fornisce come output un valore (anche stavolta a 16 bit), contenuto nei byte 34 e 35 dell'FCB, indicante il valore corretto del numero di record random dell'ultimo record appena letto o scritto del nostro file sequenziale.

Interpretando bene il significato di questo risultato ottenibile, si vede che in tal modo si può costruire un file-indice (index file) che permette di accedere in modo random ad un file sequenziale. Bello, no?!

In particolare per ottenere ciò si può procedere per vari passi: dapprima si apre il file sequenziale e poi, leggendo sequenzialmente un record alla volta, se ne può estrarre un'opportuna «chiave» per potervi accedere (ad esempio in un file contenente dati anagrafici di clienti, il cognome del cliente ennesimo).

Ora si effettua la chiamata alla funzione in esame, ottenendo il valore desiderato: a questo punto la coppia di informazioni «chiave-numero del record random» può essere scritta in un apposito record del nostro «index file».

Ripetendo tale operazione per tutti i record del file sequenziale otterremo così un index file contenente tante coppie di corrispondenza chiave-numero record, quanti erano i record del file sequenziale.

A questo punto nessuno ci impedisce ad esempio di effettuare sul file di indice un'operazione di «sort» (ordinamento) delle «chiavi» in ordine alfabetico: a questo punto è veramente semplice indirizzare subito il record del file sequenziale contenente appunto le informazioni desiderate riguardanti il cliente in oggetto.

Altra utilizzazione di questa funzione è quella di ottenere l'informazione di dove si è effettivamente posizionati in un file sequenziale, sia per avere una correlazione tra il numero del record e la sua posizione effettiva, sia per «marcare» la posizione in cui ci si trova, alla quale riposizionarci dopo eventuali spostamenti nell'ambito del file.

Funzione 37: Reset Logical Disk Drive

Ecco la funzione delle quattro analizzate, non direttamente legata alla gestione dei file, ma viceversa utile per resettare (come dice il suo nome) singole unità logiche a dischi.

In particolare con questa funzione si può specificare quale o quali unità logiche a dischi devono essere resettate, a differenza della funzione 13 del BDOS (Reset Disk System), che viceversa resettava tutte le unità.

In questo caso bisogna porre nella coppia di registri DE una cosiddetta «bit map» indicante quali unità necessitano di reset: analogamente ad altri casi il bit 0 (il meno significativo di E) rappresenta l'unità logica «A:», il bit 1 l'unità logica «B:» fino ad arrivare al bit 15 (il più significativo di D) che si riferisce ad un'ipotetica, ma possibilissima sedicesima unità logica chiamata «P:».

In particolare un bit posto ad «1» indicherà che l'unità corrispondente deve essere resettata.

Per la compatibilità con il sistema operativo multi-utente MP/M, tale funzione fornisce come risultato un valore nullo nell'accumulatore A.

Inutile ricordare la necessità di resettare una certa unità logica all'atto del cambiamento di un dischetto: per chi ancora non lo sapesse, o meglio per chi ancora non fosse incorso nel comunissimo errore, diciamo che dopo aver cambiato un dischetto, se non si effettua un reset dell'unità a dischi in questione, il CP/M considererà che è stato commesso un errore ed in via precauzionale setterà il nuovo dischetto come «Read-Only» per non incorrere in ulteriori errori. Ritornando alla chiamata della funzione in esame, facciamo un esempio: supponiamo di voler resettare le unità a dischi «B:» e «D:» per effetto di un'operazione quale il cambio dei dischetti contenuti in tali unità.

Alle unità «B:» e «D:» corrispondono rispettivamente i bit 1 e 3 dei bit che vanno da 0 a 15 della coppia DE: perciò la bit map nel nostro caso sarà la seguente:

0000000000001010

corrispondente al valore esadecimale 000AH, da porre appunto nel registro DE.

Ecco che perciò nel nostro esempio la chiamata alla funzione di reset delle unità a dischi «B:» e «D:» sarà

```
LD DE,000AH
LD C,37
CALL 0005
```

Funzione 40: Write Random with Zero-fill

Per un motivo sconosciuto, all'interno del BDOS non sono state implementate le funzioni 38 e 39 e perciò dalla 37 si passa subito alla funzione 40.

A scoraggiare subito i «ricercatori» diciamo che proprio all'interno del BDOS c'è una «jump table» con gli «entry point» relativi alle varie funzioni: in corrispondenza alle funzioni 38 e 39 c'è il salto ad un unico punto, praticamente l'«exit point» delle altre funzioni e cioè il punto al quale tornano le varie routine al termine della loro funzione.

Inutile quindi effettuare la chiamata a tali due funzioni, in quanto comporterebbero un risultato identicamente nullo.

Invece la funzione in esame è in pratica un'estensione della funzione «Write Random», nel senso che oltre ad effettuarne le operazioni riempie ogni nuovo allocation block con dei byte nulli.

Tale funzione è stata aggiunta dai progettisti della Digital Research per aiutare quelli della Microsoft nella generazione del compilatore COBOL, per il quale risulta semplificata la gestione di file completamente azzerati nei record inutilizzati.

Visto che non tutti lavorano in COBOL, ecco che la funzione in esame fornisce un metodo semplicissimo, comodo e come suol dirsi «gratuito», per azzerare completamente un file random: basta a tale scopo scrivere un solo record in ogni allocation block per ottenerne l'azzeramento totale.

Per quanto riguarda la chiamata a tale funzione, essa si rifà alla chiamata alla funzione «Write Random» già citata: bisogna infatti porre nei soliti byte 34 e 35 dell'FCB il numero del record random sul quale vogliamo operare, nel registro C si deve porre il valore 40 (il numero della funzione) e nella coppia DE l'indirizzo del File Control Block del file in questione. Come risultato, oltre all'ovvio azzeramento del record, si otterrà nell'accumulatore un valore indicante l'esito dell'operazione stessa.

In particolare un valore nullo indicherà che l'azzeramento è avvenuto senza errori, mentre un valore non nullo indicherà una condizione anomala:

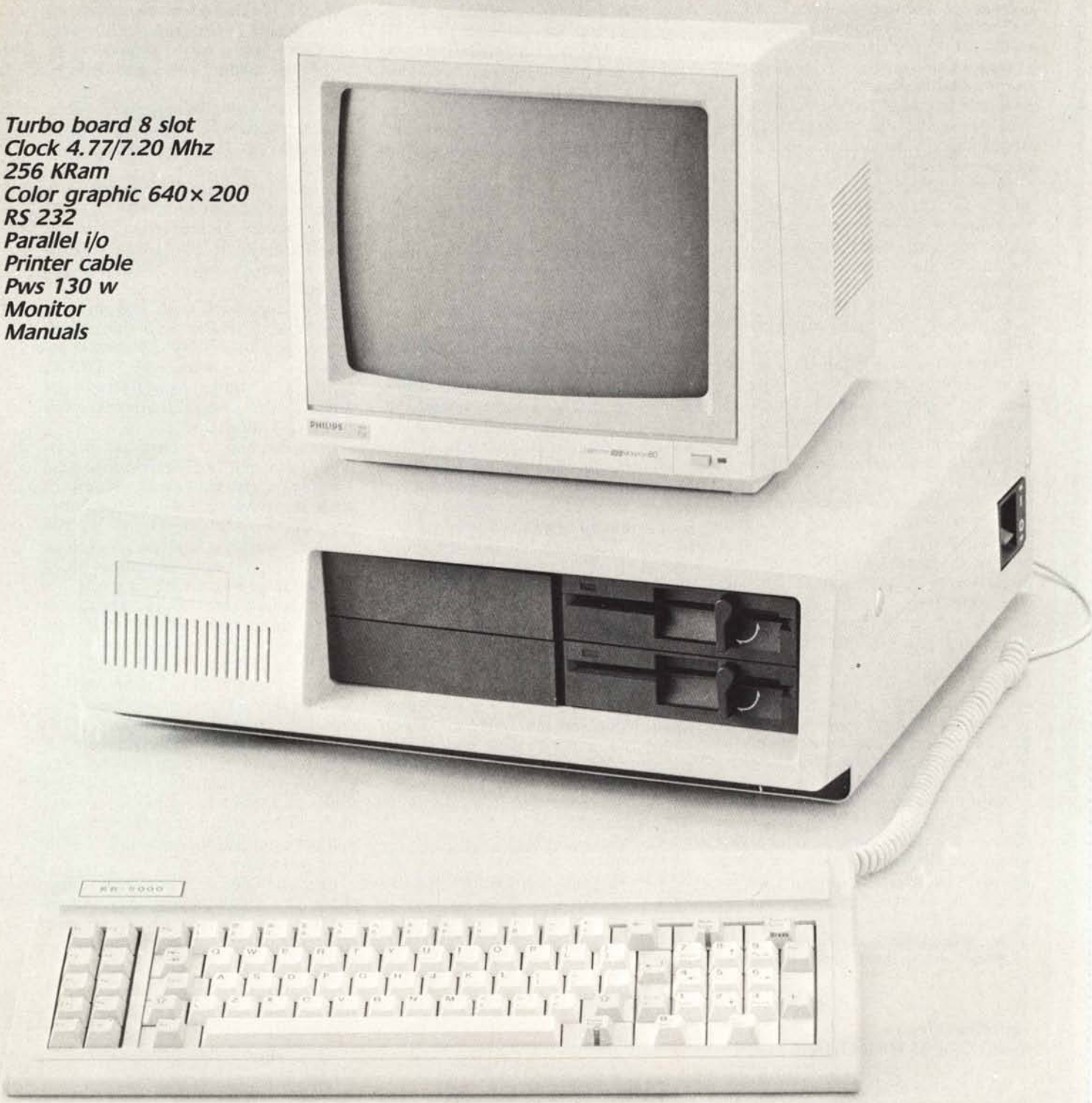
— il valore 3 indica che il CP/M non è stato in grado di chiudere l' extent corrente;

— il valore 5 indica che la directory era già piena;

— il valore 6 indica che si è tentata la scrittura al di là della fine fisica di un dischetto.

IL COMPATIBILE È APPARSO!

Turbo board 8 slot
Clock 4.77/7.20 Mhz
256 KRam
Color graphic 640 × 200
RS 232
Parallel i/o
Printer cable
Pws 130 w
Monitor
Manuals



Cerchiamo distributori per zone libere

QUASAR S.r.l. - 13050 Pratrivero (VC) - Tel. 015/778804/377 - Tx 211401 MILFIL