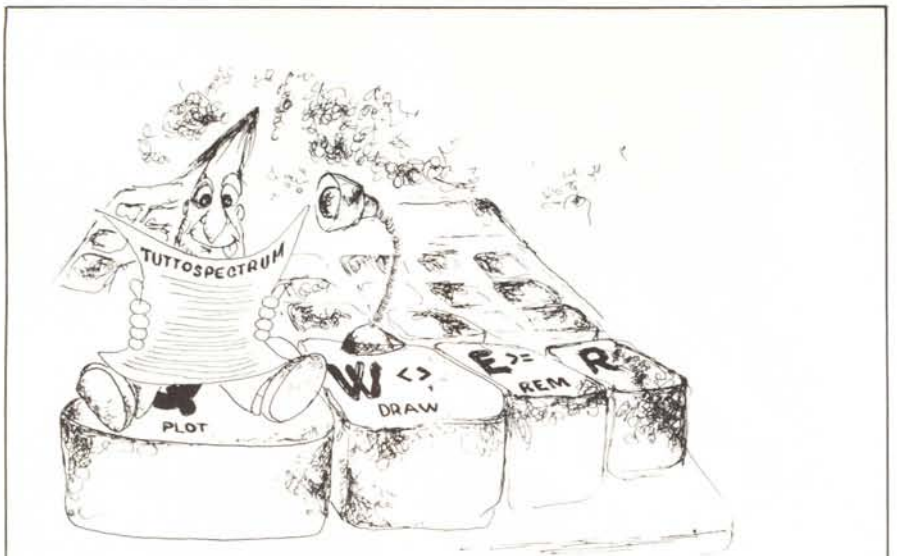

TUTTO SPECTRUM

di Fabio Schiattarella

Il linguaggio macchina sullo Spectrum

Quarta parte

In questo numero di TuttoSpectrum terminiamo la breve rassegna di tecniche per il passaggio di valori tra Basic e linguaggio macchina. Nella seconda parte della rubrica vogliamo invece cominciare ad occuparci di come sia possibile svolgere operazioni di ingresso/uscita, come la lettura della tastiera o la scrittura sullo schermo, direttamente in linguaggio macchina. Un unico esempio illustrerà tutte le tecniche che ci accingiamo ad esporre.

Per concludere il nostro discorso sul passaggio di parametri da un programma Basic ad uno in linguaggio macchina, vogliamo mostravi una tecnica che permette di realizzare qualcosa di analogo alla funzione svolta delle istruzioni DATA e READ del Basic. Il risultato viene ottenuto mediante l'utilizzazione di linee REM secondo il seguente formato

```
100 RAND USR <indirizzo del programma in l. m.>
```

```
110 REM <stringa dei dati da passare al prog.>
```

L'idea che viene sfruttata è la seguente. Nell'area delle variabili di sistema è presente la variabile NXTLIN (vedi pag. 128 del manuale) che punta all'indirizzo di memoria in cui comincia la linea di programma successiva a quella in corso di esecuzione. La variabile NXTLIN è memorizzata all'indirizzo 23637 (esadecimale 5C55, la parte meno significativa) e all'indirizzo 23638 (la parte più significativa). Nell'esempio visto sopra, durante l'esecuzione della linea

```
100 RAND USR <indirizzo programma in l. m.>
```

e quindi anche durante l'esecuzione del programma in linguaggio macchina la variabile di sistema NXTLIN punta all'inizio della linea

```
110 REM <stringa dei dati>
```

Studiamo quindi la rappresentazione in memoria di questa linea di programma. Quanto diremo vale anche in generale per la rappresentazione delle linee di programma Basic in memoria.

In figura 1 è descritta la rappresentazione in memoria della linea

```
110 REM prova
```

I primi due byte della rappresentazione contengono il numero di linea che appare nei listati del programma Basic. Notate come, contrariamente a quanto avviene in tutti gli altri casi in cui venga rappresentato un numero intero con due byte, il numero di linea viene memorizzato con la parte più significativa nel primo byte e con la parte meno significativa nel secondo, ovvero nell'ordine in cui noi «umani» siamo abituati a scrivere le cifre.

La seconda coppia di byte contiene la lunghezza della linea di programma vera e propria. Il valore della lunghezza è pari al numero dei caratteri che effettivamente la costituiscono più uno, per tenere conto del carattere ASCII 13, corrispondente al tasto ENTER, che segnala la fine della linea. Occorre anche tener conto che le parole chiave del Basic, come THEN, REM, SAVE eccetera, occupano un solo byte, in quanto hanno una codifica speciale (sono, come si usa dire «tokenizzate»).

Notate che la linea citata nell'esempio è stata scritta senza introdurre nessuno spazio bianco superfluo tra il numero di linea e la parola chiave REM, e tra questa e l'inizio della stringa. È lo Spectrum che, quando è necessario, introduce spazi per separare le parole tra loro; a questi spazi tuttavia non corrisponde nessuna rappresentazione in memoria.

Nell'esempio di figura 1 il valore della lunghezza è 7: 1 byte contiene il codice della prola chiave REM, 5 byte sono necessari per contenere le lettere della parola PROVA ed infine 1 byte contiene il carattere ASCII 13 che indica la fine della linea.

Se un programma in linguaggio macchina viene lanciato con la sequenza di linee Basic sopra mostrata

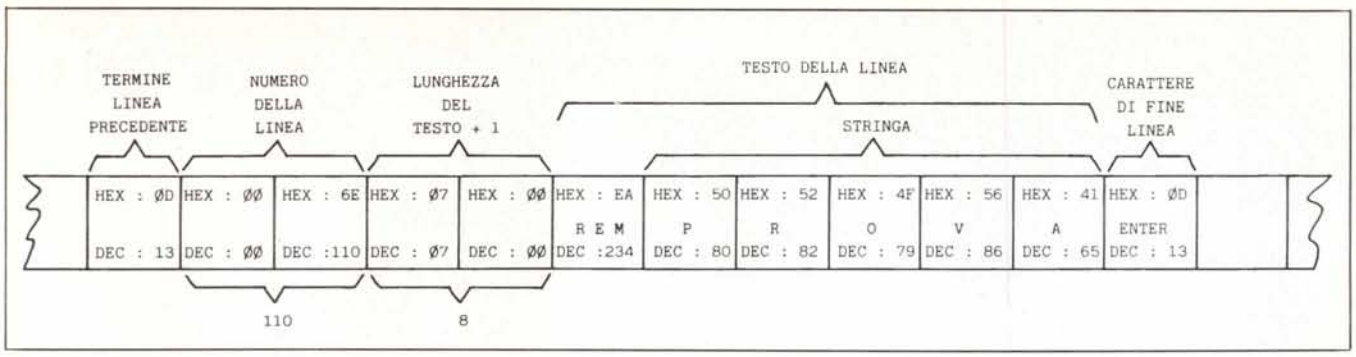

Figura 1 - Rappresentazione in memoria della linea «110 REM PROVA».

esso può accedere facilmente ai dati contenuti nella linea REM. Il loro indirizzo iniziale è dato infatti dal contenuto della variabile NXTLIN + 4, la loro lunghezza è data dalla lunghezza del testo della linea diminuita di due.

È possibile in alternativa fornire al programma in linguaggio macchina il solo indirizzo iniziale, e fare in modo che esso riconosca autonomamente la fine della stringa dati quando incontra il carattere ASCII 13 di fine linea.

L'uso di questo metodo per il passaggio di parametri da programma Basic a linguaggio macchina è limitato ad alcuni casi particolari. Innanzi tutto esso risulta conveniente quando i dati da trasferire sono sempre gli stessi ad ogni esecuzione del programma, come del resto accade anche per le linee DATA del Basic. In secondo luogo l'unico tipo di dati che è facilmente trattabile con questo medodo sono le stringhe di caratteri stampabili. Con caratteri stampabili intendiamo le lettere maiuscole e minuscole, le cifre decimali, lo spazio bianco e tutti i segni di interpunzione e i simboli che fanno parte dell'insieme dei caratteri ASCII che vanno dal 32 al 127 (vedi la tabella riportata a pag. 135 e seguenti del manuale). Di conseguenza il metodo risulta vantaggioso principalmente nel caso in cui i dati da trasferire siano parole e frasi. Un esempio potrebbe essere il caso in cui si debbano caricare in memoria in una posizione determinata dei messaggi di errore da utilizzare nell'ambito di un programma in linguaggio macchina.

In linea di principio sarebbe possibile inserire nella linea REM caratteri corrispondenti ad una qualsiasi delle 256 possibili configurazioni di un byte. Ad esempio, osservando la tabella dei caratteri riportata in fondo al manuale, notiamo come ai codici maggiori di 128 corrispondono molte parole chiave del Basic ed i caratteri definiti dall'utente. Tuttavia ci sembra che il gioco non valga la candela e che convenga utilizzare il metodo solo nel caso di stringhe alfanumeriche.

Ovviamente la sintassi utilizzata normalmente nel Basic per le linee DATA non è più valida. Le stringhe non vanno racchiuse tra virgolette e non vanno inseriti spazi superflui. Una espressione del tipo

```
110 REM 192,76
```

non viene interpretata come la richiesta di riempire due byte in memoria con le cifre 192 e 76, ma come la richiesta di caricare una sequenza di sei caratteri tra cui una virgola.

Un passo avanti

Avrete forse notato come tutti gli esempi che abbiamo descritto in questi numeri di TuttoSpectrum abbiano tutti la stessa impostazione. C'è un programma Basic chiamante che si occupa dell'ingresso dei dati da tastiera e li predispone in un formato riconoscibile dal programma in linguaggio macchina. Questo viene lanciato per mezzo della funzione USR ed al termine restituisce i risultati dell'elaborazione al programma Basic chiamante, il quale si occupa di stamparli sul video.

Si tratta di una tecnica piuttosto comune e di utilizzo molto generale. Essa semplifica il compito del programmatore in quanto lo libera dall'impegno, a volte gravoso, di dover gestire in linguaggio macchina l'ingresso e l'uscita dei dati. Tali operazioni di lettura e scrittura possono essere svolte mediante l'impiego di pochi e semplici comandi Basic, mentre si può deputare al linguaggio macchina lo svolgimento dell'elaborazione principale o di una sua parte, a tutto vantaggio della velocità di esecuzione.

Un procedimento analogo può essere seguito ogni qualvolta si presenti il caso di un'operazione complessa da programmare in linguaggio macchina, ma semplice da eseguire in Basic, ad esempio moltiplicazioni o divisioni di numeri in virgola mobile.

Sebbene questo sia il modo decisamente più semplice per gestire le operazioni di ingresso e uscita per un programma in l/m, non è detto che sia sempre il più conveniente. Ad esempio, se un programma in codice macchina dovesse eseguire la stampa di un messaggio a metà dell'elaborazione occorrerebbe ritornare al Basic, eseguire una istruzione PRINT e poi lanciare di nuovo il programma con una USR, facendolo proseguire dal punto in cui si era fermato. Se da una parte ciò risulta poco pratico, dall'altra non è detto che svolgere in linguaggio macchina una operazione di ingresso o uscita, come la scrittura di una stringa sul video o la lettura dell'ultimo tasto premuto, sia eccessivamente difficile da realizzare.

Il nostro intendimento è quindi quello di descrivere semplici tecniche per gestire in linguaggio macchina l'ingresso e l'uscita dei dati. Cominceremo con la scrittura sul vedeo.

Nello Spectrum qualsiasi operazione di scrittura di caratteri sul video viene eseguita mediante la chiamata di una apposita routine. Tale chiamata avviene per mezzo dell'istruzione

```
RST #10
```

il cui codice esadecimale è D7 (decimale 215). Per quanto ci riguarda questa istruzione è in tutto e per tutto equivalente ad una

```
CALL #0010
```

ovvero ad un salto a sottoprogramma a partire dall'indirizzo esadecimale 10. L'unica differenza tra la prima forma e la seconda è che una occupa in memoria un solo byte mentre l'altra ne occupa tre.

La chiamata di questa routine produce come effetto la stampa sul video, alla posizione di stampa corrente, del carattere il cui codice è contenuto nel registro accumulatore A. Dopo la scrittura del carattere vengono aggiornate le coordinate di stampa.

Per stampare un carattere occorre quindi caricare nel registro accumulatore A il suo codice. I codici relativi a tutti i caratteri, alle parole chiave del Basic ed ai caratteri grafici definibili dall'utente sono riportati nel manuale d'uso nell'appendice A a pagina 135 e seguenti.

```
 10 CLS
 20 FOR i=29000 TO 29036
 30 READ a
 40 POKE i,a
 50 NEXT i
 60 DATA 62,2,205,1,22,62,22,215,62,10,215,62,0,215,42,85,92,35
 70 DATA 35,78,35,70,11,11,35,35,84,93,120,177,11,200,26,215,19,24,247
100 RANDOMIZE USR 29000
110 REM FATE ATTENZIONE A NON LASCIARE  SPAZI BIANCHI PRIMA E DOPO LA   REM
```

Figura 2

```
*HISOFT GENS2 ASSEMBLER*
 Copyright HISOFT 1983
  All rights reserved

Pass 1 errors: 00

7148            10         ORG   29000
5C55            20 NXTLIN  EQU   #5C55
7148 3E02       30         LD    A,#02 ; INDIRIZZA LA STAMPA VERSO LO SCHERMO
714A CD0116     40         CALL  #1601 ; RICHIEDE L'APERTURA DEL CANALE RELATIVO
714D 3E16       50         LD    A,#16 ; MODIFICA POSIZIONE DI STAMPA
714F D7         60         RST   #10
7150 3E0A       70         LD    A,#0A ; NUOVA RIGA
7152 D7         80         RST   #10
7153 3E00       90         LD    A,#00 ; NUOVA COLONNA
7155 D7        100         RST   #10
7156 2A555C    110         LD    HL,(NXTLIN) ; PUNTATORE ALLA SUCCESSIVA LINEA BASIC
7159 23        120         INC   HL ; AVANZA FINO A TROVARE LA
715A 23        130         INC   HL ; LUNGHEZZA DELLA LINEA
715B 4E        140         LD    C,(HL) ; CARICA LA LUNGHEZZA IN BC
715C 23        150         INC   HL
715D 46        160         LD    B,(HL)
715E 0B        170         DEC   BC ; DECREMENTA DI DUE PER ELIMINARE DAL CONTO
715F 0B        180         DEC   BC ; LA "REM" E L' "ENTER" ALLA FINE
7160 23        190         INC   HL ; SI PORTA ALL' INIZIO DELLA STRINGA
7161 23        200         INC   HL
7162 54        210         LD    D,H ; CARICA L'INDIRIZZO INIZIALE IN DE
7163 5D        220         LD    E,L
7164 78        230 STAMPA  LD    A,B ;INIZIO CICLO DI STAMPA
7165 B1        240         OR    C ; L' OR DI B E C E' 0 SE LA STRINGA E' TERMINATA
7166 0B        250         DEC   BC ; DECREMENTA CONTATORE DI LUNGHEZZA
7167 C8        260         RET   Z ; RITORNA A BASIC SE LA LUNGHEZZA ERA 0
7168 1A        270         LD    A,(DE) ; CARICA IN A IL CARATTERE DA STAMPARE
7169 D7        280         RST   #10 ; ESEGUE LA STAMPA
716A 13        290         INC   DE ; SI POSIZIONA SUL CARATTERE SUCCESSIVO
716B 18F7      300         JR    STAMPA ; RIPETE IL CICLO

Pass 2 errors: 00

Table used:    39  from   228
```

Figura 3

L'esempio di figura 2 e 3 vuole essere illustrativo di tutto quanto vi abbiamo esposto in questo numero di TuttoSpectrum. Il programma in linguaggio macchina esegue la stampa sul video di una stringa di caratteri a partire dalla linea 10. La stringa da stampare è contenuta in una linea REM che segue la chiamata del programma. La stringa viene letta dal programma in linguaggio macchina secondo le modalità descritte nella prima parte di questo articolo.

Per esempio, se volessimo stampare «MC» sul video, potremmo utilizzare questo semplicissimo programma

```
LD A,#4D
RST #10
LD A,#43
RST #10
RET
```

Questa breve sequenza di istruzioni fa comparire sul video le due lettere M e C una accanto all'altra. La routine di stampa quindi, dopo aver scritto un carattere, provvede automaticamente a spostarsi a destra di una posizione, prima di scrivere il successivo. Quando arriva in fondo ad una riga prosegue a colonna uno della riga successiva. Fino ad ora abbiamo parlato della posizione di stampa corrente come di una variabile controllata dal sistema, il quale provvede ad aggiornarla automaticamente per non scrivere sopra ai caratteri precedenti. Sarebbe assai più comodo poter scrivere o incominciare a scrivere in qualsiasi posizione dello schermo si desideri.

Per aggiornare la posizione di stampa occorre fare riferimento allo schermo come ad una griglia di 24 righe per 32 colonne numerate come nella figura a pagina 76, capitolo 15 del manuale d'uso. Un primo modo per variare la posizione corrente di stampa è di agire sulla coppia di variabili di sistema S_POSN. La prima delle due variabili, memorizzata nella locazione 23688, deve contenere un numero pari a 33 meno il valore della colonna di stampa che si desidera impostare. La seconda variabile, alla locazione 23689, deve contenere 24 meno il valore della riga di stampa.

Esiste un secondo metodo che permette di evitare questa doppia sottrazione ogni volta che si desidera variare la posizione di stampa. Esso consiste nell'inviare in stampa una sequenza di tre byte. Il primo deve essere il carattere speciale di controllo corrispondente alla parola chiave del Basic AT. Il codice di questo carattere è 22 decimale, 16 esadecimale. Gli altri due byte contengono il nuovo numero di riga e di colonna che si desidera impostare.

Per esempio se volessimo incominciare a scrivere a partire dalla prima colonna della quinta riga, tenendo conto che la numerazione delle righe e delle colonne parte da zero, dovremmo impostare la seguente sequenza di istruzioni

```
LD A,#16
RST #10
LD A,#04
RST #10
LD A,#00
RST #10
```

È importante sottolineare che i comandi CLS e RUN riportano automaticamente la posizione di stampa alle coordinte 0,0 in alto a destra dello schermo. La routine chiamata mediante la RST # 10 provvede alla stampa dei caratteri su dispositivi fisici e logici diversi: la stampante, la parte superiore dello schermo, la parte inferiore riservata ai messaggi prodotti dal sistema operativo. Prima di cominciare qualsiasi operazione di stampa sul video è buona norma assicurarsi che il dispositivo di stampa indirizzato dal sistema sia proprio la parte superiore dello schermo. Per fare ciò occorre aprire il canale di uscita dei dati «S». Ciò viene ottenuto mediante la sequenza di istruzioni:

```
LD A,#02
CALL #1601
```

Per una spiegazione esauriente su cosa siano i canali di ingresso e uscita e come essi siano organizzati sullo Spectrum vi rimandiamo all'articolo apparso in TuttoSpectrum sul numero 38 di MC del febbraio 1985 a pagina 86.

Nei prossimi numeri cominceremo ad esaminare come sia possibile ottenere in linguaggio macchina tutta la flessibilità e la potenza della PRINT del Basic. Vedremo come sia possibile modificare i colori di BORDER, INK e PAPER, come sia possibile cancellare lo schermo, tutto o in parte, come si possa fare lo scroll del video per un numero di linee desiderato, e molte altre cose. MC

AL COMPLETO SERVIZIO DEI RIVENDITORI

ISFO

agente esclusivo per Lazio, Umbria e Abruzzo:

ASEM

- linea **PC 100** (compatibili IBM)
- linea **AM** (compatibili Apple)
- monitor **TAXAN** ○ memorie di massa e accessori hardware per Apple, IBM, Olivetti e compatibili

agente esclusivo per il Lazio:

telcom

- stampanti ad aghi **MITSUI**
- floppy **MAXELL**
- stampanti low cost **CP/JP - 80**
- stampanti a margherita **JUKI**
- accoppiatori acustici **NOVATION CAT, ANDERSON - JACOBSON** ○ plotter **YEW, ENTER C** digiter **GTCO** ○ mouse **MOUSE SYSTEM**

agente esclusivo per Lazio e Umbria:

J.soft ○ software **J.soft** per Apple, IBM, Olivetti M24 e compatibili IBM

GRUPPO EDITORIALE JACKSON ○ **tutti i libri della casa editrice**

disponibili LOTUS e SYMPHONY per IBM e compatibili e JAZZ per Macintosh

cover ○ accessori per home e personal computer

ISFO - 00145 Roma, Viale Leonardo Da Vinci 89 - Tel. 06. 5133791/5141371

Canon V 20 l'MSX

MSX, UN SOLO SOFTWARE PER TUTTI.

MSX vuol dire microsoft extended basic: tutti i computers prodotti dalle case aderenti a questo speciale progetto utilizzano lo stesso sistema operativo. Il vantaggio per l'utilizzatore è di portata eccezionale: la perfetta intercambiabilità dei programmi e delle periferiche - stampanti, unità floppy disk, tavoletta elettronica ecc... In pratica, tutto il software - e l'hardware - delle varie marche, è utilizzabile senza alcun problema di compatibilità!

MICROSOFT È IL NUMERO 1 DEL SOFTWARE.

Lo sviluppo del sistema MSX è stato affidato al colosso americano Microsoft, leader mondiale del software. Le società consorziate sono oltre venti, in pratica il meglio oggi esistente al mondo, ed il loro progetto è esclusivamente destinato agli utenti del sistema MSX.

SOFTWARE PER GIOCO E SOFTWARE SUL SERIO.

La biblioteca dei programmi da far girare sulle macchine MSX, tutte completamente compatibili come si è visto, è in corso di rapido sviluppo. Ai programmi di base-data base, foglio elettronico, word processing, grafica - si unisce il software applicativo, che il Canon V-20 può sfruttare al meglio con la sua versatilità e potenza. E naturalmente non mancano i videogiochi, godibili a pieno effetto nelle loro grafiche superbe su un normale TV color.

CANON V-20 MSX È UNA POTENZA.

Eccovela tutta in pochi dati eloquenti: 64 KB RAM più 16 di video RAM, 32KB ROM microprocessore Z-80; linguaggio MSX basic, due alloggiamenti (slots) per cartucce ROM o per espansioni, interfacce per stampante (Nella foto Thermal Printer Canon T-22A) e per registratore a cassette incorporate, tastiera professionale con tasti rigidi (72), tasti guida cursore di grandi dimensioni, due uscite per joystick, uscita per televisione

elevato a potenza.

sistema Pal, testi: 25 linee per 40 colonne, grafica: 256 punti×192 punti, 16 colori, suono: 8 ottave su 3 toni, possibilità di interfacciamento parallelo e seriale. Il DOS (Disk Operative System) dell'MSX permette sofisticati utilizzi tecnici e gestionali, grazie alla possibilità di unità floppy disk.

CANON V-20 MSX UNA SCELTA INTELLIGENTE.

Canon V-20 è l'Home Computer che, comprato oggi, vale per il futuro, senza rischi, senza cambi, senza problemi. Canon MSX V-20 vuol dire non avere mai in futuro alcun problema di compatibilità di hardware e di software. C'è una scelta più sicura ed intelligente?

Mi interessano più informazioni del:

☐ A-200 - Il personal computer Canon con stampante laser.

☐ X07 - Il computer portatile (hand-held) Canon con Memory-Card.

☐ V-20 - L'home computer Canon con sistema MSX.

NOME _____

COGNOME _____

VIA _____

CAP/CITTA _____

Inviare questo tagliando a: Canon Italia S.p.A.
Viale dell'Industria, 13 - 37012 Bussolengo, Verona.

UN ANNO DI GARANZIA

Canon

ITALIA