SPECIALE UTILITIES

MSX COMPUTER MAGAZINE

N.11 Sped in abb. post. Gr. III L. 9.000

UNA MAGLIETTA IN REGALO!

a chi si abbona a

MSX COMPUTER MAGAZINE

✸

sei magnifiche cassette di programmi di gioco e di utilità, sempre più belle e ricche!

✸

il prezzo dell'abbonamento (Lire 50 mila) è bloccato per sei numeri e non ti verranno quindi richiesti aumenti (già subito intanto risparmi 4 mila lire)!

✸

avrai subito, direttamente a casa, un'elegante maglietta (realizzata con le riviste consorelle Elettronica 2000 e Load'n'Run) assolutamente gratis!

ABBONATI OGGI STESSO

Basta inviare un vaglia ordinario (quello rosa, da richiedere in un qualunque ufficio postale) di lire 50 mila.
Indica esattamente da quale fascicolo desideri l'abbonamento ed i tuoi dati chiari e precisi. Indirizza a MSX Computer Magazine, C.so Vitt. Emanuele, 15 - 20122 Milano.

MSX Computer Magazine è edita da Arcadia srl,
C.so Vitt. Emanuele 15, Milano.
Tel. 02/706329 (solo giovedì h. 15-18).
Una copia L. 9.000.
Fotocomposizione: Composit.
Stampa: Garzanti,
Milano. Distribuzione: SO.DI.P. Angelo
Patuzzi srl, Via Zuretti 25, Milano.
Registrato Trib. Milano N. 52 del 2/2/85.
Resp. Sira Rocchi.
Sped. in abb. post. Gr. III/70.

MSX COMPUTER MAGAZINE 11

IN QUESTO NUMERO

★ L'ISTRUZIONE INPUT

★ CORSO LINGUAGGIO MACCHINA

★ SPRITES FACILI

★ DIECI PROGRAMMI DIECI

- TENNIS
- DRAGON BUSTER
- SLALOM
- CORSO DI INGLESE
- PARÀ
- HELLCOPTER
- SWORD AND SPELL
- POKER
- ASSEMBLER
- SCAMBIO PROGRAMMI

MSX TAPE SOFT

I programmi contenuti in questo fascicolo di MSX COMPUTER MAGAZINE sono tutti compatibili con qualsiasi sistema MSX. Ecco per voi ancora 10 bei programmi! Ricordate di collegare la spina del controllo motore alla presa REM del vostro registratore se quest'ultimo la possiede. Assicuratevi che la spina nera sia collegata alla presa EAR del registratore e che la spina rossa sia inserita nella presa MIC. Se il vostro mangiacassette non possiede la presa REM, fate particolare attenzione a quando un programma è stato caricato o deve esserlo, affinché il nastro scorra per il giusto tempo. Appena vedete apparire sul video, dopo un comando di caricamento, la scritta OK, spegnete il registratore. Nelle istruzioni che seguono troverete, accanto al titolo ed al codice di caricamento di ogni programma, la notazione della memoria minima necessaria per vedere il programma.

Nella cassetta allegata a questo fascicolo troverete, sul lato A:
Tennis, Dragon, Buster, Slalom, Corso di Inglese, Parà.
Sul lato B:
Hellcopter, Sword and Spell, Poker, Assembler, Scambio programmi.

1

TENNIS

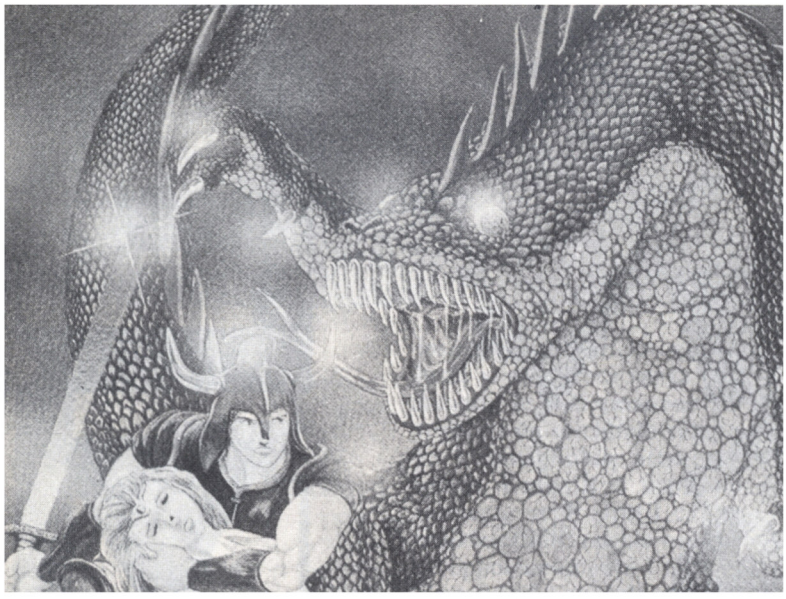
(CLOAD "Tennis" - 16 K)

di P. Parma e F. Postiglione

Di giocare a tennis siamo capaci un po' tutti, le regole almeno le conosciamo. Ma avete mai provato a sfidare l'MSX? Ecco una buona occasione per farlo. Scegliete se usare la tastiera o il joystick e la sfida avrà inizio. Il vostro è il giocatore di destra, guidarlo è semplicissimo con i tasti cursore o con il joystick.

Con la barretta o con il tasto di fuoco muovete la racchetta per colpire la pallina ricordando che quando quest'ultima è colpita nella parte bassa tende ad alzarsi, mentre quando è colpita nella parte alta tende ad abbassarsi.

Nelle prime partite lo sfidante MSX potrà sembrare un avversario imbattibile ma niente paura: con un po' di pratica gli potrete fare, come si suol dire... le scarpe!



2

DRAGON BUSTER

(CLOAD "DRAGON" - 32 K)

di A. Zanetti

Questa volta dovrete liberare la principessa tenuta prigioniera dal solito cattivissimo Drago. Dovrete affrontare duri combattimenti con maghi, dinosauri, scheletri, serpenti, pipistrelli e draghetti vari che cercheranno a tutti i costi di impedirvi di raggiungere il vostro scopo. Passerete di volta in volta da un corridoio ad una stanza: se nel corridoio incontrerete un pipistrello o un serpente vi basterà toccarli per perdere ben due vite. Usate dunque la spada (premendo il pulsante di fuoco) e fate attenzione alla bocca ed al teschio che vi inseguono e che, se vi toccano, vi faranno perdere la partita. In una delle stanze potreste incontrare il mago, il dinosauro, oppure lo scheletro: con lo scheletro vi toccherà sostenere un vero e proprio duello e dovrete, per ucciderlo, colpirlo due volte. Anche il dinosauro, che vi sputa fiamme addosso, va colpito due volte perché possa considerarsi eliminato. Per distruggere il mago, invece, che vi lancia contro lingue di fuoco e che non potete schivare, dovrete colpirlo ben cinque volte! Colpito un nemico in una stanza, avrete diritto ad un premio (casuale) fra questi: un'ampolla che aumenta la vostra vitalità; una fiamma per combattere il drago; una chiave; niente (succede a volte, se avete già trovato le tre chiavi). Il combattimento con il Drago è simile a quello con gli altri nemici: la sua vitalità è però di 60! Premendo F1 sparerete le fiamme che avrete guadagnato e se queste colpiranno il Drago diminuiranno la sua vitalità di 2. Ucciso il Drago passerete allo schermo successivo nel quale il numero degli edifici e la vitalità dei nemici saranno maggiori.

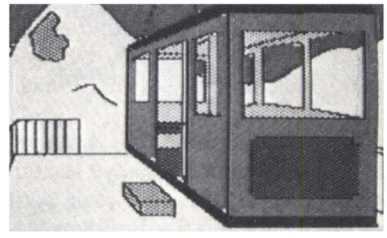
3

SLALOM

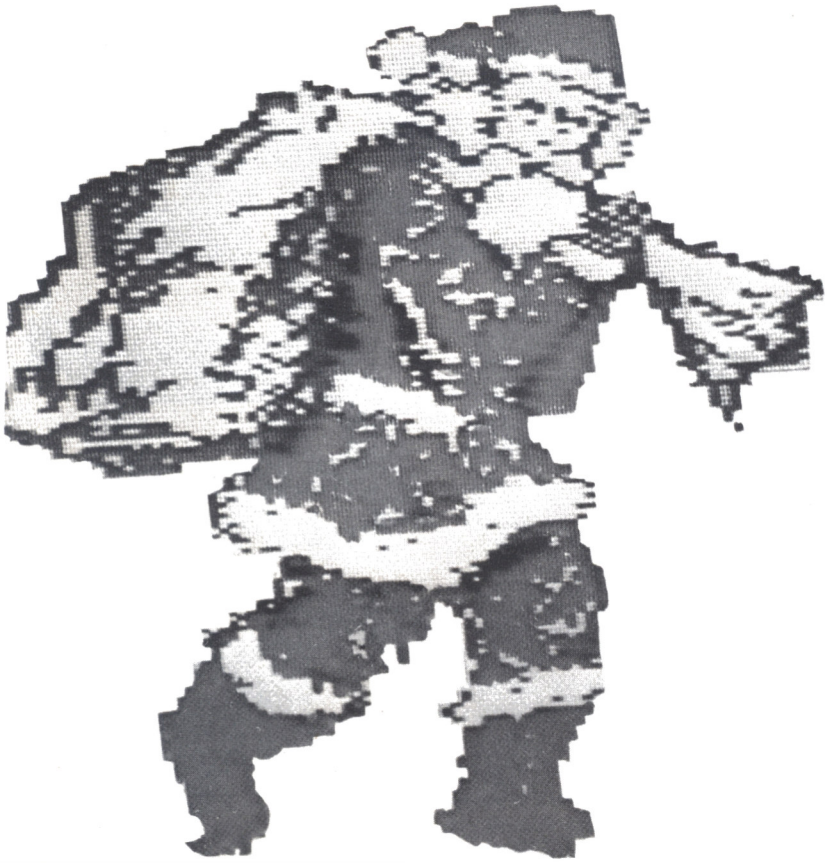
(CLOAD "SLALOM" - 16 K)

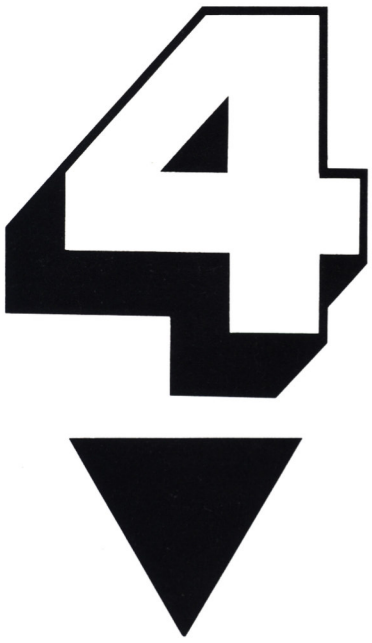
di C. Rocco

Sci ai piedi, lanciamoci tutti nello slalom più difficile ed appassionante della stagione. Possono gareggiare fino a nove sciatori; udito il segnale di partenza il giocatore parte e, a poca distanza dalla prima porta che incontra, entrano in funzione i tasti cursore o il joystick. Potete spostare il vostro sciatore a destra o a sinistra per farlo passare fra le porte. La velocità aumenta o diminuisce a seconda della posizione dello sciatore. Superata l'ultima porta, il giocatore taglia il traguardo ed il computer segnala il tempo di manche. Hanno accesso alla



seconda manche, naturalmente, solo gli sciatori che hanno concluso lodevolmente la prima. Al termine della sfida viene redatta una classifica finale nella quale vengono comunicati il tempo del vincitore (in minuti, secondi e centesimi di secondo) ed il distacco dagli altri concorrenti classificatisi. La sfida sui campi di neve è dunque aperta. Vinca il migliore!



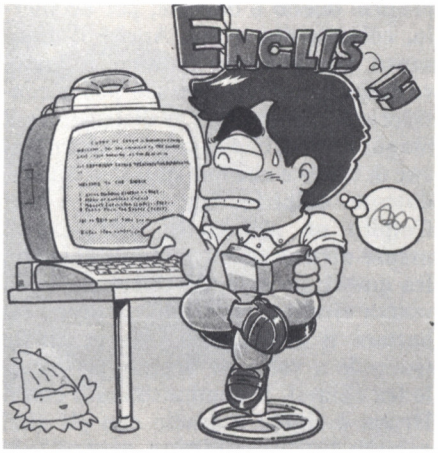


INGLESE

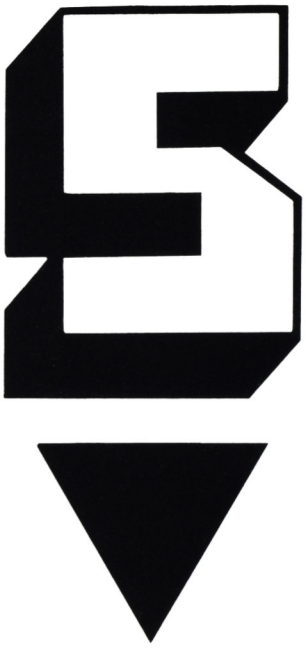
(CLOAD "PARTE 1" - 32 K)

di G. Bellomusto

Attesissimo, richiestissimo, ecco un corso-bliz d'inglese in quattro puntate per tutti quelli che il computer vogliono usarlo anche per imparare. Il programma è interattivo, facilissimo da utilizzare: sarete chiamati a rispondere a varie domande che il vostro MSX vi porrà di volta in volta e dovrete impostare delle frasi che il computer stesso, nelle vesti di insegnante, controllerà.



Un po' programma didattico, un po' gioco, questa prima parte comprende: verbo essere; che ora è; articoli indefiniti; articoli dimostrativi; pronomi possessivi.

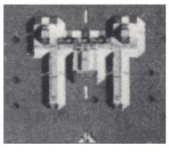


PARÀ

(CLOAD "PARÀ" - 32 K)

di M. Pessina e R. Brun

Aiutate l'amico paracadutista ad atterrare sull'isoletta in mezzo al mare. Gli schermi sono tre: nel primo si effettua un lancio di allenamento con vento costante; nel secondo si eseguono tre lanci con il vento che, però, varia d'intensità e direzione; nel terzo, le condizioni di vento sono come nel secondo ma, in sovrappiù, l'isola si sposta! Gli schermi 2 e 3 verranno superati solo quando tutti e tre i lanci saranno stati effettuati senza errore.



Inoltre, a seconda del livello il vento varia sempre più frequentemente e nel terzo schermo l'isola si sposta di tanti quadretti quanto è il valore del livello raggiunto. Premuto SPAZIO per giocare attendete un po', quindi premete fuoco e SPAZIO: una sola volta per lanciare il parà in caduta libera, due volte per aprire il paracadute. Premete i tasti cursore o il joystick per i movimenti laterali. Attenzione: il gioco si divide in due parti: basic (CLOAD "PARÀ") ed L/M (si carica dopo aver dato RUN al programma basic).

6

HELLCOPTER

(CLOAD "HELL" - 32 K)

di P. Sigalini

Sembra facile: dovete penetrare con il vostro elicottero in un dedalo di gallerie, raggiungere il mostro che vi risiede e distruggerlo. Per farlo, usate i tasti cursore o il joystick; per sparare usate la barra di spazio o il pulsante di fuoco. Dovrete bombardare le torri, i depositi e le basi, scansare mostri e missili, raccogliere i piloti, fare rifornimento. Non dovrete invece sbattere contro le pareti, finire contro depositi, mostri, mine etc., far esaurire il carburante, bombardare i piloti! Difficoltà: i mostri ameboidi sono parecchio tenaci e vi seguono inesorabilmente. I generatori di turbolenze rendono poco controllabile il vostro elicottero: per questo dovete cercare di distruggerli. La torre laser è pericolosa: se il suo mirino vi raggiunge, il laser vi abbatterà senza pietà alcuna. Attenti: le basi emanano ad intervalli regolari un micidiale raggio verticale ed è consigliabile attaccarle a bassa quota. I missili si muovono orizzontalmente negli stretti cunicoli e sono indistruttibili: occhio quindi a scegliere opportunamente i tempi. La mina vi segue ed è più veloce delle amebe. Il mostro finale è protetto da uno scudo che dovrete distruggere: ricordate che usa i suoi poteri per sbattacchiarvi in ogni direzione. Fuel: colpendolo farete rifornimento di carburante. I piloti si sbracciano per chiedere aiuto: salvandoli otterrete un elicottero in più; lo stesso accadrà ad ogni 750 punti totalizzati. Coraggio, tocca a voi!

7

SWORD/SPELL

(CLOAD "S&S" - 32 K)

di S. Guarnaschelli

Siete in un labirinto e dovete uscirne utilizzando un calderone nel quale debbono essere sacrificati tutti gli oggetti raccolti durante il percorso. Per muovervi usate i cursori o il joystick, per sparare, premete la barra. A seconda del personaggio scelto, la cadenza di fuoco sarà più o meno elevata a partire dal cavaliere (meno veloce) fino all'elfo. È possibile usare gli oggetti magici tramite i tasti funzione:

F1 = anello = permette di uscire dalla stanza
F2 = croce = attacca il fantasma e il demone
F3 = armatura = aumenta le difese
F4 = scroll = attacca qualsiasi mostro
F5 = heal = risana le ferite
F6 = calderone = richiama il calderone.

F6 può essere usato solo se si sono raccolti almeno 30 punti di energia che si ottengono eliminando mostri e raccogliendo oggetti: il tasto fa apparire il calderone ed eventualmente la porta. Sullo schermo a sinistra appare la stanza nella quale si svolge l'azione. In alto a destra vedete nell'ordine: la vitalità dell'eroe, quella del mostro, il numero degli oggetti, il potere accumulato.

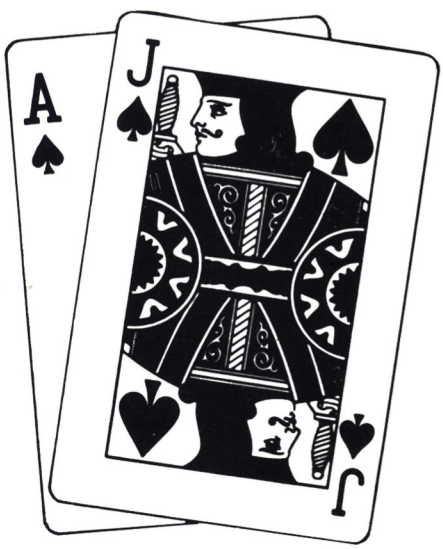
POKER

(CLOAD "Poker" - 16 K)

di P. Tamburelli

Questo Poker, versione computerizzata dell'omonimo gioco d'azzardo, riproduce molto fedelmente quel "Poker elettronico" presente in tutti i Casinò che dilapida le sostanze di molti giocatori. Ma tranquillizzatevi, voi non sarete realmente "spennati"! Il funzionamento è molto semplice: quando vi sarà richiesta la puntata, dovrete battere la somma che intendete giocare.

Qualora la puntata fosse di un numero con meno di tre cifre, premete ENTER dopo l'ultima cifra (esempio: 30 $ sara 30 seguito da ENTER).



Il calcolatore vi darà allora le carte e vi chiederà quante e quali vogliate cambiare; per fare ciò basterà semplicemente premere il numero corrispondente alla carta o alle carte da sostituire, che compare sopra le carte stesse.

Dopo aver indicato i numeri per la sostituzione, dovrete premere lo 0 per permettere al calcolatore di riordinare le carte secondo la nuova combinazione e, se sarete fortunati, di pagarvi la vincita.

Piuttosto semplice la strutturazione del programma dal punto di vista tecnico; la creazione delle carte è divisa in semplici print per i simboli grafici e in alcune centina di vpoke, posizionate da opportuni calcoli, per le figure; il tutto è corredato da delle Line, BF che danno la colorazione.

Osservando i data relativi alla loro definizione e gli innumerevoli calcoli eseguiti dagli anelli di For, ci si potrà rendere conto dell'enorme quantità di tempo richiesto per la loro realizzazione.

Grazie a ciò potrete però toccare con mano la grande potenzialità grafica del vostro MSX. ■

ASSEMBLATORE

(CLOAD "ASSEMB" - 32 K)

di D. Montresor

Un'utility davvero... utile che permette di caricare in memoria programmi in assembler con la sola limitazione di non poter usare le label. È scritta completamente in basic, quindi può essere modificata a seconda delle esigenze (utilizzo stampante o Quick-Disk). Dato il RUN appare la prima riga di programma, naturalmente vuota. Potete quindi cominciare a scrivere subito le istruzioni, oppure passare al menù premendo SELECT. I valori o indirizzi in esadecimale necessari per alcune istruzioni devono sempre avere: due cifre per valori ad un byte e quattro cifre per valori a due byte (non si possono tralasciare gli zero). Questi indirizzi possono anche essere in decimale, basta delimitare il numero tra due parentesi quadre. I numeri decimali ad un byte possono andare da —128 a 255, quelli a due byte da —32768 a 65535 (—1 corrisponde, in quest'ultimo caso, a 65535). Volendo usare numeri ottali come indirizzi o valori, si può far precedere il numero dal prefisso &0 sempre tra parentesi quadre. Nel caso l'istruzione scritta sia errata (numeri troppo grandi o istruzione non valida), essa viene cancellata oppure riappare la precedente istruzione già scritta in quella particolare riga. Se invece è giusta, vengono visualizzati i byte di codifica sulla sinistra dello schermo e si passa alla riga seguente.

Per cambiare una riga già scritta, premete il tasto SELECT senza scrivere nulla prima; scegliete l'opzione 1 del menù e quindi specificate il numero della linea da correggere. Correggete ora la linea visualizzata aiutandovi con il tasto di spostamento a sinistra del cursore. Premete dunque ENTER e ripetete tutto per tornare all'ultima riga del programma (riga vuota), che corrisponde al secondo numero tra parentesi della domanda "Linea di partenza". Finito il programma, premete SELECT per passare al menù. Le scelte possibili sono:

1) corregge o aggiunge linee in fondo al programma come già descritto;
2) visualizza tutto il listato: usate il tasto STOP per fermare lo scroll;
3) registra il programma sotto forma di file Ascii in modo da poterlo caricare in seguito nuovamente in memoria;
4) recupera il file senza però effettuare l'assemblaggio;

5) carica il linguaggio macchina in memoria a partire dalla locazione specificata e dopo aver ottenuto conferma (la memoria disponibile per il L/M va da 57000 a 62335). Di seguito si può registrare la routine con il normale BSAVE in modo da poterla caricare con altri programmi (tipo Monitor);
6) visualizza tutte le istruzioni dello Z80;
7) toglie righe al programma specificando il numero della prima riga e poi il numero totale delle righe da cancellare. Volendo, si può cancellare anche tutto il programma;
8) con questa opzione si aggiungono

```
E8,F8,E8,C8,90,F0
00,1F,1F,3F,3F,3F
00,80,E0,E0,F0,F0

1F,20,20,40,44,44
80,60,10,10,88,88

1F,20,20,40,50,50
80,60,10,10,08,08

00,07,1F,1F,3F,3F
00,E0,E0,F0,F0,F0

07,18,20,20,40,40
E0,10,10,08,28,28

07,18,20,20,40,40
E0,10,10,08,08,08

1F,20,40,84,84,84
```

delle righe contenenti dei NOP che possono poi essere modificati. Specificate il numero delle righe da aggiungere e poi la riga di partenza delle stesse. Non è però possibile aggiungere righe lasciandone alcune vuote tra la prima parte del programma e quella aggiunta.

La variabile DI alla riga 120 del programma fissa il numero massimo di righe dell'assemblatore. Il CLEAR che predispone lo spazio per le stringhe e quello per il linguaggio macchina si trovano alla linea 100. Nel caso questi valori vengano modificati è necessario usare le dovute cautele per evitare l'overflow. Se durante la stesura del listato assembler viene superato il massimo numero di linee, appare un apposito messaggio. Si può ovviare dividendo il programma in diverse parti. Gli MSX, quando operano con molte variabili stringa come in questo caso, devono effettuare ogni tanto l'operazione di garbage collection. Questa richiede più tempo se le linee del programma sono molte, quindi basta attendere che il cursore torni visibile.

SCAMBIO PROGRAMMI

(CLOAD "PROEXX" - 16 K)

di D. Montresor

Ecco una bellissima utility che permette di caricare nel computer due programmi differenti e di passare dall'uno all'altro senza doverli caricare nuovamente da cassetta.

Caricata l'utility, eseguite il RUN; caricate quindi il primo programma con CLOAD o LOAD, premete CLS (SHIFT+HOME) poi ESCAPE. Caricate ora il secondo programma come avete fatto con il primo quindi, per passare da un programma all'altro in qualsiasi momento, basterà che premiate i seguenti tasti:
CTRL + STOP
SCREEN 0
ENTER
ESCAPE
SCREEN 0
ENTER

Il programma principale in linguaggio macchina si trova alla locazione 63860, quindi non occupa memoria perché risiede in una zona dedicata ad eventuali cartucce di espansione. Attenzione: nel caso non vengano eseguite correttamente le istruzioni indicate sopra, possono sorgere diversi problemi in quanto la memoria video resta invariata per ambedue i programmi. Se si fanno girare programmi che scrivono direttamente nella memoria (con POKE) o che sono troppo lunghi può esserci un tilt!

UTILITY

L'ISTRUZIONE INPUT IN SCHERMO GRAFICO

UNA ROUTINE PER INTRODURRE DA TASTIERA STRINGHE DI PIÙ DI UN CARATTERE: BASTA CON IL RITORNO IN TEXT MODE!

di G. RICCOBONO

I computers MSX posseggono la noiosa caratteristica di ritornare in "text mode" (SCREEN 0) dallo schermo grafico ogni volta che l'istruzione INPUT viene eseguita.

Un modo per aggirare questo ostacolo è quello di utilizzare dei cicli facenti capo a un'istruzione INKEY$. Supponendo di avere ad esempio la necessità di introdurre un carattere che sia una lettera maiuscola oppure uno spazio, possiamo a tal fine sfruttare un ciclo del tipo:

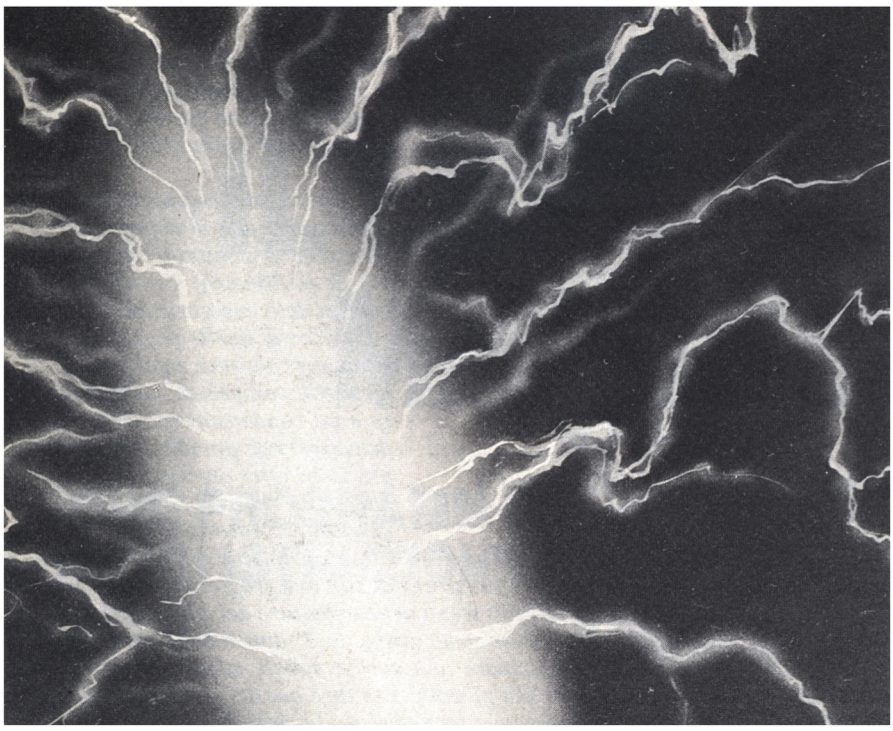
```
100 A$="":A$ = INKEY$
110 IF (A$ <CHR $ (65) OR A$ (90))
AND A$ <>CHR$ (32) THEN 100
```

Dalla riga 110 in poi avremo conservato, nella variabile A$, il carattere richiesto dal nostro programma e da noi introdotto da tastiera. Come si nota però immediatamente, sorgono in questo caso due tipi di inconveniente: prima di tutto diventa lungo e tedioso l'inserimento di stringhe composte da più di un carattere e, secondariamente, non è agevole la correzione di eventuali errori di battitura tramite l'uso del tasto "Back Space". Vi proponiamo allora una piccola routine che, inserita nei vostri programmi, vi permetterà di introdurre da tastiera stringhe di più di un carattere, consentendo contemporaneamente la correzioni di eventuali errori di battitura tramite l'uso del tasto "Back Space". La routine qui presentata, e riportata nel listino numero uno, consente nella fattispecie l'inserimento di stringhe di numeri composti di più cifre e dà la possibiltà di gestire la stringa introdotta sia come stringa che come numero; è però più che ovvia l'estensione a stringhe alfabetiche, ma di questo argomento parleremo comunque fra poco.

Veniamo ora al commento ed all'analisi del listato, in modo da studiarne insieme le modalità di impiego.

Prima di tutto la routine richiede l'uso di alcune variabili che vanno assegnate prima che venga eseguita la chiamata.
Tali variabili sono:
BAC: colore di fondo richiesto;
FC: colore di primo piano richiesto;

```
1000 '-------- Listato 1 ----------
1010 OPEN"GRP:" AS#1
1020 A$=""
1030 A$=INKEY$
1040 IF A$="" THEN 1030
1050 A=ASC(A$)
1060 IF (A<48 OR A>57) AND A<>13 AND A<>8 THEN 1020
1070 IF A=8 AND ST$="" THEN 1020
1080 IF A=8 THEN BS=1
1090 IF A=13 THEN CLOSE#1:ST=VAL(ST$):RETURN
1100 IF BS=0 THEN ST$=ST$+A$:PRESET(X,Y):COLOR FC:PRINT#1,ST$
1110 IF BS=1 THEN BS=0:COLOR BAC:PRESET(X,Y):PRINT#1,ST$:ST$=LEFT$
(ST$,LEN(ST$)-1):COLOR FC:PRESET(X,Y):PRINT#1,ST$
1120 GOTO 1020
```



X, Y: posizione sullo schermo grafico alla quale desiderate che il primo carattere della stringa introdotta appaia.

Alla riga 1010 viene aperto un file di uscita per la stampa di caratteri sul video in schermo grafico.

Le righe 1020-1040 leggono la tastiera; se un tasto viene schiacciato, il carattere relativo a quel tasto viene conservato nella variabile A$.

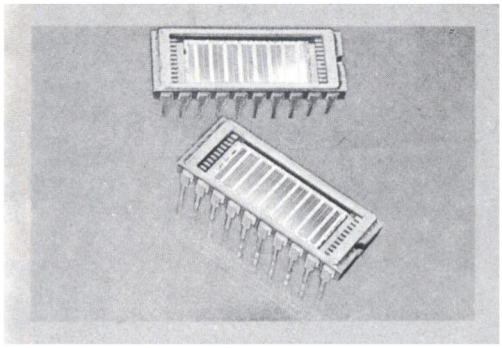
La riga 1050 conserva nella variabile A il codice di carattere del carattere A$, mentre la riga 1060 controlla che il carattere introdotto sia un numero (48 è il codice ASCII del numero 0 mentre 57 è il codice ASCII del numero 9), oppure il tasto "Return" (codice ASCII 13), oppure il tasto "Back Space" (codice ASCII 8); se il tasto schiacciato non è né un numero, né il "Return", né il "Back Space", allora il carattere introdotto viene rifiutato, in quanto è supposto sia dovuto ad un errore di battitura ed il controllo viene rimandato alla riga 1020 per la lettura da tastiera di un altro carattere (ovviamente ancora numerico). Se voi, anziché numeri, siete interessati ad introdurre lettere alfabetiche, dovrete dunque cambiare la riga 1060 con la seguente:

1060 IF (A<65 OR A>90) AND A<>13 AND A<>8 THEN 1020

Se volete invece introdurre stringhe alfanumeriche (cioè composte sia da numeri che da lettere alfabetiche) dovrete battere nel vostro listato la riga:

```
5  '-------- Listato 2 ---------
10 INPUT "BAC";BAC
20 INPUT "FC";FC
30 INPUT "X";X
40 INPUT "Y";Y
50 COLOR FC,BAC,BAC:SCREEN 2
60 GOSUB 1000
70 END
```

I listati accanto riprodotti consentono di introdurre delle stringhe di testo con la funzione Input senza che il computer torni necessariamente nella condizione SCREEN 0. Intervenendo sulle condizioni poste alla linea 1060 è possibile stabilire se i caratteri dati in Input debbano essere numerici o alfanumerici.
La 1070 determina il salto alla routine di correzione se viene premuto il tasto Back Space.



UNA RAM PER IL FUTURO

Toshiba ha realizzato la prima DRAM (Memoria dinamica ad accesso casuale) da 4 MBit, presentandone il prototipo alla Conferenza internazionale sui Circuiti a Stato Solido (ISSCC) tenuta in California.
4 M bit (4 M byte mettendo in parallelo tanti chip quanti sono i bit nel byte caratteristico del computer) sono una dimensione da fantascienza. Solo ieri l'altro abbiamo avuto le RAM su chip da 64 K. Solo ieri o meglio oggi abbiamo le RAM da 256 K che pure hanno posto tanti problemi per la loro messa a punto. Adesso si parla molto timidamente, della RAM da 1 M, ed ecco Toshiba che rilancia, quadruplicando. 4 M vuol dire una memoria 16 volte più capace della 256 K, addirittura 64 volte più capace della classica "64".
Come è stato possibile arrivare al prodigio Toshiba, che vanta, oltre a tutto un tempo di accesso di 80 n sec, un consumo in funzionamento di 300 mW (in Standby si scende a soli 2,5 mW) e una grandissima affidabilità?
La DRAM di Toshiba raggruppa più di 8,7 milioni di componenti discreti su una superficie di 137 mm^2. I problemi da risolvere per sistemare questo incredibile numero di unità erano grandissimi.
Occorreva una capacità fotolitografica in grado di definire caratteristiche di 1 micron (un millesimo di millimetro) contro gli 1,2 micron delle memorie da 1 M e gli 1,8 micro di quelle da 256 K. Come inoltre saprete, ogni "cella" di memoria comprende un transistore e un condensatore. Ora i condensatori non possono essere rimpiccioliti oltre un certo limite perché se ne ridurrebbe la capacità, rendendo aleatorio il loro funzionamento. Toshiba ha risolto il problema scavando dei microalveoli nel substrato di silicio e trasformando le pareti dei microalveoli stessi nelle armature del condensatore. Ha sfruttato, in un certo senso, la 3ª dimensione consentendo di immagazzinare una carica uguale su una superficie di chip molto più piccola. Per fare questo Toshiba ha dovuto sviluppare nuove tecniche per la diffusione delle impurità nelle pareti dei microalveoli.

1060 IF (A<65 OR A>90) AND (A<48 OR A>57) AND A<>13 AND A<>8 THEN 1020

La riga 1070 fà in modo che, se il primo tasto schiacciato è un Back Space, il controllo venga rimandato alla riga di lettura della tastiera in quanto anche questo è considerato un errore di battitura (è impossibile cancellare qualcosa se ancora niente è stato introdotto!).

La riga 1080 utilizza la variabile BS come "flag", cioè come "segnalino"; in altri termini la variabile BS viene posta uguale ad 1 se il tasto Back Space è stato schiacciato, altrimenti viene lasciata pari al valore iniziale (cioè 0).

Questo utilizzo, tipico delle variabili in qualità di flag, serve a ricordare, in un successivo momento, se un determinato tasto è stato schiacciato, in modo da poterci occupare al presente di altri problemi, per poi ritornare al significato di quel tasto in un secondo tempo.

La riga 1090 controlla se il tasto che era stato schiacciato era il tasto "Return"; se così è, vuol dire che la stringa che volevamo introdurre è stata completata. Il file di scrittura viene allora chiuso, il valore numerico della stringa viene conservato nella variabile ST sulla quale potremo dunque svolgere delle operazioni matematiche ed il controllo viene infine rimandato alla riga successiva a quella di chiamata tramite la classica istruzione RETURN. È meglio, a tale proposito, fare due considerazioni: prima di tutto, nel caso che la stringa da noi introdotta non sia puramente numerica, oltre a cambiare opportunamente la riga 1060 dovremo togliere l'istruzione ST=VAL(ST$), presente alla riga in questione; se poi la chiamata della routine qui presentata viene da voi svolta tramite un'istruzione GOTO anziché tramite un'istruzione GOSUB (che noi in ogni caso consigliamo), allora anche l'istruzione RETURN, sempre presente alla riga 1090, và tolta e sostituita con un GOTO che rimandi alla riga da voi desiderata.

La riga 1100 controlla il contenuto della flag BS; se tale contenuto è 0, allora il tasto BS non è stato schiacciato. D'altra parte, neanche il tasto Return è stato schiacciato, altrimenti la precedente riga 1090 sarebbe diventata attiva e avrebbe fatto ritornare il controllo al programma principale; ciò vuol dire che il tasto schiacciato è per forza uno di quelli da noi voluti (nel caso da noi presentato, un numero fra 0 e 9). Tale carattere viene allora sommato a quelli da noi introdotti precedentemente e già presenti nella stringa ST$, dopodiché la stringa ST$ viene stampata col colore di primo piano da noi scelto (e memorizzato nella variabile FC) e nella posizione voluta (e memorizzata nelle coordinate X,Y).

La riga 1110 controlla se il flag BS è posto uguale a 1: in caso affermativo il tasto BS è stato schiacciato e l'ultimo carattere utile (cioè numerico) introdotto và cancellato dalla stringa ST$; sullo schermo ciò viene effettuato sovrapponendo alla scritta relativa alla stringa ST$ la stessa scritta in colore di sfondo (colore che è conservato nella variabile BAC); la stringa ST$ viene poi posta pari al suo contenuto precedente meno l'ultimo carattere e viene poi ristampata in colore di primo piano. La riga 1120 rimanda infine alla riga 1020 per l'introduzione del successivo carattere.

Concludiamo le considerazioni sull'input in schermo grafico con un programmino atto a provare la routine proposta. Tale programmino, presentato nel listato numero 2, permette l'inserimento da tastiera dei colori di sfondo e di primo piano desiderati e della posizione alla quale volete far apparire la stringa introdotta. Nel caso che la routine del listato 1 non sia da voi digitata partendo dalla riga 1000, ricordate di sostituire la riga 60 del programmino di prova con il GOSUB opportuno.

■

Ⓐ Ⓢ Ⓢ Ⓔ Ⓜ Ⓑ Ⓛ Ⓔ Ⓡ

IL LINGUAGGIO MACCHINA

COME PROGRAMMARE IN LINGUAGGIO MACCHINA. I CODICI ISTRUZIONE DEL MICROPROCESSORE Z80 A

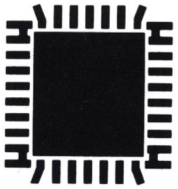
(6ª PUNTATA)

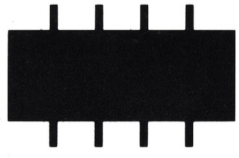
di EMANUELE DASSI

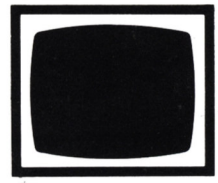
Concludiamo con questo articolo la descrizione dell'intero set d'istruzioni dello Z80 parlando di quei comandi dedicati alla comunicazione tra il microprocessore ed i circuiti collegati ad esso: le istruzioni di input/output e di interrupt.

LE ISTRUZIONI DI INPUT/OUTPUT

Per comprendere pienamente il significato ed il funzionamento di queste istruzioni è opportuno rivedere alcuni concetti hardware della CPU Z80. Lo Z80, oltre che "dialogare" con la memoria, può essere interfacciato con altri dispositivi esterni quali per esempio la tastiera, una stampante, il video ecc. Per poter comunicare fisicamente con questi dispositivi il microprocessore utilizza quei suoi pin appositamente dedicati a tale scopo e, per quello che a noi più interessa, delle istruzioni che agiscono su questi pin modificando i loro segnali. Così come nel caso della memoria è possibile indirizzare fino a 65536 locazioni distinte, anche per i dispositivi collegati alla CPU c'è un massimo di 256 unità collegabili. Questo vuol dire che il bus d'indirizzamento non sarà formato da 16 linee ma da 8







linee (2^8=256). Difatti l'indirizzamento di un dispositivo viene posto sulle otto linee meno significative del bus indirizzi, cioè da A0-A7 (con la sigla A s'intende una linea del bus indirizzi).

Le istruzioni che generano impulsi di sincronizzazione per dispositivi esterni diversi dalla memoria vengono chiamate istruzioni di input e di output.

Le istruzioni di input sono: IN A,(n), IN r,(C), INI, INIR, IND e INDR. Vediamo di commentarle una ad una.

L'istruzione IN A,(n) riceve nell'accumulatore il dato (byte) proveniente dal dispositivo il cui indirizzo è n, compreso tra 0 e 255. L'istruzione pone il valore di n nella parte meno significativa del bus indirizzi (A0-A7) ed il valore dell'accumulatore, non ancora modificato dalla lettura, nella parte alta del bus indirizzi (A8-A15). Questa istruzione non modifica lo stato dei flags.

IN r,(C) (dove r può essere il registro A,B,C,D,E,H o L) produce l'effetto dell'istruzione precedente con la sola differenza che l'indirizzo è contenuto nel registro C e che la lettura è memorizzata nel registro r. L'istruzione IN r, (C) setta il flag di zero (Z), di parità (P/V) e di segno (S) secondo i risultati dell'operazione.

Lo Z80 supporta le istruzioni di input di blocchi di dati in modo analogo alle istruzioni di spostamento e di ricerca blocchi considerate nella scorsa puntata; è per questo che i registri B, C ed HL vanno inizializzati con i seguenti contenuti:

B= numero di byte da leggere
C= indirizzo del dispositivo da leggere
HL= indirizzo d'inizio dove depositare i byte letti

L'istruzione INI legge il byte dal dispositivo collegato all'indirizzo (C), lo memorizza nella locazione (HL), decrementa il valore di B ed incrementa il registro HL. Tale istruzione modifica il flag zero (Z) ponendolo ad 1 se il valore di B è pari a 0. L'istruzione INIR lavora come la precedente ma ripete la lettura, e quindi l'incremento di HL, fino a che il registro B raggiunge il valore 0. Infine le

SERVE LA STAMPANTE?

Sono in molti a ritenere che NL10, la nuova stampante presentata dalla giapponese Star Micronics (leader indiscusso nel mondo delle stampanti, rappresentato in Italia da Clairtron S.p.A.), costituirà ben presto il nuovo standard cui dovranno riferirsi i settori dell'home computer e del personal computer "Low-end".

È sorprendente, su questa macchina, la quantità di funzioni di stampa controllabile dall'utilizzatore tramite un pannello frontale che consente operazioni di solito riservate a macchine di categoria di gran lunga superiore.

Equipaggiata con una testina a 9 aghi che stampa a 120 caratteri al secondo su 80 colonne, (30 caratteri al secondo in near letter quality), l'autentica "rivoluzione" di questa printer è costituita dalla concezione assolutamente nuova della zona d'interfaccia. Grazie ad una nuova architettura dell'elettronica di stampa è possibile connettere alla nuovissima NL10 una serie di "cartridge" (cartucce) d'interfaccia che vanno dalla parallela Centronics (Epson compatibile), alla parallela PC IBM, alla Commodore C64 e C128. A queste già disponibili, si affiancheranno l'interfaccia RS232C, l'Apple II C e l'Apple Macintosh.

istruzioni IND e INDR svolgono rispettivamente la funzione di INI ed INIR con la sola differenza che, anziché incrementare il valore del registro HL, lo decrementano (la sigla "D" in IND e INDR sta infatti ad indicare la parola Decrement).

INPUT OUTPUT ADDRESS

INDIRIZZO	DISPOSITIVO
00H-3FH	NON SPECIFICATO
40H-7FH	RISERVATO
80H-87H	INTERFACCIA RS-232C
88H-8FH	LIBERO
90H-97H	INTERFACCIA STAMPANTE
98H-9FH	VDP
A0H-A7H	PSG
A8H-AFH	PPI
B0H-B3H	MEMORIA ESTERNA
B4H	OROLOGIO-CALENDARIO
B5H-B7H	LIBERO
B8H-BFH	INTERFACCIA LIGHT PEN
C0H-CFH	LIBERO
D0H-D7H	INTERFACCIA FLOPPY DISK
D8H-DFH	ROM CARATTERI CINESI
E0H-F6H	LIBERO
F7H	CONTROLLER AUDIO/VIDEO
F8H-FFH	LIBERO

FIG. 1 - Tabella degli indirizzi dei dispositivi di I/0

Parallelamente alle istruzioni di input troviamo quelle di output, il cui comportamento è identico a quello delle prime cambiando solo il flusso dei dati; questa volta, cioè, è la CPU che li invia al dispositivo anziché riceverli. Quindi abbiamo OUT (n),A, OUT (C),r, OUTI, OTIR, OUTD e OTDR.

Per quanto riguarda gli indirizzi va detto che, a parte le istruzioni IN A, (n) e OUT (n),A, il contenuto del registro C viaggia, sia in input che in output, tra i pin A0 e A7 del bus indirizzi mentre tra A8 e A15 si trova sempre il valore del registro B.

Vediamo ora di fare qualche considerazione su queste istruzioni. Innanzitutto va specificato che sono istruzioni molto "preziose", che non tutti i microprocessori hanno e che permettono, sia come software che come hardware, di distinguere una locazione di memoria da una dispositivo. Poi va riconosciuta la quantità d'istruzioni dedicate all'input/output con le quali è possibile far transitare addirittura un flusso di dati con una sola istruzione.

Per usare queste istruzioni non bisogna però soltanto conoscere il software del sistema sul quale si lavora ma occorre soprattutto conoscere l'hardware. Per questa ragione abbiamo voluto pubblicare la tabella di figura 1 che riassume gli indirizzi assegnati allo standard MSX per il collegamento di interfacce e dispositivi vari.

LE ISTRUZIONI DI INTERRUPT

L'interrupt (interruzione) è un segnale inviato alla CPU da parte di un dispositivo esterno per segnalare la necessità di svolgere un servizio prioritario. Supponiamo per esempio che lo Z80 stia eseguendo una lunga elaborazione e che improvvisamente l'utente voglia sospendere tale lavoro: per far ciò egli utilizza la tastiera premendo il tasto break. La tastiera è un dispositivo periferico della CPU che, per avvertire lo Z80 che deve sospendere ogni lavoro per dedicarsi ad uno particolare (attivato col tasto break), invia un segnale al microprocessore: l'interrupt.

Vi sono due tipi di interruzioni, quelle non mascherabili

MUSIC MASTER

Dalla Philips una novità (buona per MSX1 e 2) per gli appassionati di musica. Un'interfaccia MIDI che può essere utilizzata con qualsiasi computer MSX collegato con un televisore o monitor senza nessun cavo altro che quello d'antenna.
Ingressi/uscite: microfono esterno con controllo di volume - Linea - Audio out (2 mono) - MIDI IN - MIDI OUT - MIDI TROUGH - Ingresso tastiera.
Microfono incorporato - Microprocessore sonoro FM-Y8950.
Indicazioni sullo schermo:
Scrittura musicale delle note in chiave di violino e chiave di basso. Indicazioni sullo schermo: annotazioni musicali (note diesis, bemolle, pause, etc.). Indicazioni sullo schermo: "menù" di riffs musicali, dell'uso di tastiere esterne, di sonorità diverse, di campionamento sonoro e di melodie.
Sintetizzatore sonoro FM: 60 voci predeterminate: archi, fiati, piano, xylofono, organo, chitarra, basso, cosmic wow, treno, etc.
Altre sonorità come per es.: tuoni, spari, onde, etc. disponibili su cassetta audio, per il campionamento.
20 accompagnamenti: 4 voci d'accompagnamento - 1 sequenza ritmica - 1 accompagnamento di basso.
10 accompagnamenti ritmici richiamabili anche solo.

e quelle mascherabili.

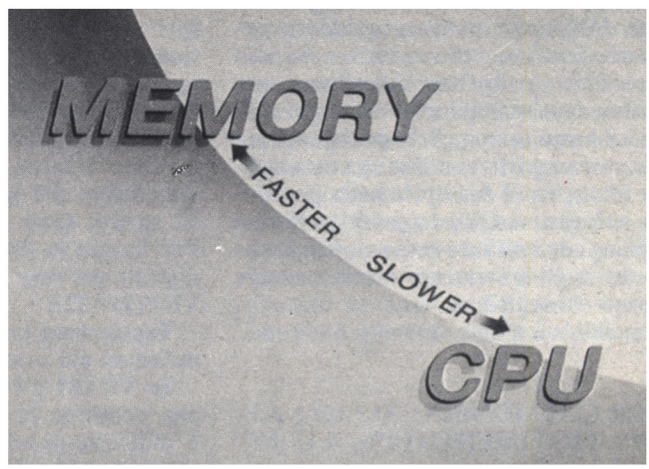
L'interruzione non mascherabile non può essere ignorata dalla CPU ed è sempre attiva perché dipende esclusivamente dal segnale presente sul pin NMI dello Z80; quando tale pin è a livello logico 0 (assenza di segnale) lo Z80 sospende l'elaborazione in corso ed esegue una restart (cioè una istruzione di CALL) all'indirizzo 66H dove risiede la subroutine di gestione dell'interruzione, finita la quale l'elaborazione riprende da dove era rimasta sospesa al momento dell'interrupt. Per quanto riguarda l'MSX questo tipo di interruzione non è implementata perché l'indirizzo 66H della ROM è utilizzato dall'MSX-DOS. L'interruzione mascherabile invece è attivabile o meno via software. Dal punto di vista hardware l'interruzione mascherabile agisce non più sul pin NMI ma sul pin INT, anch'esso attivo quando si trova a livello logico 0.

IL MASCHERAMENTO VIA SOFTWARE

Dicevamo che l'interruzione mascherabile è attivabile o meno via software: l'istruzione DI (Disable Interrupt) disabilita infatti il riconoscimento dell'interruzione, mentre l'istruzione EI (Enable Interrupt) lo riabilita. A questo punto però il discorso non è finito perché, come se non bastasse, lo Z80 prevede ben tre tipi diversi di interruzioni mascherabili, ognuno di essi programmabile via software tramite le istruzioni IM0, IM1 e IM2.

Utilizzando l'istruzione IM0 sul bus dati il dispositivo porrà l'istruzione che lo Z80 dovrà eseguire. Essendo l'informazione ad un byte, l'istruzione è una delle otto RST (restart) che provoca il salto all'indirizzo dove risiede la subroutine di gestione dell'interruzione.

IM1, invece, fa eseguire alla CPU soltanto l'istruzione RST38H. Infine, IM2 rappresenta l'interruzione più complessa, ovvero quella nella quale l'indirizzo dove saltare è più difficile da ricostruire. Entra in gioco infatti il registro I (Interrupt), forse da molti dimenticato. Questo registro, ad 8 bit, può essere modificato e letto con le istruzioni LD I,A ed LD A,I. Quando viene eseguita una interruzione in modo 2 (IM2), la CPU ricostruisce l'indirizzo al quale saltare nel modo seguente: legge il byte sul bus dati impostato dal dispositivo e lo utilizza, come parte meno significativa, insieme al valore del registro I, come parte più significativa, per ricostruire un indirizzo a 16 bit puntante ad una locazione di memoria dove prelevare l'indirizzo (sempre a 16 bit) di prosecuzione del programma.



Sì, la cosa è un po' complessa, ma è anche vero che si tratta dell'interruzione più potente.

Per quanto riguarda l'MSX lo standard utilizza come interruzione mascherabile il modo 1 (IM1, RST38H), attiva ogni cinquantesimo di secondo perché generata dal VDP per il sistema di visualizzazione delle immagini PAL.

Termina qui il nostro viaggio alla scoperta dell'Assembler Z80: avremo modo di riprendere gli argomenti trattati fino adesso in altre occasioni.

LA GESTIONE DELLE SPRITES

NEL LABIRINTO DELLE SPRITES: DEFINIZIONI E METODI PER LA GESTIONE AVANZATA IN LINGUAGGIO MACCHINA.

a cura di G. RICCOBONO

Una delle maggiori caratteristiche dello standard MSX è costituito dalle sprites. Ma cosa sono esattamente le sprites e come possiamo impostare una loro gestione avanzata in Linguaggio Macchina?

Supponiamo di scrivere un gioco in cui dobbiamo muovere qualcosa armoniosamente sullo schermo. Se non avessimo le sprites dovremmo per prima cosa stampare un carattere sullo schermo in una certa posizione, poi sovrastamparlo con uno spazio e infine stampare il carattere nella posizione successiva. Questo processo risulta essere non soltanto tedioso, ma anche lento e il movimento risulta essere rozzo e a scatti.

PER L'ARMONIA DEL MOVIMENTO

Grazie all'uso delle sprites, muovere oggetti sullo schermo è molto più semplice e il movimento stesso risulta molto più armonioso. Immaginate una sprite come un normale carattere: si possono definire la sua forma e il suo colore, in aggiunta però a queste normali caratteristiche dei caratteri, le sprites godono della proprietà che non sono confinate all'essere stampate dentro una cella di carattere e, inoltre, non è necessario cancellarle prima di muoverle. Così, muovere una sprite col BASIC o col Linguaggio Macchina, implica il semplice cambiamento di una cella di memoria.

È possibile avere sullo schermo fino a 32 sprites contemporaneamente (anche se esiste a questo fatto una limitazione che vedremo più avanti). Le sprites possono avere 4 diverse dimensioni, ma non è possibile avere contemporaneamente sprites di diverse dimensioni sullo schermo. Negli esempi successivi useremo le sprites di formato ridotto, che hanno le stesse dimensioni dei normali caratteri. Le 32 sprites sono numerate da 0 a 31. Per fissare la dimensione della sprite digitate per ora:

VDP(1)=224

Tratteremo insieme le altre tre dimensioni più avanti.

In VRAM c'è un'area di memoria che contiene la forma delle sprites. L'indirizzo di inizio di quest'area può essere trovato digitando l'istruzione:

PRINT BASE(9)

Nel mio MSX quest'indirizzo è 14336. Chiameremo d'ora in poi quest'indirizzo INIZIO FORMA SPRITE. Come per i normali caratteri, la forma delle sprites formato piccolo è contenuta in otto bytes. Nel mio MSX gli otto bytes di dati per la sprite numero 0 saranno quelli dal 14336 al 14343. Riempiremo ora questi otto bytes con dei dati rappresentanti un UFO (vedi Fig. 1).

È inverosimile che qualcosa sia apparsa sullo schermo, questo perché non abbiamo ancora definito dove vogliamo che la sprite appaia. Nella VRAM c'è un'area di memoria che contiene le informazioni riguardanti la posizione e il colore di ogni sprite. Ognuna delle 32 sprites ha quattro variabili ad essa associate:

1) Posizione verticale
2) Posizione orizzontale
3) Numero della sprite
4) Colore

CON QUALI ISTRUZIONI

L'indirizzo di inizio di queste quattro variabili può essere trovato tramite l'istruzione:

PRINT BASE(B)

Nel mio MSX questo indirizzo di inizio è 6912. Nel seguito chiameremo questo indirizzo "INIZIO VAR DI SPRITE". L'indirizzo di inizio delle quattro variabili può essere trovato, per ogni singola sprite, svolgendo il seguente calcolo:

('NUMERO DI SPRITE'*4)+'INIZIO VAR DI SPRITE'

Dove al posto di "INIZIO VAR DI SPRITE" dovete mettere l'indirizzo da voi trovato attraverso l'istruzione più sopra menzionata e al posto di "NUMERO DI SPRITE" dovete mettere il numero con cui avete chiamato la sprite in questione. Così, nel

QUALCHE SIMPATICO ESEMPIO

```
VPOKE 'INIZIO FORMA SPRITE' +0,0
VPOKE 'INIZIO FORMA SPRITE' +1,24
VPOKE 'INIZIO FORMA SPRITE' +2,36
VPOKE 'INIZIO FORMA SPRITE' +3,66
VPOKE 'INIZIO FORMA SPRITE' +4,255
VPOKE 'INIZIO FORMA SPRITE' +5,126
VPOKE 'INIZIO FORMA SPRITE' +6,36
VPOKE 'INIZIO FORMA SPRITE' +7,66
```

FIG. 1

```
VPOKE('INIZIO VAR DI SPRITE'+0),104
VPOKE('INIZIO VAR DI SPRITE'+1),50
VPOKE('INIZIO VAR DI SPRITE'+2),0
VPOKE('INIZIO VAR DI SPRITE'+3),10
```

FIG. 2

Gli otto bytes (vedi anche nel testo) sono stati da noi riempiti con dei dati rappresentanti un UFO. Naturalmente il disegno non è visibile sullo schermo. Perché questo appaia dovete porre gli indirizzi relativi alle quattro singole variabili (inizio VAR di Sprite). Qui sopra trovate un esempio pratico della procedura da adottare perché il piccolo UFO possa essere visualizzato sul display monitor o sul TV.

mio MSX, la prima delle quattro variabili relative alla sprite numero 15 è: (15*4)+6912

Se voi inserite dunque le quattro istruzioni seguenti, la sprite che abbiamo disegnato apparirà sullo schermo (Fig. 2).

Le posizioni orizzontale e verticale si riferiscono a dove appare l'angolo in alto a sinistra della sprite. A tale proposito ricordiamo che per posizionare la sprite dovete considerare lo schermo come suddiviso in 256 posizioni orizzontali (numerate da 0 a 255) e in 192 posizioni verticali (numerate da 0 a 191), come del resto avviene in BASIC. Assumendo quindi, come noi abbiamo fatto nel nostro esempio, una posizione verticale di 100 e una orizzontale di 50, dovremo vedere apparire la sprite più o meno nel centro dello schermo per quanto riguarda l'alto-basso e a un quinto dello schermo per quanto riguarda destra-sinistra. Provate a questo punto a usare l'istruzione VPOKE per cambiare la posizione orizzontale e quella verticale diverse volte, usando numeri sempre differenti, in modo da prendere pratica sul posizionamento della sprite sullo schermo. Provate anche a fare il VPOKE della posizione verticale con un numero più grande di 191, vedrete la sprite scomparire dallo schermo. All'inizio di questo articolo abbiamo menzionato come sia possibile avere 32 sprites sullo schermo, ma abbiamo anche detto che esiste una limitazione, alla quale dovrete stare particolarmente attenti scrivendo programmi di gioco. Non è infatti possibile visualizzare su una stessa linea orizzontale più di 4 sprites. Se per sbaglio, dunque, posizionate 5 o più sprites su una stessa linea, la quinta e le successive sprites scompariranno. Questo avviene anche per le sprites che sono allineate solo parzialmente.

LA COLLISIONE DELLE SPRITES

Associato con le sprites esiste un registro che può essere letto per vedere se due sprites hanno colliso. Il contenuto di questo registro è normalmente 1 mentre, ogni volta che due sprites si sovrappongono, il valore del registro passa automaticamente a 0. Per testare il contenuto del registro in questione e passare il controllo del programma alla routine interessata quando le sprites hanno colliso, dovete usare due istruzioni del tipo:

CALL 318 BIT 5,A

ANCORA PER PROVA

```
10 FOR A=0 TO 31
20 READ B
30 VPOKE('INIZIO FORMA SPRITE'+A),B
40 NEXT A
50 STOP
60 DATA 255,129,129,129,129,129,129,255
70 DATA 255,195,165,153,153,165,195,255
80 DATA 255,153,153,255,255,153,153,255
90 DATA 231,165,255,36,36,255,165,231
```

FIG. 3

```
SCREEN 1
VDP(1)=226
VPOKE('INIZIO VAR DI SPRITE'+0),100
VPOKE('INIZIO VAR DI SPRITE'+1),50
VPOKE('INIZIO VAR DI SPRITE'+2),0
VPOKE('INIZIO VAR DI SPRITE'+3),10
```

FIG. 4

Se il risultato di quest'ultima istruzione è zero, allora due sprites sono sovrapposte. Per una più approfondita conoscenza delle istruzioni Assembler "CALL" e "BIT" rimandiamo alla rubrica sull'Assembler dello Z80 pubblicata sul numero 6 del Gennaio '86 e successivi; oppure a un libro che si occupi specificatamente dell'Assembler dello Z80.

Da ultimo occupiamoci di vedere come ottenere e gestire le sprites degli altri tre formati, finora tralasciati. Digitando l'istruzione BASIC:
VDP(1)=224
si ottengono le sprites del formato 8x8 finora trattato. Digitando l'istruzione:
VDP(1)=225
e usando le stesse VPOKE degli esempi precedenti, vedrete apparire la stessa sprite in dimensione raddoppiata, questo è il secondo formato di sprite. Cambiate la posizione orizzontale e quella verticale usando l'istruzione VPOKE agli stessi indirizzi precedenti.

Il successivo formato di sprite è in realtà un insieme di 4 sprites, collegate a quadrato, con la sprite 0 in alto a sinistra, la sprite 1 in basso a sinistra, la sprite 2 in alto a destra e infine la sprite 3 in basso a destra. Per determinare la posizione del gruppo di 4 sprites, basta effettuare il VPOKE delle variabili della prima sprite (quella in alto a sinistra per intenderci) e automaticamente le altre tre sprites si sposteranno mantenendosi solidali a questa.

Provate a usare il (vedi Fig. 3) programma BASIC per definire la forma delle quattro sprites.

Digitate poi le 6 istruzioni di Fig. 4 di cui la seconda è quella che seleziona "l'incollaggio" delle sprites quattro a quattro.

QUAL È L'EFFETTO GLOBALE

Provate ancora a effettuare il VPOKE della posizione orizzontale e verticale (terza e quarta delle istruzioni appena elencate) per vedere qual è l'effetto globale.

Dunque, anche se di fatto in questo formato 4 sprites sono incollate insieme, esse sono viste come un'unica sprite. È possibile avere 32 di questi gruppi sullo schermo. Le quattro variabili relative a ogni gruppo sono calcolate in maniera differente rispetto a quanto avveniva nel formato 8x8. Usate la seguente formula per calcolare l'indirizzo iniziale delle quattro variabili di ogni gruppo:
('NUMERO DI SPRITE'*16)+
'INIZIO VAR DI SPRITE'

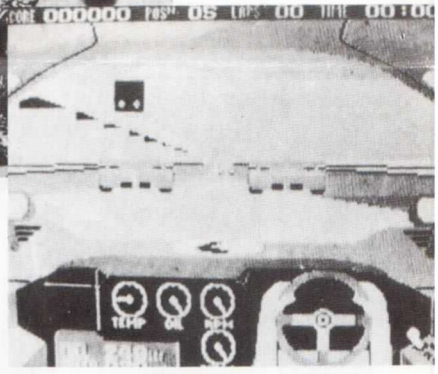
Le sprites anche in questo formato sono numerate da 0 a 31.

L'ultimo formato è identico a quello appena trattato eccetto che le sprites sono raddoppiate in dimensione. Tale formato può essere ottenuto digitando:
VDP(1)=227.

■

una vasta g

tutti in ec



GIOCHI
UTILITY
LINGUAGGI

IL MEGLIO PER IL TUO COMPUTER

PER LE NOVITÀ
PUOI ANCHE TELEFONARE 0422/671167

software MSX

UNA BUONA OCCASIONE PER IL SOFTWARE CHE ABBIAMO GIÀ PUBBLICATO. DUE CASSETTE INSIEME A PREZZO SPECIALE!

La tua copia subito a casa!
Richiedila, con vaglia postale di lire 12 mila, ad Arcadia, C.so Vitt. Emanuele 15, Milano.

SPLENDIDO

UN LOOK
COLORATO
PER LA TUA
CASSETTA

RITAGLIA
LUNGO
IL BORDO
SEGNATO
IN NERO
E
PIEGA
SEGUENDO
IL
TRATTEGGIO
INDICATO

PERSONALIZZA
LA
CASSETTA
CON IL
TUO NOME

MSX COMPUTER
MAGAZINE
PER LA TUA
SOFT-TECA

RIVISTA PROGRAMMI PER MSX

MSX Computer Magazine 11

Per caricare digitare CLOAD

SOMMARIO

LATO A:

- **TENNIS**
- **DRAGON BUSTER**
- **SLALOM**
- **CORSO DI INGLESE**
- **PARA**

LATO B:

- **HELLCOPTER**
- **SWORD AND SPELL**
- **POKER**
- **ASSEMBLER**
- **SCAMBIO PROGRAMMI**

QUESTA CASSETTA È DI

NOME _____

COGNOME _____

VIA _____ N. _____

CITTÀ _____

MSX COMPUTER MAGAZINE