Phonola

VG 8000 HOME COMPUTER

MANUALE D'USO

MSX

Phonola

VG 8000

HOME COMPUTER
MANUALE D'USO

MSX

Indice

Introduzione

Il VG8000 è un moderno e versatile home computer che può essere usato per imparare a scrivere i propri programmi, dai più semplici fino ai più complessi. È inoltre possibile usarlo per eseguire numerosi programmi "pronti" disponibili in commercio.

Il VG8000 è conforme allo standard più recente definito per gli home computers: lo standard MSX. Sotto questo aspetto, VG8000 offre un importante vantaggio e cioè la possibilità di usare qualsiasi programma disponibile nell'estesa libreria MSX. La rispondenza a questo standard significa inoltre che l'utente ha a sua disposizione un'ampia scelta di periferiche, ad esempio stampanti, unità disco, ecc. perfettamente compatibili col VG8000.

Il VG8000 incorpora un interprete MSX-BASIC, identico a quello di tutti gli altri home computers realizzati secondo lo standard MSX. Il linguaggio MSX-BASIC riconosce pressoché tutte le istruzioni della versione MBASIC-80 della Microsoft, integrate da istruzioni per suonare musica, usare il colore, creare effetti di animazione con oggetti in movimento (i cosiddetti "sprites") e usare i comandi manuali (i cosiddetti "joysticks").

In questo manuale si troveranno innanzitutto le istruzioni per predisporre il VG8000.
Il Capitolo 2 spiega il significato e l'uso delle varie istruzioni MSX-BASIC. Di queste, si troverà una descrizione ancora più elaborata nello speciale manuale di riferimento.
Il Capitolo 3 ("Hardware") spiega come estendere le possibilità dell'home computer.
Nel Capitolo 4 ("Appendici") si troveranno ulteriori importanti informazioni.

Leggendo attentamente questo manuale e provando praticamente tutti gli esempi forniti, si sarà presto in grado di scrivere propri programmi col VG8000. A questo punto buon divertimento con l'incredibile e versatile home computer VG8000!

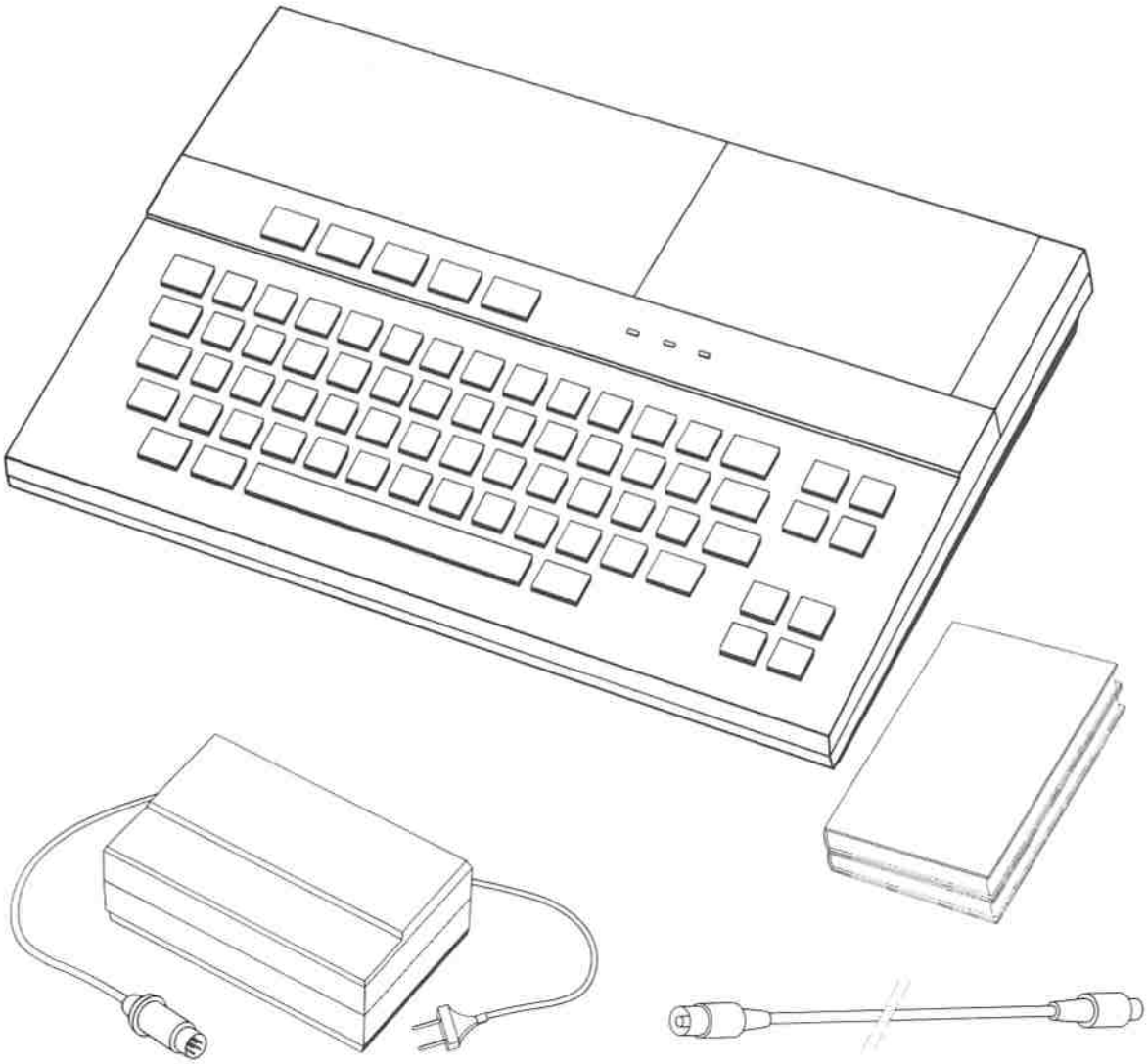
Capitolo 1

Installazione del VG8000

1.1 Apertura dell'imballo

Aprendo la scatola si troverà tutto ciò che occorre per cominciare a lavorare con il nuovo home computer MSX. È in ogni caso opportuno controllarne il contenuto. La scatola dovrebbe comprendere quanto segue:

1. La console VG8000 con tastiera alfanumerica.
2. Un alimentatore per fornire al computer l'appropriata tensione.
3. Un cavo per il collegamento con il televisore.
4. I manuali VG8000.



Per usare l'home computer VG8000 occorre soltanto un televisore in bianco e nero o a colori o un monitor. Ovviamente, per sfruttare adeguatamente le numerose possibilità offerte dal colore, occorre un ricevitore a colori.

Seguire le istruzioni di installazione nel successivo capitolo 1.2 che illustra tutto quanto occorre per cominciare.

Per l'home computer VG8000 MSX è disponibile una vasta gamma di accessori speciali, che ne estendono notevolmente le possibilità.

1. Un registratore di dati, da usare come memoria esterna per la memorizzazione di programmi, di file, ecc.

2. Joysticks per utilizzare i numerosi videogiochi speciali disponibili per l'home computer MSX.

3. Un monitor, che consente di avere immagini più nitide e più brillanti... e di evitare possibili conflitti a proposito dell'uso del televisore di famiglia.

4. Cartucce di espansione per aumentare la memoria del VG8000.

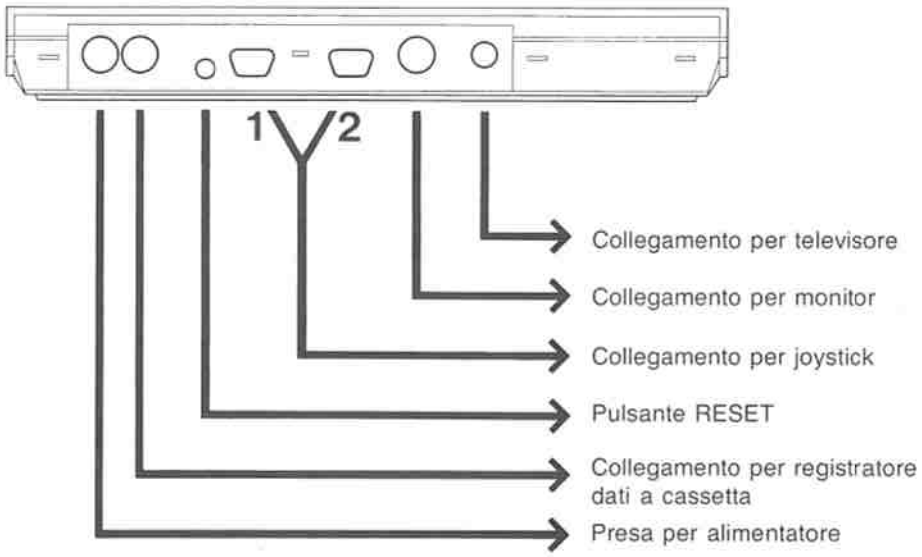
5. La stampante per produrre registrazioni stampate permanenti dei programmi e dei file. La stampante, in combinazione con un programma di word processing permette di usare il VG8000 per scrivere lettere, relazioni, ecc.

6. Varie cartucce per programmi speciali.

7. Unità a dischetti da utilizzare come memoria esterna di grande capacità facilmente accessibile per la memorizzazione di programmi e di file.

1.2 Connessioni

Nella parte posteriore del computer si troveranno le seguenti prese per i vari collegamenti:



Connessioni

1. Per prima cosa collegare il cavo speciale alla presa per televisore che si trova nella parte posteriore del computer. Collegare quindi l'altra estremità di questo cavo alla presa di ingresso d'antenna del televisore.

2. Collegare l'alimentatore alla relativa presa di ingresso del computer, quindi collegare l'altra estremità ad una normale presa di corrente.

3. Accendere il televisore. Scegliere uno dei canali che non è usato per la normale ricezione televisiva e riservarlo per l'uso con il computer. Scegliere il canale appropriato (indicato su un'etichetta autoadesiva applicata alla parte posteriore del computer di fianco alla presa).

4. Accendere ora il computer con l'apposito interruttore posto sull'alimentatore.

5. Si vedrà comparire sullo schermo la scritta

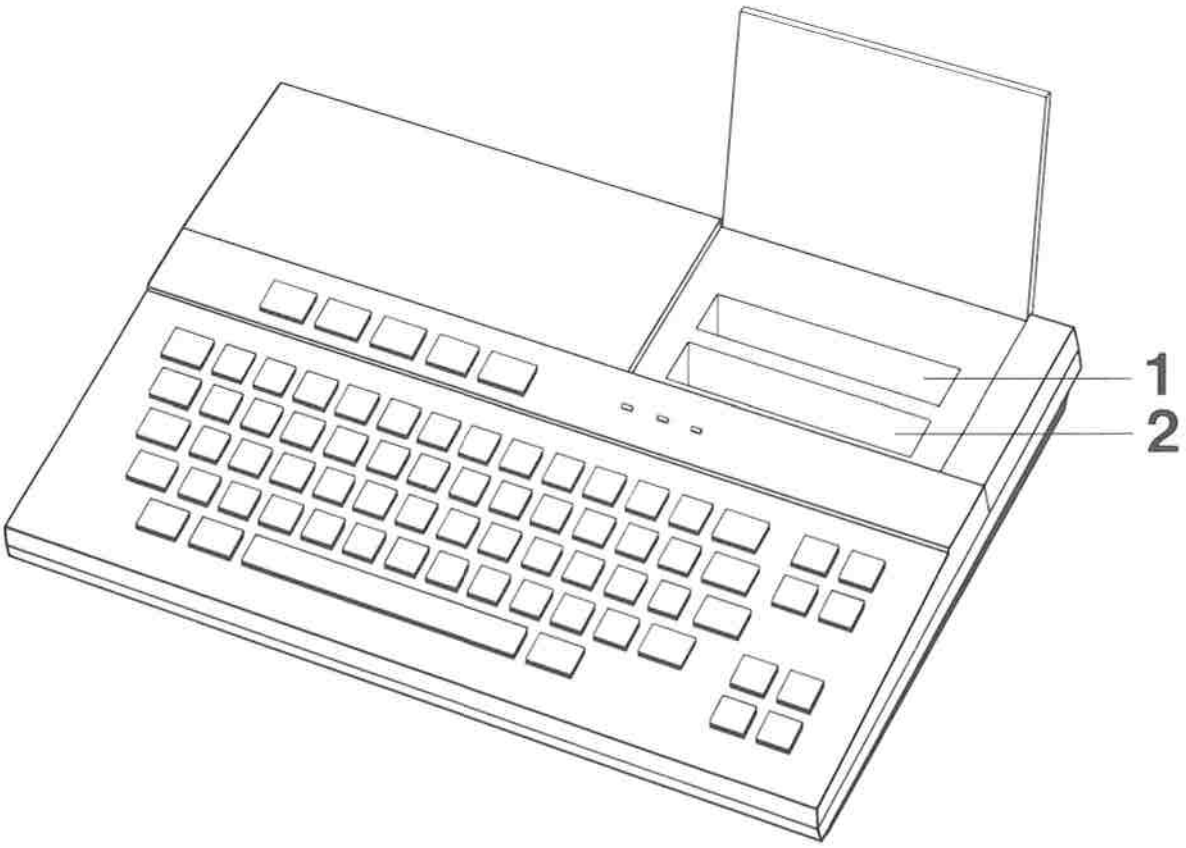
   ```
   MSX system
   ```
   ecc.

   Se si usa un televisore a colori o un monitor, lo schermo assumerà un colore blu e le lettere compariranno in bianco.

6. Quando non sono collegate cartucce, sullo schermo comparirà il testo seguente:

```
MSX BASIC
ecc.
```

Il VG8000 è ora pronto per l'uso.



Inserimento della cartuccia

Spegnere il VG8000 prima di inserire una cartuccia in uno degli appositi connettori. Inserire la cartuccia, con l'etichetta rivolta in avanti in uno degli appositi connettori. Quindi riaccendere il computer.

Collegamento dei joysticks

Spegnere il computer prima di collegare i joysticks.
Il VG8000 prevede due prese speciali per due comandi manuali del tipo joystick.
La presa N. 1 è per il comando di destra.
La presa N. 2 è per il comando di sinistra.

Collegamento di un registratore dati a cassetta

Spegnere il computer prima di collegare il registratore dati.

Ciò significa che occorre collegare il registratore prima di iniziare a scrivere un programma! In caso contrario tutte le informazioni presenti nella memoria del computer vanno perse nel momento in cui viene a mancare la corrente o si spegne l'apparecchio!

Quando si acquista un registratore dati, assicurarsi che il rivenditore fornisca anche un adatto cavo di collegamento, che va collegato all'apposita presa sul computer. L'altra estremità del cavo presenta tre spine, che devono essere collegate al registratore come segue:

1. La spina nera va nella presa contrassegnata REM del registratore.

2. La spina rossa va nella presa contrassegnata MIC del registratore.

3. La spina bianca va nella presa contrassegnata EAR del registratore.

Per il funzionamento del registratore, fare riferimento al capitolo 2.5.

Collegamento di un monitor

È stata prevista una presa speciale per collegare un monitor. Un monitor, come è noto, è un tipo speciale di televisore che non riceve però i normali programmi televisivi, ma che produce immagini da una fonte diretta, ad esempio un videoregistratore o un computer. Acquistando un monitor, assicurarsi che il rivenditore fornisca anche l'adatto cavo di collegamento.

Uso del pulsante RESET

Quando si preme il pulsante RESET (posto nella parte posteriore del computer), il funzionamento del computer viene temporaneamente interrotto. Ciò significa anche che la memoria del computer viene cancellata completamente, pronta per ripartire da capo, esattamente come se si accendesse il computer per la prima volta.

Importante
Per rimuovere il coperchio dai connettori delle cartucce, aprirlo completamente quindi spingerlo leggermente all'indietro fino a che non si libera con un piccolo scatto.

1.3 La tastiera

Si può a questo punto cominciare a provare tutti i tasti senza timore di danneggiare il computer, supponendo naturalmente che non si usi una forza eccessiva. In effetti i tasti del VG8000 richiedono una pressione minima. Abituarsi a lavorare sulla tastiera per rendersi conto di qual è la pressione più adatta per far comparire velocemente e correttamente i caratteri sullo schermo.

Se si tiene un tasto abbassato più a lungo del necessario, si ottiene lo stesso effetto che si avrebbe battendo ripetutamente lo stesso tasto. Lo stesso carattere viene cioè ripetuto fino a che il tasto non viene rilasciato.

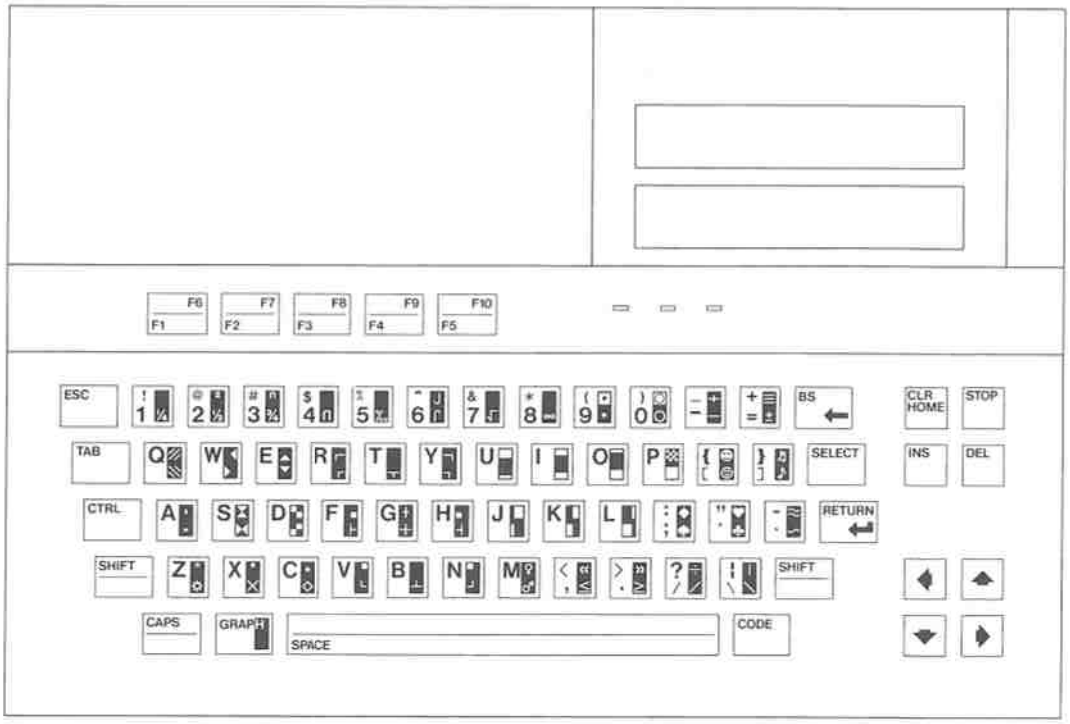
I tasti presenti sulla tastiera sono divisi in quattro gruppi. Iniziando dall'alto:

1. Cinque tasti di funzione nella fila superiore.

2. Quattro tasti di comando "HOME", "STOP", "INS" e "DEL" all'estrema destra.

3. Quattro tasti per il controllo del cursore immediatamente sotto i quattro tasti di comando.

4. La rimanente parte della tastiera comprende tutti i tasti standard per macchina per scrivere, unitamente ad alcuni tasti speciali ad esempio "ESC", "TAB", "CTRL", ecc.



I tasti di funzione

Le funzioni effettive di questi tasti sono programmabili (vedere capitolo 2.1). I tasti sono comunque preprogrammati in maniera standard prima che il computer lasci la fabbrica. Le funzioni corrispondenti ai tasti da F1 a F5 compaiono automaticamente alla base dello schermo. Premendo il tasto SHIFT compaiono invece le funzioni standard da F6 a F10 degli stessi tasti. Per eseguire una funzione, basta premere il tasto corrispondente. Premendo ad esempio simultaneamente il tasto SHIFT ed il tasto F1, viene eseguita la funzione F6.

I tasti di comando

I tasti di comando svolgono le seguenti funzioni:

HOME	Fa spostare il cursore alla prima posizione della prima riga.
HOME + SHIFT	Premuti simultaneamente, questi due tasti ripuliscono lo schermo, ossia cancellano dallo schermo tutto il testo e le eventuali altre immagini presenti.
STOP	Fa interrompere un programma. Il programma continua l'esecuzione quando si preme di nuovo lo stesso tasto. In effetti, esegue una funzione di pausa.
INS	Dà inizio alla funzione di inserimento, che è possibile interrompere premendo di nuovo lo stesso tasto (INS sta per "inserimento").
DEL	Rimuove dallo schermo il carattere sul quale si trova il cursore (DEL sta per "cancellazione").
Tasti di comando del cursore	Ciascun tasto sposta il cursore nella direzione corrispondente alla freccia.

Tasti per la macchina per scrivere

Questi tasti funzionano esattamente come quelli di una macchina per scrivere standard. Se si preme il tasto della lettera "A", sullo schermo compare la lettera "a". Se si preme lo stesso tasto premendo contemporaneamente il tasto SHIFT, sullo schermo compare la lettera "A". Gli altri tasti per le cifre e i simboli di punteggiatura funzionano allo stesso modo.
Intorno ai tasti standard per macchina per scrivere si noteranno anche alcuni tasti speciali:

ESC	Questo tasto funziona soltanto quando al VG8000 è stata collegata una stampante. (ESC sta per "Escape-Uscita").
TAB	Quando si preme questo tasto, il cursore si sposta alla successiva posizione di tabulazione. (TAB sta per "Tabulazione").
CTRL	Questo tasto serve soltanto quando viene usato in combinazione con un altro tasto, come descritto nell'Appendice D (CTRL sta per "controllo").
SHIFT	La funzione di questo tasto è stata già descritta nel precedente paragrafo.

CAPS Se si preme una volta questo tasto, le lettere successivamente battute compariranno in maiuscolo. Se si preme di nuovo lo stesso tasto, tutte le lettere successivamente battute ricompariranno in minuscolo (CAPS sta per "maiuscole").

BS Quando si preme questo tasto, il carattere immediatamente a sinistra del cursore scompare dallo schermo. (BS sta per Backspace — "ritorno unitario").

SELECT Questo tasto è di utilità soltanto quando si usa una cartuccia programma. Premendolo, è possibile effettuare una selezione delle varie opzioni che il programma offre.

RETURN Questo tasto viene usato dopo aver battuto un'istruzione o un comando MSX-BASIC per segnalare la fine dell'istruzione o del comando stessi.

GRAPH Premendo questo tasto, si entra nel modo grafico; dopo di che, se si preme ad esempio il tasto "S", si vedrà comparire sullo schermo un simbolo grafico. Se invece si premono simultaneamente i tasti "S" e SHIFT si vedrà comparire un altro simbolo. Premere di nuovo GRAPH per riportare tutti i tasti alle rispettive funzioni normali. (GRAPH sta per graphic — "grafico").

CODE Quando si preme questo tasto, si illumina la spia corrispondente e, da questo momento, praticamente tutti i tasti produrranno alcune lettere specifiche, tipiche della lingua in uso. Lo stesso succede quando si preme uno dei tasti simultaneamente al tasto SHIFT. Premere di nuovo CODE per riportare tutti i tasti alle rispettive funzioni normali.

Riassumendo

Praticamente ogni tasto della tastiera del VG8000 è in grado di produrre sei caratteri diversi.

1. Lettere minuscole
2. Lettere maiuscole (tasto + SHIFT)
3. Simboli grafici 1, dopo aver premuto il tasto GRAPH
4. Simboli grafici 2 (tasto + SHIFT), dopo aver premuto il tasto GRAPH
5. Caratteri tipici, dopo aver premuto il tasto CODE
6. Caratteri tipici, (tasto + SHIFT) dopo aver premuto il tasto CODE

Si troverà un ripasso completo nell'Appendice G di questo manuale.

Capitolo 2

MSX-BASIC

2.1 Uso del linguaggio MSX-BASIC

Quando si accende l'home computer VG8000, sullo schermo si vede comparire il seguente testo:

```
MSX system
ecc.
```

Successivamente, il computer controlla se negli appositi connettori sono state inserite eventuali cartucce. In caso contrario, sullo schermo compare automaticamente il seguente testo:

```
MSX BASIC
ecc.
```

Il piccolo rettangolo immediatamente al disotto della scritta "Ok" è detto "cursore" ed indica il punto in cui comparirà qualsiasi testo che verrà successivamente battuto.
Il numero indica le posizioni ancora libere nella memoria del computer.
La scritta "Ok" compare sullo schermo ogni volta che MSX-BASIC è a livello comandi, ossia quando è pronto a ricevere le istruzioni.
A questo punto c'è la possibilità di scegliere uno dei due modi d'impiego dell'MSX-BASIC: il "modo diretto" ed il "modo indiretto".

Il modo diretto

Lavorando nel modo diretto, è possibile usare tutti i comandi e praticamente tutte le istruzioni (con o senza funzioni). Si troverà un sommario completo di tutti i comandi, istruzioni e funzioni nel manuale di riferimento.
Nel modo diretto, tutti i comandi e le istruzioni sono eseguiti immediatamente non appena si preme il tasto RETURN ma non sono caricati nella memoria del computer.
Battere il seguente esempio:

```
PRINT 5+2
```

Se si compie un errore di battitura, usare il tasto BS (= Back Space-Ritorno unitario) per rimuovere il carattere immediatamente alla sinistra del cursore. In questo modo è possibile cancellare tutte le lettere necessarie per effettuare la correzione. Quando la riga è corretta, premere il tasto RETURN.

Da ora in poi è importante ricordarsi di premere questo tasto dopo aver completato la battitura di un comando o di un'istruzione.

MSX-BASIC eseguirà immediatamente l'istruzione e sullo schermo comparirà il risultato:

```
7
Ok
```

La comparsa della scritta "Ok" dice che MSX-BASIC è ritornato a livello comandi, in attesa della successiva istruzione.

Il modo indiretto

Lavorando nel modo indiretto, tutte le istruzioni sono precedute da un numero di riga. Queste istruzioni non vengono eseguite immediatamente quando si preme il tasto RETURN ma vengono inserite nella memoria del computer in sequenza di numero di riga. Battere ora questo esempio:

```
10 PRINT 5+2
```

Premere il tasto RETURN. Sembra che l'operazione non abbia prodotto apparentemente alcun risultato, a parte il fatto che il cursore si è spostato all'inizio della riga successiva, in attesa di successive istruzioni. In effetti, la riga 10 è stata caricata nella memoria del computer. Una combinazione di tali righe contenenti istruzioni, ecc. è detta "programma". Tutte le istruzioni in tale programma verranno eventualmente eseguite in sequenza per numero di riga quando si impartisce il comando "RUN" e si preme il tasto RETURN.

Pertanto, lavorando nel modo indiretto, ogni nuova riga deve iniziare con un proprio numero. Sotto questo aspetto non ci sono problemi in quanto è possibile usare qualsiasi numero compreso tra 0 e 65529. È consigliabile non usare numeri successivi di riga nello scrivere un programma ma saltarne qualcuno. Così facendo sarà sempre possibile in una fase successiva, se necessario, aggiungere righe ulteriori tra quelle esistenti.

È inoltre possibile chiedere a MSX-BASIC di eseguire la numerazione automaticamente. Basta battere il comando "AUTO" prima di iniziare la battitura delle righe di programma. Se a questo punto si preme RETURN si vedrà che MSX-BASIC indica automaticamente 10 come numero della prima riga. L'asterisco dopo il 10 dice che nella memoria del computer c'è già una riga numero 10. Premendo un'altra volta il tasto RETURN, la riga successiva porterà il numero 20, ecc. Per interrompere il processo di numerazione automatica, occorre premere simultaneamente i tasti CTRL e C.

MSX-BASIC permette di inserire un massimo di 255 caratteri (compresi gli spazi ed i numeri di riga) per ogni riga di programma.
Una riga di programma può contenere più di un'istruzione posto che queste vengano separate da due punti (:). Provare questo esempio:

```
10 PRINT 8+2:PRINT 6*4
RUN
 10
 24
Ok
```

MSX-BASIC ha eseguito entrambi i calcoli separatamente, dato che le due istruzioni erano separate da due punti.

Per rimuovere tutte le righe di programma precedentemente caricate nella memoria del computer, si usa il comando "NEW" prima di premere di nuovo il tasto RETURN.

Le differenze tra comandi e istruzioni

I comandi e le istruzioni possono essere usati sia nel modo diretto che nel modo indiretto. Dopo aver eseguito un comando, MSX-BASIC ritorna sempre a livello comandi. Provare l'esempio che segue battendo:

```
10 PRINT 3+1
20 LIST
30 PRINT 6-3
RUN
```

"LIST" è un comando MSX-BASIC che viene usato per far comparire ossia "listare" sullo schermo tutte le righe del programma. Si tratta di un comando che viene usato per riesaminare e controllare un programma o per apportarvi modifiche o aggiunte.

Poiché "LIST" è un comando, MSX-BASIC ritornerà a livello comandi dopo aver eseguito la riga 20 di questo esempio. Significa che la riga 30 non sarà eseguita.

Nel manuale di riferimento MSX-BASIC si può vedere di volta in volta se si ha a che fare con un comando o con un'istruzione.

La differenza tra istruzioni e funzioni

Una funzione deve sempre essere preceduta da un'istruzione. "FRE(0)" è un esempio di funzione, che viene usata per chiedere a MSX-BASIC quanti bytes ci sono ancora a disposizione nella memoria del computer. Provare il seguente esempio. Il comando "NEW" è usato per rimuovere dalla memoria del computer tutte le righe precedentemente immesse:

```
NEW
10 FRE(0)
```

Impartire ora il comando "RUN". Si vedrà che MSX-BASIC fa comparire sullo schermo un messaggio di errore: "Syntax error in 10" (Errore di sintassi alla riga 10). Ciò in quanto si è usata una funzione senza aver aggiunto l'istruzione richiesta. Provare l'esempio che segue:

```
10 PRINT FRE(0)
RUN
```

Ora MSX-BASIC dice quanti bytes sono ancora disponibili in memoria per continuare il programma.

Battitura delle istruzioni

"PRINT" è una delle istruzioni che verranno eseguite abbastanza spesso.
Per risparmiare tempo e fatica, è possibile usare in sua vece un punto interrogativo (?). MSX-BASIC trasformerà il punto interrogativo nell'istruzione "PRINT". Basta provare con un esempio. Battere:

```
10 ? 2+9
```

Premere il tasto RETURN e battere il comando "LIST". Premendo di nuovo il tasto RETURN si vedrà comparire:

```
10 ? 2+9
LIST
10 PRINT 2+9
```

MSX-BASIC fornisce la possibilità di battere comandi ed istruzioni mediante un solo tasto. I tasti di funzione da F1 a F10 sono stati previsti proprio a questo scopo. Alla base dello schermo si vedrà quali sono le funzioni che corrispondono ai tasti F1 e F5. Quando

si preme il tasto SHIFT, è possibile leggere sullo schermo quali comandi ed istruzioni vengono attivati con i tasti F6 e F10.

È inoltre possibile richiedere a MSX-BASIC di produrre un listato o elenco di comandi e di istruzioni che sono stati programmati per i tasti di funzione.
Battere l'istruzione "KEY LIST" e premere il tasto RETURN.

È inoltre possibile modificare i comandi e le istruzioni assegnati ai tasti di funzione usando l'istruzione "KEY". Provare l'esempio che segue e battere:

```
KEY 1,"GOSUB"
```

Premere quindi il tasto RETURN. D'ora in poi il tasto F1 serve a produrre le istruzioni "GOSUB".
È possibile rimuovere dallo schermo il riferimento programmato per i tasti di funzione con l'istruzione "KEY OFF". I riferimenti ricompariranno con l'istruzione "KEY ON".

Quando e come usare "" : ; . ,

MSX-BASIC eseguirà alla perfezione tutti gli ordini che gli verranno impartiti, posto che siano battuti correttamente. Ciò vale anche per i vari segni di punteggiatura che svolgono delle funzioni specifiche nell'ambito di un programma.

Tutti i caratteri disposti tra le virgolette, ad esempio, saranno scritti sullo schermo man mano che vengono battuti. È possibile rendersene conto facilmente provando con i seguenti due esempi:

```
10 PRINT 5+2
20 PRINT "5+2"
RUN
```

La prima volta MSX-BASIC eseguirà il calcolo 5 + 2 = 7 ma nel secondo esempio esso si limiterà a riprodurre il testo che è stato battuto e cioè 5 + 2.

Il due punti (:) serve, come già detto, a dividere due istruzioni scritte nelle stessa riga di programma.

Il punto e virgola (;) può essere usato in un'istruzione "PRINT". Dopo aver eseguito le istruzioni ed aver raggiunto il punto e virgola, il cursore non procede immediatamente alla riga successiva.
Il punto e virgola viene anche usato in combinazione con l'istruzione "INPUT" quando la situazione richiede una frase interrogativa.

Il punto (.) può essere usato unitamente ai comandi "LIST", "AUTO" e "DELETE" se si desidera fare riferimento all'ultima riga di programma.

La virgola (,) può essere usata soltanto in combinazione con i comandi e le istruzioni descritti nel manuale di riferimento MSX-BASIC.

Messaggi di errore

Quando MSX-BASIC incontra un errore in una delle righe di programma, fa comparire sullo schermo un messaggio di errore. Un sommario completo dei messaggi di errore è nell'Appendice A di questo manuale.

2.2 Modifica di un programma

Una riga di programma può essere facilmente modificata in qualsiasi momento battendo una nuova riga con lo stesso numero.
L'aggiunta di nuove righe al programma può essere effettuata battendo le nuove righe ed attribuendo loro numeri non ancora usati, tenendo presente naturalmente che occorre scegliere un numero che inserisca la nuova riga nella sequenza appropriata.
Le righe di programma possono essere cancellate con il comando "DELETE". Se si tratta di rimuovere una singola riga, è possibile farlo impostando il numero di quella riga seguito dal tasto RETURN.
È inoltre possibile modificare le righe di programma sullo schermo. Questa operazione è detta "editing" o "correzione sullo schermo".

Correzione sullo schermo

La correzione sullo schermo dà la possibilità di modificare le righe di programma che compaiono sullo schermo attraverso i tasti INS, BS e DEL.
Provare l'esempio che segue battendo:

```
NEW
10 pontt
```

e premere il tasto RETURN. In realtà si voleva scrivere "PRINT" e si è ovviamente compiuto un errore di battitura. Non occorre però preoccuparsi se sono state usate le lettere minuscole. MSX-BASIC converte automaticamente in maiuscole tutti i comandi, le istruzioni e le funzioni.
Battere ad esempio:

```
LIST
```

e premere il tasto RETURN. Sullo schermo si vedrà comparire:

```
10 PONTT
```

Dato che si desidera che "PONTT" venga modificato in "PRINT", occorre correggere questa riga di programma. Per prima cosa disporre il cursore sulla lettera "O" di "PONTT" usando i tasti del cursore. Premere quindi il tasto INS (inserimento). Si noterà che il cursore si è trasformato in una strisciolina molto sottile. Ciò significa che ogni lettera o cifra che si batte da questo momento in poi sarà inserita tra le lettere presenti in quel punto sullo schermo. Premendo il tasto "R" si vedrà che la lettera "R" è stata disposta tra la "P" e la "O". Premere di nuovo il tasto INS. Il cursore ritornerà alla sua dimensione normale. Ciò significa che ogni lettera o cifra battuta d'ora in poi sostituirà qualsivoglia simbolo presente sullo schermo. Premere il tasto "I" in modo che la riga si presenti nel modo seguente:

```
10 PRINTT
```

Il cursore copre ora la "N". A questo punto non resta che rimuovere una "T". Per far

ciò, si sposta il cursore alla prima "T" usando i tasti di comando del cursore, quindi si preme il tasto DEL (delete-cancellazione).

La prima "T" è stata rimossa e la seconda "T" è stata spostata di una posizione verso sinistra, per riempire il vuoto così creato. Premere ora il tasto RETURN.

> In MSX-BASIC le modifiche vengono immesse, ossia convalidate, soltanto quando si preme il tasto RETURN.

Quando si batte il comando "LIST" seguito dal tasto RETURN, MSX-BASIC mostrerà che la riga è stata modificata in:

```
10 PRINT
```

il che è esattamente quello che si desiderava.

Assicurarsi sempre che i nuovi comandi ed istruzioni dopo la correzione sullo schermo siano sempre immessi in una riga vuota. Si supponga per esempio di avere il seguente testo sullo schermo:

```
10 PRINT 9+2
20 PRINT 3-6
```

Si supponga ora che il cursore copra il "2" della riga di programma 20 dato che è stata appena modificata la riga 10. Se si batte ora il comando "LIST", si ottiene questo risultato:

```
10 PRINT 9+2
LISTRINT 3-6
```

Quando si preme il tasto RETURN, MSX-BASIC emette un messaggio di errore dato che il comando "LISTRINT 3-6" non esiste in MSX-BASIC. È naturalmente possibile evitare ciò spostando il cursore su una riga vuota dello schermo oppure premere simultaneamente i tasti SHIFT e HOME prima di battere il comando "LIST".

Se il contenuto di una particolare riga non è di ulteriore importanza è inoltre possibile premere il tasto CTRL ed il tasto E simultaneamente. Ciò non fa che rimuovere tutti i simboli su quella riga dopo il comando "LIST".

Il tasto STOP

Il tasto STOP non ha alcuna funzione quando MSX-BASIC è a livello comandi.

Quando per contro MSX-BASIC sta eseguendo un programma, è possibile usare il tasto STOP per interromperlo. Dopo aver premuto il tasto STOP, sullo schermo ricompare il cursore. Il programma può riprendere quando si preme di nuovo il tasto STOP.
Se si desidera interrompere un programma e ritornare a livello comandi MSX-BASIC, premere simultaneamente i tasti CTRL e STOP. Si vedrà comparire sullo schermo il seguente messaggio:

```
Break in nnnnn
(Interruzione in nnnnn)
```

Qui "nnnnn" indica il numero di riga alla quale il programma è stato interrotto.

2.3 Costanti e variabili

La memoria del computer MSX è in grado di accogliere non soltanto righe con comandi, istruzioni e funzioni ma anche altri elementi che svolgono un ruolo importante durante l'esecuzione di un programma. Questi elementi possono essere divisi in due categorie: "costanti" e "variabili".

Costanti

Le costanti contengono informazioni che non cambiano nel corso del programma. A loro volta le costanti possono essere di due tipi diversi: costanti alfanumeriche e costanti numeriche.
Una costante alfanumerica può contenere un massimo di 255 caratteri e deve iniziare e terminare con un segno di virgolette (" . . . ").
Esempio: "HALLO" o "12345".
Quantunque la costante "12345" contenga cifre, non può essere usata per eseguire calcoli.
Una costante numerica è invece rappresentata da una cifra positiva o negativa. In MSX-BASIC, la virgola dei decimali deve essere sostituita da un punto (.).
Esempio: in MSX-BASIC, 123,45 viene scritto nella forma 123.45.

MSX-BASIC riconosce sei diversi tipi di costanti numeriche:

1. **Costante intera**
   Può trattarsi di un qualsiasi numero intero compreso tra -32768 e 32767
   Esempio: 13 oppure -25

2. **Costante in virgola fissa**
   È quella costituita da un numero che prevede cifre dopo il punto decimale.
   Esempio: 15.73 oppure -178.3

3. **Costante in virgola mobile**
   È quella costituita da un numero scritto in notazione scientifica.
   Esempio: 2.34E8 (=2.34*10^8=234,000,000)

4. **Costante esadecimale**
   È quella costituita da un numero esadecimale, che per convenzione viene fatto precedere dai caratteri "&H"
   Esempio: &H3F

5. **Costante ottale**
   È costituita da un numero espresso nel sistema di numerazione con base 8, che per convenzione viene fatto precedere dai caratteri "&O" oppure "&".
   Esempio: &037 oppure &37

6. **Costante binaria**
   È costituita da un numero binario, che per convenzione viene fatto precedere dai caratteri "&B"
   Esempio: &B01010111

Variabili

Una variabile è una parte della memoria utilizzata per la memorizzazione di un elemento di informazioni ad uso del computer.

Il programmatore attribuisce un nome a ciascuna variabile, consentendo così al computer di localizzarla quando necessario. Ci sono due tipi di variabili: variabili alfanumeriche e variabili numeriche.

Il nome della variabile alfanumerica termina sempre con il segno del dollaro ($).
Esempio: `A$="HALLO"` oppure `AB$="12345"`

Una variabile numerica può contenere un numero positivo o negativo.
Una variabile numerica intera può contenere soltanto un numero intero compreso tra —32768 e 32767. Il nome di tale variabile termina sempre con il segno di percento (%).
Esempio: `A%=10`

Precisione singola e doppia

Le costanti e le variabili numeriche possono essere espresse con precisione singola o doppia.

Nella **forma a precisione singola**, i numeri occupano 4 byte della memoria. I numeri possono essere composti da un massimo di 6 cifre.

Una costante in virgola fissa a precisione singola è contraddistinta da un punto esclamativo (!).
Esempio: `12.5!`

Una costante in virgola mobile a precisione singola è contraddistinta dalla "E" come simbolo di esponente.
Esempio: `2.34E8`

Una variabile a precisione singola è contraddistinta da un punto esclamativo (!) posto dopo il nome della variabile stessa.
Esempio: `AB!`

Nella **forma a doppia precisione** i numeri occupano 8 byte della memoria. Un numero in questa forma può comprendere un massimo di 14 cifre.

Una costante in virgola fissa a doppia precisione è contraddistinta dal simbolo "#".
Esempio: `23.33#`

Una costante in virgola mobile a doppia precisione è contraddistinta da una "D" come simbolo di esponente.
Esempio: `3.768D-7`

Una variabile a doppia precisione è contraddistinta dal simbolo "#" come ultimo carattere del nome della variabile stessa.
Esempio: `DA#`

Se il nome di una variabile non contiene "$", "%", "!" oppure "#" come ultimo carattere, MSX-BASIC la considera una variabile a doppia precisione.

Esempio: `ME#` è una variabile a doppia precisione, che occupa 8 byte della memoria.

PT% è una variabile intera, che occupa 2 byte di memoria

ZS! è una variabile a precisione singola, che occupa 4 byte di memoria

ME è una variabile a doppia precisione, che occupa 8 byte di memoria

AB$ è una variabile alfanumerica che occupa 3 byte di memoria più lo spazio dei byte richiesti per il contenuto effettivo della variabile.

In totale MSX-BASIC consente di caricare nella memoria del computer 200 caratteri sotto forma di variabili alfanumeriche. L'istruzione "CLEAR" rende possibile cambiare questo massimo di 200.
Con MSX-BASIC il tipo di variabile può anche essere determinato con le istruzioni: "DEFINT", "DEFSTR", "DEFSNG" e "DEFDBL". Si troveranno dettagli completi che riguardano queste istruzioni nel manuale di riferimento.

Conversione di tipo

MSX-BASIC convertirà, se necessario, le variabili numeriche in un diverso tipo di costante.
Esempio 1:

```
NEW
10 A%=13.03
20 PRINT A%
RUN
 13
```

In questo esempio una costante in virgola fissa viene inserita in una variabile intera. Le cifre che stanno dietro il punto decimale vanno perse.
Esempio 2:

```
NEW
10 E#=6!/7!
20 PRINT E#
RUN
 .85714285714286
```

Quantunque il calcolo "6!/7!" sia indicato a precisione singola, il risultato è rappresentato in doppia precisione, dato che la variabile ricevente "E#" è indicata a sua volta a doppia precisione.
Esempio 3:

```
NEW
10 E!=6/7
20 PRINT E!
RUN
 .857143
```

Quantunque il calcolo "6/7" sia indicato a doppia precisione, il risultato è espresso in precisione singola dato che la variabile ricevente "E!" è indicata con precisione singola.
Esempio 4:

```
NEW
10 B!=SQR(2)
20 A=B!
30 PRINT B!;A
```

```
RUN
 1.41421   1.41421
```

Nonostante il fatto che la variabile "A" sia a doppia precisione, nell'istruzione "PRINT" vengono indicate soltanto sei cifre dato che il contenuto della variabile a doppia precisione (A) è stato reso identico a quello della variabile a precisione singola (B!).

La conversione dei tipi di variabili può anche essere ottenuta con le funzioni "CBDL", "CINT", "CSNG" e "STR$". Si troveranno ulteriori dettagli riguardanti queste funzioni nel manuale di riferimento.

Nomi delle variabili

Il programmatore può scegliere da sé i nomi delle variabili posto che rispetti le seguenti condizioni:

1. Il primo carattere del nome deve sempre essere alfanumerico (da A a Z) mentre i successivi possono essere alfabetici o numerici.

2. Quantunque un nome possa comprendere più di due caratteri, MSX-BASIC riconosce soltanto i primi due. Per MSX-BASIC, ad esempio, i nomi "PO" e "POST" per esempio si riferiscono alla stessa variabile.

3. Il nome della variabile non può corrispondere ad una delle parole riservate indicate nell'Appendice F di questo manuale.

Contenuto delle variabili

Ad una variabile può essere attribuito un certo valore usando l'istruzione "LET". Esempio:

```
10 LET A=10
20 LET B$="HALLO"
```

L'uso della parola "LET" non è strettamente necessario: anche se la si tralascia si ottiene lo stesso effetto.
Ad una variabile può anche essere attribuito un contenuto usando l'istruzione "INPUT". Quando un computer esegue un'istruzione "INPUT", attende fino a che sulla tastiera sono state battute tutte le informazioni. Solo dopo aver premuto il tasto RETURN, le informazioni vengono inserite nella variabile indicata unitamente all'istruzione "INPUT". Può essere inoltre aggiunta una frase di interrogazione usando l'istruzione "INPUT". Esempio:

```
NEW
10 INPUT "DAI UN NUMERO";A
20 INPUT "AL TUO NOME";B$
RUN
DAI UN NUMERO?
```

Il punto interrogativo viene aggiunto automaticamente. Nella riga di programma numero 10, MSX-BASIC accetta soltanto informazioni numeriche. Nella riga di programma 20 possono essere usati simboli alfanumerici.

Variabili speciali

Scrivendo un programma con MSX-BASIC è possibile usare numerose variabili speciali. Queste variabili sono: "TIME", "VDP", "BASE" e "SPRITE$".
Esempio:

```
NEW
10 PRINT "INIZIO":TIME=0
20 FOR I%= 1 TO 1000:NEXT I%
30 PRINT "ORA SONO TRASCORSI";TIME/50;"SECONDI"
RUN
INIZIO
ORA SONO TRASCORSI .8 SECONDI
```

Il suddetto esempio mostra che al computer occorrono 0,8 secondi per eseguire l'istruzione "FOR...NEXT" nella riga di programma 20.

La variabile speciale "SPRITE$" è spiegata in dettaglio nel capitolo 2.12. Le variabili "VDP" e "BASE" dovrebbero essere usate soltanto da coloro che hanno sufficiente familiarità con il funzionamento del chip video.

2.4 Calcoli

MSX-BASIC riconosce cinque operazioni aritmetiche, vale a dire:

1. Addizione (+)
2. Sottrazione (—)
3. Moltiplicazione (*)
4. Divisione (/)
5. Elevamento a potenza (^)

Queste operazioni possono essere combinate in una sola istruzione come nel caso seguente:

```
NEW
10 PRINT 40/5+3
RUN
 11
```

Tenere presente che le operazioni aritmetiche vengono eseguite nell'ordine seguente:

1. Elevamento a potenza
2. Moltiplicazione e divisione
3. Addizione e/o sottrazione

L'ordine di esecuzione può naturalmente essere modificato usando le parentesi come nell'esempio che segue:

```
NEW
10 PRINT 40/(5+3)
RUN
 5
```

Divisione con numeri interi

MSX-BASIC è in grado di dividere nel modo interi, il che significa che la divisione verrà eseguita con numeri senza cifre decimali e che il risultato sarà a sua volta indicato in numeri senza cifre decimali. Una divisione di questo tipo è indicata da una "barretta invertita". Provare innanzitutto questa normale divisione:

```
NEW
10 PRINT 10/4
RUN
 2.5
```

Notare la differenza quando si prova l'esempio che segue:

```
NEW
10 PRINT 10\4
RUN
 2
```

Nella divisione di interi il dividendo ed il divisore sono sempre convertiti in interi.

Calcolo del modulo

Il calcolo del modulo dà il valore residuo o resto sotto forma di numero intero dopo una divisione di interi. Il calcolo del modulo è eseguito facendolo precedere dalla parola "MOD". Per esempio:

```
NEW
10 PRINT 25.68 MOD 6.99
RUN
 1
```

In questo esempio i numeri 25.68 e 6.99 sono convertiti in interi (rispettivamente 25 e 6), dopo di che viene eseguita la divisione. Il quoziente è 4 con un resto (modulo) di 1.

Divisione per zero e supero di capacità

Se durante l'esecuzione di un calcolo viene eseguita una divisione per zero, MSX-BASIC emette un messaggio di errore ed interrompe il programma.
Analogamente, se il campo ricevente è troppo piccolo per contenere il risultato di un calcolo, MSX-BASIC emette un messaggio di errore ed interrompe l'ulteriore esecuzione del programma.

Variabili

Se viene usata una variabile in un'operazione matematica, l'operazione viene eseguita col contenuto di quella variabile come nel caso seguente:

```
NEW
10 A=10:B=5
20 C=A*B
30 PRINT C
RUN
 50
```

Nella riga di programma 20 viene eseguito il calcolo di dieci (il contenuto di "A") moltiplicato per cinque (il contenuto di "B").
Non è possibile eseguire operazioni matematiche con dati alfanumerici. È per contro possibile abbinare variabili alfanumeriche.
Esempio:

```
NEW
10 B$="MSX":C$="- COMPUTER"
20 A$=B$+C$
30 PRINT A$
40 PRINT B$+"- BASIC"
RUN
MSX - COMPUTER
MSX - BASIC
```

In questo caso il contenuto di "A$" è "MSX - COMPUTER".

Funzioni matematiche

Numerosi calcoli matematici di uso frequente sono rappresentati in MSX-BASIC sotto forma di funzioni.

La funzione "ABS" si riferisce al valore assoluto di una cifra.
La funzione "COS" dà il coseno di un angolo espresso in radianti.
La funzione "EXP" dà il quadrato di un numero.
La funzione "LOG" dà il logaritmo naturale di un numero.
La funzione "SIN" dà il seno di un angolo espresso in radianti.
La funzione "SQR" dà la radice quadrata di un numero.
La funzione "TAN" dà la tangente di un angolo espresso in radianti.

Provare l'esempio che segue:

```
NEW
10 X=25
20 PRINT X;SQR(X)
RUN
  25  5
```

È possibile comporre altre funzioni matematiche come indicato nell'Appendice B. È possibile inoltre definire altre funzioni matematiche con l'istruzione "DEF FN". Battere il seguente esempio:

```
NEW
10 DEF FNAB(X,Y)=X^3/Y^2
20 X=6:Y=2
30 T=FNAB(X,Y)
40 PRINT T
RUN
 54
```

I valori di X e Y, come indicato nella riga di programma 20, sono sostituiti dalla definizione matematica della riga 10 moltiplicata per la riga 30.
La variabile "T" contiene il risultato di 6^3/2^2.

2.5 MSX-BASIC e il registratore dati a cassetta

MSX-BASIC consente l'uso di un registratore dati per la conservazione di programmi e di file su una cassetta di nastro. Questi dati possono essere successivamente letti nella memoria del computer.

Registrazione di programmi su una cassetta

Un programma può essere trasferito ad una cassetta attraverso il comando "CSAVE". La procedura è la seguente:

1. Inserire una cassetta vuota nel registratore e riavvolgerla completamente.
2. Far avanzare il nastro contenuto nella cassetta fino a che non si è superata la striscia di guida.
3. Premere simultaneamente i tasti PLAY e REC del registratore.
4. Battere il comando "CSAVE" seguito immediatamente dal nome del programma che si sta per immettere nella cassetta. Inserire il nome tra virgolette (CSAVE "nome del programma").
5. Premere il tasto RETURN.

Il programma sarà trasferito alla cassetta.

Lettura di un programma dalla cassetta

Un programma può essere letto nella memoria del computer dalla cassetta attraverso il comando "CLOAD".

1. Inserire la cassetta contenente il programma nel registratore e riavvolgerla completamente.
2. Premere il tasto PLAY del registratore.
3. Battere il comando: CLOAD "nome del programma".
4. Premere il tasto RETURN.

Se non è stato attribuito alcun nome al programma, unitamente all'istruzione "CLOAD", nella memoria del computer verrà letto il primo programma registrato sul nastro. Quando MSX-BASIC trova un nome di programma diverso da quello chiesto, sullo schermo compare il testo che segue:

```
SKIP (PROGRAM NAME)
```
(Salta (nome programma))

Quindi MSX-BASIC continua la ricerca del programma giusto. Se il programma viene trovato, sullo schermo compare il testo che segue ed il programma viene letto nella memoria del computer.

```
FOUND (PROGRAM NAME)
```
(Trovato (nome programma))

Al termine del lavoro, MSX-BASIC ritorna a livello comandi.

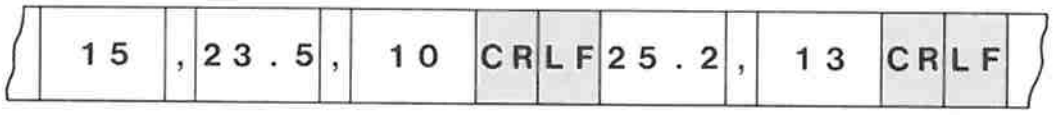
Registrazione di dati su una cassetta

Le informazioni raccolte usando un programma possono a loro volta essere conservate su una cassetta.

Le informazioni sono registrate in sequenza. Questo insieme di informazioni è detto "file". Ad ogni file viene attribuito un nome ed il file deve essere "aperto" per potervici lavorare. Il numero dei file all'interno di un programma deve essere specificato con l'istruzione "MAXFILES". Ecco un esempio:

```
5 MAXFILES=1
10 OPEN "CAS:ADDR" FOR OUTPUT AS#1
20 A=15:B=23.5:C=10:D=25.2:E=13
30 PRINT #1,A;B;C
40 PRINT #1,D;E
50 CLOSE #1
```

In questo esempio, alla riga 10, viene aperto il file denominato "ADDR" (si sta qui facendo riferimento ad un file di indirizzi, ma può funzionare con qualsiasi nome, naturalmente in relazione al tipo di file).
La parola "CAS" indica che il file è registrato su una cassetta.
Le informazioni sono effettivamente registrate sulla cassetta con l'istruzione "PRINT #".
Dopo ogni istruzione "PRINT#" sul nastro viene inserito un codice "CR" e "LF". Tra le variabili numeriche viene automaticamente inserita una virgola. L'esempio suddetto avrebbe la seguente "immagine" sul nastro:



Battendo una variabile alfanumerica, la virgola (,) deve essere aggiunta manualmente con l'istruzione "PRINT#" come nell'esempio che segue:

```
5 MAXFILES =1
10 OPEN "CAS:NAME" FOR OUTPUT AS#1
20 A$="JOHN":B$="PETE"
30 PRINT #1,A$","B$
40 CLOSE #1
```

Questa produrrà sul nastro il seguente risultato:

Notare che le fasi procedurali da 1 a 3, come spiegato a proposito della registrazione di un programma, devono essere eseguite prima di trasferire i dati sulla cassetta!

Lettura di informazioni da una cassetta

I dati che sono stati registrati su una cassetta possono essere riletti nella memoria del computer attraverso l'istruzione "INPUT #".
Per leggere le variabili numeriche dell'esempio precedente occorre usare la seguente istruzione:

```
5 MAXFILES=1
10 OPEN "CAS:ADDR" FOR INPUT AS#1
20 INPUT #1;V,W,X
30 INPUT #1;Y,Z
40 CLOSE #1
```

Per le variabili alfanumeriche occorrono ad esempio le seguenti istruzioni:

```
5 MAXFILES=1
10 OPEN "CAS:NAME" FOR INPUT AS#1
20 INPUT #1,Y$,Z$
30 CLOSE #1
```

Un'istruzione "INPUT #" riempie le variabili con le informazioni fino alla prima virgola o fino al codice "CR" e "LF".
Un'istruzione "LINE INPUT #" riempie le variabili indicate con le informazioni fino al successivo codice "CR" e "LF", come in questo esempio:

```
NEW
5 MAXFILES=1
10 OPEN "CAS:NAME" FOR INPUT AS#1
20 LINE INPUT #1,X$
30 CLOSE #1
```

In questo esempio "X$" contiene "JOHN,PETE".
Con la funzione "EOF" si può verificare se si è raggiunta la fine di un file.
È necessario attenersi rigorosamente alle fasi procedurali da 1 a 2, indicate nelle istruzioni per la lettura di una cassetta prima di leggere le informazioni nella memoria del computer.

Un file di indirizzi

Segue ora un programma che è possibile usare per inserire nomi, indirizzi, ecc. Le informazioni vengono trasferite sulla cassetta e possono essere lette in qualsiasi momento.

```
NEW
10 SCREEN 0:KEY OFF:WIDTH 39:MAXFILES=1
20 CLS:LOCATE 10,2:PRINT "MENU"
30 ON KEY GOSUB 100,400,700
40 LOCATE 5,5:PRINT "F1=IMMETTI INDIRIZZO"
50 LOCATE 5,7:PRINT "F2=SCRIVI INDIRIZZO"
60 LOCATE 5,9:PRINT "F3=FINE PROGRAMMA"
70 KEY (1) ON:KEY (2) ON:KEY (3) ON
80 GOTO 80
100 CLS
110 PRINT:PRINT "PREDISPONI REGISTRATORE
E PREMI SIMULTANEAMENTE PLAY E RECORD"
120 PRINT:PRINT:INPUT "PRONTO (S/N)";F$
130 IF F$="N" GOTO 120
140 IF F$="n" GOTO 120
150 OPEN "CAS:ADDRESS" FOR OUTPUT AS#1
160 CLS
170 LOCATE 1,5:INPUT "NOME";A$
180 LOCATE 1,7:INPUT "INDIRIZZO";B$
190 LOCATE 1,9:INPUT "CITTA'";C$
200 LOCATE 1,11:INPUT "TELEFONO";D$
```

```
210 LOCATE 1,20:INPUT "ESATTO (S/N)";F$
220 IF F$="N" GOTO 160
230 IF F$="n" GOTO 160
240 PRINT #1,A$;",";B$;",";C$;",";D$
250 LOCATE 1,22:INPUT "ALTRO INDIRIZZO (S/N)";F$
260 IF F$="N" GOTO 290
270 IF F$="n" GOTO 290
280 GOTO 160
290 CLOSE #1:RETURN 20
400 CLS
410 PRINT:PRINT "PREDISPONI REGISTRATORE E PREMI PLAY
KEY"
420 PRINT:PRINT:INPUT "PRONTO (S/N)";F$
430 IF F$="N" GOTO 420
440 IF F$="n" GOTO 420
450 OPEN "CAS:ADDRESS" FOR INPUT AS#1
460 CLS:PRINT
470 IF EOF(1) THEN 570
480 INPUT# 1,A$,B$,C$,D$
490 PRINT "NOME      :";A$:PRINT
500 PRINT "INDIRIZZO:";B$:PRINT
510 PRINT "CITTA'    :";C$:PRINT
520 PRINT "TELEFONO  :";D$
530 LOCATE 1,22:INPUT "ALTRO INDIRIZZO (S/N)";F$
540 IF F$="N" GOTO 570
550 IF F$="n" GOTO 570
560 GOTO 460
570 CLOSE #1:RETURN 20
700 CLS:PRINT "FINE PROGRAMMA"
710 END
```

Eseguire il programma e provare ad immettere alcuni indirizzi. Questi verranno registrati sul nastro quando si preme il tasto F1 e saranno riletti nella memoria del computer quando si preme il tasto F2. Non dimenticarsi di riavvolgere la cassetta prima di accingersi a leggere gli indirizzi.
Se il programma non funziona correttamente, controllare che non ci siano errori di battitura.

2.6 Tabelle

Quando in un programma vengono usati molti dati costanti, è consigliabile registrarli in una tabella servendosi dell'istruzione "DATA". I dati sono richiamati dalla tabella attraverso l'istruzione "READ" come nell'esempio che segue:

```
NEW
10 READ X$:PRINT X$
20 GOTO 10
30 DATA GEN,FEB,MAR
RUN
GEN
FEB
MAR
Out of DATA in 10
```

Notare che gli elementi di informazione che compaiono insieme all'istruzione "DATA" sono separati da una virgola (,). L'istruzione "READ" estrae dalla tabella (creata con l'istruzione DATA) l'elemento di informazioni immediatamente successivo. L'istruzione "GOTO" fa sì che MSX-BASIC ritorni di nuovo alla riga 10. Il messaggio di errore "Out of DATA in 10" (Mancanza di dati in 10) è provocato dal fatto che non c'erano altri dati disponibili nel file "DATA" quando è stata eseguita l'istruzione "READ" per la quarta volta.

L'istruzione "RESTORE" forza la prima istruzione che segue "READ" a leggere le informazioni dall'istruzione "DATA" dal primo elemento. Aggiungere la riga seguente al precedente programma ed eseguirlo:

```
15 RESTORE
```

Il programma ora continua a ripetere indefinitamente "GEN" fino a che non lo si interrompe premendo simultaneamente i tasti CTRL e STOP. Usare il comando "NEW" per azzerare di nuovo la memoria.

Tabelle con variabili

In una tabella è anche possibile memorizzare variabili, servendosi dell'istruzione "DIM". Esempio:

```
10 DIM A(20)
```

Ciò significa che nella memoria vengono riservate 21 aree (da 0 a 20), tutte contrassegnate dal nome "A". Quando si desidera usare una delle variabili in una riga di programma, il nome da solo ovviamente non è più sufficiente. Occorrerà pertanto indicare quale delle 21 variabili serve. Ciò può essere fatto aggiungendo un numero, disposto tra parentesi, immediatamente dopo il nome della variabile.

```
20 PRINT A(3)
```

Invece di un numero è anche possibile usare il nome di un'altra variabile come nel caso seguente:

```
NEW
10 DIM A(18)
20 A(3)=15:B=0
30 PRINT A(B)
RUN
 0
```

Il risultato fornito da MSX-BASIC è 0 in quanto il contenuto del campo di memoria A(0) è uguale a zero. Ora cambiare come segue la riga di programma numero 20:

```
20 A(3)=15:B=3
```

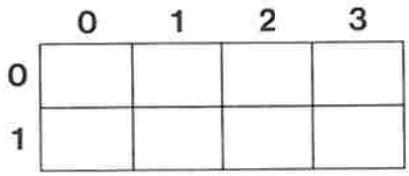
Quando si esegue di nuovo il programma, MSX-BASIC darà come risultato "15", tale essendo il contenuto del campo di memoria A(3).

Tabelle bidimensionali

È anche possibile riservare due tabelle bidimensionali in MSX-BASIC come nell'esempio che segue:

```
10 DIM A(1,3)
```

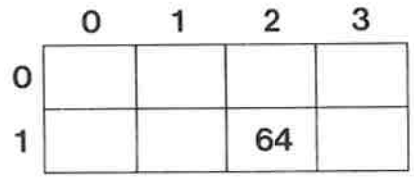
L'istruzione che precede riserva nella memoria l'area seguente:



Facendo riferimento ad una variabile "A" in una riga di programma, devono essere indicate entrambe le dimensioni, ad esempio nel modo seguente:

```
20 A(1,2)=64
```

Questa istruzione inserisce il numero 64 nella riga 1, colonna 2 nell'area riservata della memoria.



Per tutte le variabili numeriche, MSX-BASIC riserverà 11 campi di memoria (da 0 a 10), anche senza l'istruzione "DIM".

È possibile costruire tabelle anche per le variabili alfanumeriche.

Le tabelle usate in un programma attraverso l'istruzione "DIM" possono essere rimosse durante l'esecuzione del programma usando l'istruzione "ERASE". Ecco un esempio:

```
10 DIM A(20)
20 ERASE A
30 DIM A(25)
```

Se si dovesse cancellare da questo esempio la riga di programma 20, MSX-BASIC emetterebbe un messaggio di errore in quanto la tabella "A" è stata riservata due volte. Introducendo l'istruzione "ERASE" nella riga 20, la prima tabella "A" viene cancellata prima dell'apertura della seconda tabella "A".

2.7 Decisioni

L'esecuzione di un programma può essere modificata usando l'istruzione "IF" come nell'esempio che segue:

```
10 INPUT A,B
20 IF A=B THEN PRINT "A=B" ELSE PRINT "A<>B"
```

In questo caso, nella riga di programma 20, vengono confrontati i contenuti delle variabili "A" e "B". Con l'istruzione "IF" viene chiesto a MSX-BASIC se la condizione è vera. In caso affermativo, viene eseguita l'istruzione che segue la parola "THEN". In caso contrario, il programma seguirà le istruzioni identificate dalla parola "ELSE".
La parola "ELSE" e le successive istruzioni della riga di programma, iniziando con l'istruzione "IF" sono facoltative. Se la riga non contiene la parola "ELSE", il programma procede con la successiva riga quando la condizione non è vera.
Se la parola "THEN" è seguita dall'istruzione "GOTO", non si deve aggiungere la parola "THEN". Un'alternativa consiste nel cancellare la parola "GOTO" come illustrato dal successivo esempio:

```
10 INPUT A,B
20 IF A=B THEN 50 ELSE GOTO 40
30 END
40 PRINT "A<>B":GOTO 30
50 PRINT "A=B":GOTO 30
```

Insieme all'istruzione "IF" possono essere incluse le seguenti condizioni:

`<` Minore di

`>` Maggiore di

`=` Uguale

`<>` Diverso da

`>=` Uguale a o maggiore di

`<=` Uguale a o minore di

Se un'istruzione "IF" contiene un calcolo matematico, questo viene eseguito prima di passare all'istruzione condizionale.
Esempio:

```
10 INPUT A,B
20 IF A MOD B=0 THEN PRINT "REST=0"
```

L'istruzione che segue la parola "THEN" viene eseguita soltanto se il risultato del calcolo del modulo "A MOD B" è uguale a 0.

Se si usa la parola "NOT", la condizione può contenere un'ipotesi negativa come illustrato dal successivo esempio:

```
10 INPUT A,B
20 IF NOT A=B THEN PRINT "A<>B"
30 IF A<>B THEN PRINT "A<>B"
```

Le righe di programma 20 e 30 daranno lo stesso risultato.

Combinazione di condizioni

È possibile includere più di una condizione dopo un'istruzione "IF". Il risultato finale dipenderà dalla relazione tra queste condizioni.

1. **Relazione data:** "AND"

condizione 1	condizione 2	risultato
Vera	Vera	Vera
Vera	Non vera	Non vera
Non vera	Vera	Non vera
Non vera	Non vera	Non vera

2. **Relazione data:** "OR"

condizione 1	condizione 2	risultato
Vera	Vera	Vera
Vera	Non vera	Vera
Non vera	Vera	Vera
Non vera	Non vera	Non vera

3. **Relazione data:** "XOR"

condizione 1	condizione 2	risultato
Vera	Vera	Non vera
Vera	Non vera	Vera
Non vera	Vera	Vera
Non vera	Non vera	Non vera

4. **Relazione data:** "EQV"

condizione 1	condizione 2	risultato
Vera	Vera	Vera
Vera	Non vera	Non vera
Non vera	Vera	Non vera
Non vera	Non vera	Vera

5. **Relazione data:** "IMP"

condizione 1	condizione 2	risultato
Vera	Vera	Vera
Vera	Non vera	Non vera
Non vera	Vera	Vera
Non vera	Non vera	Vera

Ecco alcuni esempi:

```
10 IF 3>1 AND 3>2 THEN PRINT "ESATTO"
20 IF 3>1 OR 7<7 THEN PRINT "ESATTO"
```

```
30 IF 3>1 XOR 6<1 THEN PRINT "ESATTO"
40 IF 4<3 EQV 6<7 THEN PRINT "ESATTO"
50 IF 4<3 IMP 3>2 THEN PRINT "ESATTO"
```

Con tutte le suddette righe di programma, vengono eseguite le istruzioni che seguono la parola "THEN".

Confronto di dati alfanumerici

In una condizione possono essere usate non solo le informazioni numeriche ma anche i dati alfanumerici. Il confronto viene quindi eseguito fra il codice dei caratteri di tutti i simboli usati. Si troverà un sommario completo di tutti i codici dei caratteri nell'Appendice E di questo manuale.
Esempio:

```
10 IF "A"<"B" THEN PRINT "A E' MINORE DI B"
20 IF A$<B$ THEN PRINT "IL CONTENUTO DI A$ "E' MINORE DEL CONTENUTO DI B$"
30 IF A$="JOHN" THEN PRINT "IL CONTENUTO DI A$ E' JOHN"
40 IF "JOAN"<"JOHN" THEN PRINT "ESATTO"
```

In una stessa istruzione non si possono combinare istruzioni numeriche ed alfanumeriche.

Iterazioni di programma

Azzerare la memoria del computer battendo il comando "NEW", quindi battere le seguenti righe:

```
10 CT=1
20 PRINT "MSX-BASIC"
30 CT=CT+1
40 IF CT<11 GOTO 20
RUN
```

Quando si esegue questo programma, si vedrà che il testo "MSX-BASIC" compare sullo schermo per 10 volte. C'è un modo più facile per ottenere lo stesso effetto:

```
NEW
10 FOR CT=1 TO 10
20 PRINT "MSX-BASIC"
30 NEXT CT
40 END
RUN
```

Quando si esegue questo programma si ottiene esattamente lo stesso risultato dell'esempio precedente.
Nella riga di programma 10 alla posizione di memoria "CT" viene attribuito il valore 1. Successivamente nella riga di programma 20, viene scritto il testo "MSX-BASIC". Nella riga di programma 30, il valore di "CT" viene aumentato di 1. Quando "CT" diventa maggiore di dieci, viene eseguita la riga di programma numero 40.

Le righe 20 e 30 del programma verranno ripetute fino a che "CT" è minore di 10 o uguale a 10.

Nell'esempio con l'istruzione "NEXT" la variabile "CT" è incrementata del valore 1. Questo valore può essere modificato usando l'opzione "STEP" nell'istruzione "FOR".

A loro volta, i valori in un'istruzione "FOR" possono essere sostituiti da una variabile come nell'esempio che segue:

```
NEW
10 A=4
20 FOR CT=8 TO A STEP -2
30 PRINT "MSX-BASIC"
40 NEXT CT
RUN
```

In questo esempio, il valore della variabile "CT" non è incrementato eseguendo la successiva istruzione "NEXT" bensì decrementato del valore 2 dell'opzione "STEP". L'importante è che il valore iniziale della variabile "CT" sia maggiore del valore finale. Se si esegue il suddetto programma, si vedrà che il testo "MSX-BASIC" viene riprodotto tre volte.

Ordinamento

L'esempio che segue riproduce un programma per ordinare una serie di numeri forniti dall'utente in una determinata sequenza. L'utente deve comunicare al computer quanti sono i numeri coinvolti e quindi batterli.

```
NEW
10 DIM G(100)
20 INPUT "QUANTI NUMERI (TRA 2 E 100)";A
30 IF (A<2)OR(A>100) GOTO 20
40 FOR I=1 TO A
50 PRINT "NUMERO";I;
60 INPUT G(I)
70 NEXT I
80 PRINT "STO ORDINANDO"
90 FOR I=2 TO A
100 X1=1:X2=I
110 X3=X1+INT((X2-X1)/2)
120 IF G(I)<G(X3) THEN 150
130 IF G(I)>G(X3) THEN 170
140 X2=X3
150 IF X2=X3 THEN 190
160 X2=X3:GOTO 110
170 IF X1=X3 THEN 190
180 X1=X3:GOTO 110
190 H=G(I)
200 FOR J=I TO X2+1 STEP -1
210 G(J)=G(J-1)
220 NEXT J
230 G(X2)=H
240 NEXT I
250 PRINT "QUESTI SONO I NUMERI IN SEQUENZA"
260 FOR I=1 TO A
270 PRINT G(I);
280 NEXT I
```

Eseguendo questo programma si vedranno i numeri rapidamente riordinati e messi in sequenza. Se un programma non funziona correttamente, occorre innanzitutto assicurarsi che sia stato battuto correttamente.

Programmi con un menu

Taluni programmi in grado di eseguire numerose funzioni iniziano spesso con un menu di scelte o di opzioni. Il menu presenta un indice delle varie sezioni del programma. Selezionando il numero appropriato viene eseguita la corrispondente sezione del programma. Ecco un esempio di tale programma con menu:

```
NEW
10 PRINT "MENU DELLE OPZIONI"
20 PRINT "1=RIGA PROGRAMMA 100"
30 PRINT "2=RIGA PROGRAMMA 200"
40 PRINT "3=FINE PROGRAMMA"
50 INPUT "FA UNA SCELTA";A
60 IF (A<1)OR(A>3) GOTO 50
70 ON A GOTO 100,200,300
100 PRINT "QUESTA E' LA RIGA PROGRAMMA 100":GOTO 10
200 PRINT "QUESTA E' LA RIGA PROGRAMMA 200":GOTO 10
300 END
RUN
```

La riga più interessante di questo programma è la numero 70. Quando il contenuto della variabile "A" è uguale a 1, il programma continua con la riga 100. Quando è uguale a 2, il programma continua con la riga 200 mentre quando è uguale a 3, viene eseguita la riga di programma 300.

La gestione degli errori

MSX-BASIC consente di controllare e correggere gli errori compiuti durante l'esecuzione di un programma. Ecco un esempio:

```
NEW
10 ON ERROR GOTO 50
20 INPUT "NUMERO";A%
30 END
50 IF ERR=6 THEN PRINT "IL NUMERO DEVE ESSERE COMPRESO TRA
-32768 E +32767":RESUME 0
60 ON ERROR GOTO 0
RUN
```

In questo esempio l'istruzione nella riga 10 fa sì che il programma continui con la riga 50 se viene scoperto un errore. Se l'utente imposta una cifra minore di —32768 o maggiore di +32767 in risposta alla domanda nella riga di programma 20, MSX-BASIC segnala un errore di supero capacità in quanto la variabile "A%" è una variabile intera. L'errore di supero di capacità ha il codice 6 (vedere Appendice A di questo manuale). Alla riga di programma 50 viene posta la domanda se il codice di errore 6 è effettivamente la causa della routine di errore.

In caso affermativo, il testo della riga 50 viene proiettato sullo schermo.
Dopo di che viene usata l'istruzione "RESUME" per continuare il programma iniziando alla riga 20. Alla riga 60, MSX-BASIC interrompe l'esecuzione del programma se la routine di errore è provocata da un errore diverso da quello corrispondente al codice di errore 6.
MSX-BASIC consente inoltre di definire propri codici di errore.
In questo caso bisogna fare in modo che i propri codici di errore siano maggiori di quelli usati da MSX-BASIC. Ecco un esempio:

```
NEW
10 ON ERROR GOTO 100
20 INPUT "FINE PROGRAMMA";A$
30 IF A$="SI" THEN ERROR 250
40 PRINT "ESECUZIONE PROGRAMMA":GOTO 20
50 END
100 IF ERR=250 THEN INPUT "SEI SICURO";A$:IF A$="SI" THEN
RESUME 50 ELSE RESUME 20
110 ON ERROR GOTO 0
RUN
```

Nella riga di programma 30 viene usato il codice di errore 250 quando la variabile "A$" è identica a "SI" ed in tal caso viene eseguita la riga 100 come indicato nella riga 10. Alla riga 100 viene chiesto se la routine di errore è provocata dal codice 250. In caso affermativo, MSX-BASIC chiede all'utente una conferma. Se la risposta dell'utente è positiva, il programma continua usando l'istruzione "RESUME 50" alla riga 50. Se la risposta dell'utente è negativa, il programma continua con la riga 20 usando l'istruzione "RESUME 20".

2.8 Subroutines

Si presuma che in memoria sia caricato il seguente programma:

```
10 INPUT A:INPUT B
20 C=A*100/B:PRINT A;
30 PRINT "E'";C;"PERCENTO DI";
40 PRINT B
50 C=B*100/A:PRINT B;
60 PRINT "E'";C;"PERCENTO DI";
70 PRINT A
80 END
```

Si sarà senza dubbio notato che la riga del programma 60 è identica alla riga 30. Si potrebbe inoltre usare una riga speciale e rinviare ad essa il computer quando raggiunge la riga 30 e la riga 60 con l'istruzione "GOSUB".
In tal caso il programma esemplificativo si presenterebbe come segue:

```
10 INPUT A:INPUT B
20 C=A*100/B:PRINT A;
30 GOSUB 90
40 PRINT B
50 C=B*100/A:PRINT B;
60 GOSUB 90
70 PRINT A
80 END
90 PRINT "E'";C;"PERCENTO DI";
100 RETURN
```

Quando raggiunge la riga 30, MSX-BASIC continua il programma saltando alla riga 90. L'istruzione "RETURN" fa sì che MSX-BASIC ritorni indietro alla riga 40. Lo stesso accade quando raggiunge la riga 60. Quando per contro raggiunge la riga 100, MSX-BASIC ritorna alla riga 70.
Le istruzioni delle righe 90 e 100 sono dette "subroutine". Quando MSX-BASIC arriva alla riga 90 come risultato dell'istruzione "GOSUB", esso ricorda la riga di provenienza e ritorna alla riga immediatamente seguente quando incontra l'istruzione "RETURN".
La riga 80 è di particolare importanza nel programma esemplificativo. Se la si dimenticasse e se la riga 90 seguisse immediatamente la riga 70, MSX-BASIC emetterebbe un messaggio di errore "RETURN without GOSUB" (RETURN senza GOSUB) al raggiungimento della riga 100.

Subroutine a richiesta

Una subroutine può anche essere richiamata dall'indicazione:

1. del contenuto di una variabile
2. dell'ora

3. di uno o più tasti di funzione
4. dei tasti STOP e CTRL
5. di una collisione tra due sprites
6. del pulsante di azione di un comando manuale

Le ultime due indicazioni saranno trattate più in dettaglio nei capitoli 2.12 e 2.14. Ecco una subroutine richiamata dall'indicazione del contenuto di una variabile usando la parola "ON". Ecco un esempio:

```
10 INPUT A
20 IF (A<0)OR(A>255) GOTO 10
30 ON A GOSUB 100,200,300
40 END
100 PRINT "SUBROUTINE 100":RETURN
200 PRINT "SUBROUTINE 200":RETURN
300 PRINT "SUBROUTINE 300":RETURN
```

In questo esempio, la subroutine della riga 100 è eseguita se "A" è uguale a 1. La subroutine della riga 200 è eseguita quando "A" è uguale a 2. Il contenuto della variabile "A" nella riga 30 non può essere negativo o maggiore di 255 e in presenza di queste condizioni MSX-BASIC ripete l'istruzione 10. Invece della variabile "A" potrebbe anche essere indicata una formula matematica, ad esempio:

```
30 ON A+1 GOSUB 100,200,300
```

Quando il contenuto della variabile "A" è 1, verrà eseguita la riga 200. Nel caso di calcoli, vale la stessa regola e cioè il risultato non può essere negativo né maggiore di 255. Quando la variabile "A" o il risultato del calcolo sono maggiori del numero delle righe "GOSUB", viene eseguita l'istruzione successiva. Se, ad esempio, la variabile è "A" + 3, viene eseguita la riga 40, in quanto A + 1 = 4 e sono state immesse soltanto tre righe di subroutine dopo il "GOSUB" nella riga 30.

Se una subroutine è richiamata dall'indicazione dell'ora, dai tasti di funzione e dal tasto STOP unitamente al tasto CTRL, queste funzioni devono essere attivate quando necessario.

```
10 ON STOP GOSUB 100
20 ON INTERVAL=250 GOSUB 200
30 ON KEY GOSUB 300
40 STOP ON:INTERVAL ON:KEY(1) ON
50 GOTO 50
100 PRINT "FINE":END
200 PRINT "PASSATI ALTRI 5 SECONDI"
210 RETURN
300 PRINT "PREMUTO TASTO FUNZIONE 1"
310 RETURN
```

Nel suddetto esempio nelle righe 10, 20 e 30 viene indicato a quali subroutine occorre saltare quando vengono premuti simultaneamente i tasti CTRL e STOP (riga 10), quando il tempo è scaduto (in questo caso la riga 20 con 5 secondi — pari a 250 × 1/50 di secondo) e quando viene premuto il tasto di funzione 1. Nella riga 40 vengono attivate le funzioni richieste nelle righe da 10 a 30. La riga 50 dà l'impressione di essere assolutamente inutile in quanto costringerebbe l'MSX-BASIC a ripeterla in continuazione.

In ogni caso, mentre procede, MSX-BASIC si chiede se è stato premuto uno dei tasti attivati oppure se è scaduto l'intervallo richiesto, perché in tal caso verrebbero eseguite

le rispettive subroutine. Fare particolare attenzione al fatto che la riga della subroutine 100 non è terminata con un "RETURN" ma con un'istruzione "END" (la presenza di un RETURN in questa posizione renderebbe impossibile interrompere il programma senza spegnere il computer!).

Quale giorno?

Nel successivo esempio, introducendo una data specifica, MSX-BASIC determinerà il giorno della settimana in cui cade. Azzerare innanzitutto la memoria con il comando "NEW". Quindi battere il seguente programma:

```
10 DATA DOMENICA,LUNEDI,MARTEDI,MERCOLEDI,GIOVEDI,VENERDI,
SABATO
20 PRINT "IMMETTI LA DATA"
30 INPUT "GIORNO (1-31)";DD
40 IF (DD<1) OR (DD>31) GOTO 30
50 INPUT "MESE (1-12)";MM
60 IF (MM<1) OR (MM>12) GOTO 50
70 INPUT "ANNO (1-3000)";YY
80 IF (YY<1) OR (YY>3000) GOTO 70
90 IF MM>2 GOTO 110
100 YY=YY-1:MM=MM+12
110 H1=INT(YY/100):H2=YY-H1*100
120 H3=INT(2.6001*(MM-2)-.2)+DD+H2+INT(H2/4)+INT(H1/4)-2*H1
130 H3=H3-INT(H3/7)*7+1
140 RESTORE
150 FOR I=1 TO H3:READ H$:NEXT I
160 PRINT "QUESTA DATA CADE IL ";H$
170 END
```

Eseguire questo programma e battere la data richiesta da MSX-BASIC. Se il programma non funziona correttamente, controllare accuratamente di non aver fatto errori nell'immettere le varie istruzioni.

2.9 Funzioni

Oltre ai comandi ed alle istruzioni, MSX-BASIC dispone anche di un grande numero di funzioni, che danno il risultato di un calcolo con una costante o una variabile. Queste funzioni possono essere usate soltanto in combinazione con un'istruzione. Ecco un esempio:

```
NEW
10 PRINT SIN(5)
RUN
-.9589242746631
```

oppure:

```
NEW
10 X=5
20 PRINT SIN(X)
RUN
-.9589242746631
```

Si troverà un sommario completo di tutte le funzioni disponibili nel manuale di riferimento. Esse possono essere divise in diversi gruppi:

1. **Funzioni matematiche**

   ABS, ATN, COS, EXP, INT, LOG, SGN, SIN, SQR, TAN.

2. **Funzioni alfanumeriche**

   ASC, LEFT$, LEN, MID$, RIGHT$, STR$, STRING$, VAL, BIN$, HEX$
   OCT$, INSTR$.

3. **Funzioni di conversione**

   CDBL, CINT, CSNG, FIX.

4. **Funzioni di input**

   INKEY$, INPUT$.

5. **Funzioni di output**

   CHR$, POS, LPOS, SPC, TAB.

6. **Funzioni varie**

   CSLRIN, ERL, ERR, FRE, PEEK, RND, USR, VARPTR, POINT, VPEEK,
   STICK, STRIG, PDL, PAD, PLAY, EOF.

   L'utente può anche definire proprie funzioni usando l'istruzione "DEF FN" (vedere capitolo 2.4).

Un numero casuale

La funzione "RND" è spesso usata per far sì che MSX-BASIC determini una cifra casuale. Tale cifra sarà compresa tra 0 e 1.

Se l'utente desidera una cifra casuale tra 1 e 6 (ad esempio per il lancio di un dado), deve moltiplicare la cifra per 6 ed aggiungere 1 al risultato. Ecco un esempio:

```
NEW
10 A=INT(RND(1)*6+1)
20 PRINT A
RUN
```

Nel suddetto esempio, alla riga 10 è stata usata una combinazione di due funzioni in una sola istruzione. La funzione "INT" crea una cifra intera dal numero selezionato casualmente.

Numeri del Lotto

Il successivo esempio è un programma che sceglie 6 numeri del lotto.

```
NEW
10 X=0
20 FOR I=1 TO 6
30 A=INT(RND(1)*41+1)
50 IF A=X(1) GOTO 30
60 IF A=X(2) GOTO 30
70 IF A=X(3) GOTO 30
80 IF A=X(4) GOTO 30
90 IF A=X(5) GOTO 30
110 X(I)=A
120 NEXT I
130 PRINT "I NUMERI DEL LOTTO SONO";
140 FOR I=1 TO 6:PRINT X(I);:NEXT I:PRINT
160 INPUT "UN ALTRO (Y/N)";A$
170 IF A$="Y" GOTO 10
180 END
```

Eseguire questo programma e vedere quali sono i numeri del lotto che MSX-BASIC produce. Se il programma non funziona, controllare accuratamente che le istruzioni siano state battute correttamente.

Spazio di memoria

La funzione "FRE" consente di vedere quanto spazio è ancora disponibile in memoria. Provare:

```
PRINT FRE(0)
 nnnnn
```

Ciò significa che si avranno ancora nnnnn byte liberi a disposizione di MSX-BASIC.

Se si desidera vedere quanto spazio di memoria è rimasto per le variabili alfanumeriche, usare l'istruzione:

```
PRINT FRE("")
 xxx
```

Ciò significa che ci sono ancora "xxx" byte liberi per le variabili alfanumeriche.

2.10 MSX-BASIC e lo schermo

Con MSX-BASIC lo schermo può essere usato in quattro modi diversi:

1. Modo testo 1
2. Modo testo 2
3. Modo grafico 1
4. Modo grafico 2

Modo testo 1

In questo modo è possibile inserire sullo schermo un massimo di 40 caratteri per riga e lo schermo dispone di 24 di tali righe. Ogni posizione sullo schermo è definita dalla coordinata X (che indica la colonna) e dalla coordinata Y (che indica la riga). La coordinata X più piccola è 0 e corrisponde alla colonna più a sinistra, mentre quella più elevata è 39, che corrisponde alla colonna all'estrema destra. Le coordinate Y vanno da 0 (corrispondente alla riga superiore) fino a 23 (corrispondente alla riga inferiore).
Con l'istruzione "WIDTH" è possibile indicare al computer il numero di caratteri desiderati per riga.
Con l'istruzione "LOCATE" il cursore può essere spostato in qualsiasi posizione desiderata sullo schermo.
Questo modo testo (modo testo 1) può essere attivato con l'istruzione "SCREEN 0".
Esempio:

```
NEW
10 SCREEN 0
20 WIDTH 40
30 LOCATE 15,10:PRINT "HALLO"
RUN
```

In questo esempio nella riga 10 viene attivato il modo testo 1. La riga 20 dà la possibilità di inserire 40 caratteri in una riga.
Nella riga 30 il cursore è spostato alla posizione 15 (16-esima colonna) della riga 10 (11-esima riga) dove viene scritta la parola "HALLO".
Quando MSX-BASIC è a livello comandi oppure deve elaborare un'istruzione "INPUT", lo schermo è sempre nel modo testo.

Modo testo 2

Questo modo testo consente di avere sullo schermo un massimo di 32 caratteri per riga. Anche in questo caso ogni posizione sullo schermo è definita dalla coordinata X per la colonna e dalla coordinata Y per la riga. La coordinata X è un numero compreso tra 0 per la colonna di sinistra e 31 per la colonna di destra. La coordinata Y va da 0 per la riga superiore a 23 per la riga inferiore.
Il numero massimo di caratteri per riga può essere determinato usando l'istruzione "WIDTH".

Usando l'istruzione "LOCATE" il cursore può essere spostato in qualsiasi posizione dello schermo.
Questo modo testo (modo testo 2) è attivato con l'istruzione "SCREEN 1". Ecco un esempio:

```
NEW
10 SCREEN 1
20 WIDTH 32
30 LOCATE 15,10:PRINT "HALLO"
RUN
```

Questo esempio dà esattamente lo stesso risultato del precedente, che descriveva il modo testo 1.
In ogni caso, nella riga 10, è stato attivato il modo testo 2. Nella riga 20 si è creata la possibilità di usare 32 caratteri per riga. Se ci sono difficoltà a leggere i caratteri nel modo testo 1, usare il modo testo 2 di MSX-BASIC.

Ricordarsi che nel modo testo 2 ci sono aree destinate al bordo blu-verde nella parte superiore e nella parte inferiore dello schermo e che è possibile cambiare il colore di queste aree usando l'istruzione "COLOR".

Per ritornare al modo testo 1, battere:

```
SCREEN 0:WIDTH 40
```

Modo grafico 1

In questo modo lo schermo è diviso in 256 punti in senso orizzontale per 192 punti in verticale. La posizione di ciascun punto è determinata da una coordinata X (la colonna) e da una coordinata Y (la riga). La coordinata X può essere qualsiasi numero compreso tra 0 (per la colonna più a sinistra) e 255 (per la colonna più a destra). La coordinata Y può variare da 0 (per la riga superiore) a 191 (per la riga inferiore).
In questo modo, per creare immagini sullo schermo possono essere usate soltanto istruzioni grafiche. Questo modo è attivato con l'istruzione "SCREEN 2". Le istruzioni grafiche sono spiegate nel capitolo 2.11.
Esempio:

```
NEW
10 SCREEN 2
20 CIRCLE (185,115),10
30 GOTO 30
RUN
```

Con questo esempio il computer disegnerà un cerchio sullo schermo, usando il punto di coordinate 185 (la coordinata X) e 115 (la coordinata Y) come suo centro.
In questo caso la riga 30 è di particolare importanza. Usando questa istruzione "GOTO", la riga 30 viene eseguita ripetutamente. Se questa riga mancasse, MSX-BASIC ritornerebbe a livello comandi dopo aver eseguito le righe 10 e 20 e verrebbe riattivato automaticamente il modo testo 1. Considerando la velocità con cui MSX-BASIC opera, non si avrebbe nemmeno il tempo di vedere il cerchio.
È possibile interrompere il programma premendo contemporaneamente i tasti CTRL e STOP.

Modo grafico 2

In questo modo lo schermo è diviso esattamente allo stesso modo del modo grafico 1.

La differenza tra i due è che le figure che compaiono sullo schermo sono tutte costituite da piccoli quadratini, che misurano ciascuna 4x4 punti di immagine.
Il modo grafico 2 è attivato con l'istruzione "SCREEN 3".
Lavorando in questo modo, possono essere usate soltanto le istruzioni grafiche. Si troveranno ulteriori dettagli sulle varie istruzioni grafiche nel capitolo 2.11.
Ecco ora un esempio:

```
NEW
10 SCREEN 3
20 LINE (125,10)-(200,85)
30 GOTO 30
RUN
```

Con questo esercizio verrà disegnata una linea dal punto di coordinate 125,10 (rispettivamente coordinata X e Y), fino al punto di coordinate 200,85 (rispettivamente coordinata X e Y).
La linea è costituita da tanti piccoli quadratini. È possibile interrompere il programma premendo simultaneamente i tasti CTRL e STOP.
Si vedrà ora cosa succede quando si cambia la riga 10 in:

```
10 SCREEN 2
```

Eseguire di nuovo il programma.

Una volta osservata la differenza tra il modo grafico 1 ed il modo grafico 2, interrompere l'esempio premendo contemporaneamente i tasti CTRL e STOP.

Cancellazione dello schermo

È possibile cancellare o "ripulire" lo schermo con l'istruzione "CLS". La si può usare con tutti i modi dello schermo.

2.11 Istruzioni per il colore ed i grafici

MSX-BASIC consente l'uso di 16 colori, numerati da 0 a 15, che possono essere usati come colore di primo piano e di fondo e per colorare le aree del bordo nel modo testo 2 o nei modi grafici 1 e 2.
Nell'Appendice C si troverà una lista completa dei vari colori unitamente ai rispettivi numeri di codice.

Il colore numero 0 (trasparente) è di particolare importanza. Quando viene usato, i colori prevalenti compariranno soltanto nelle aree da esso coperte. Ciò significa che il colore 0 è usato soltanto quando necessario come colore di primo piano. Quando sia il colore di primo piano che il colore di fondo sono trasparenti, l'area si presenta nera.

I colori possono essere scelti usando l'istruzione "COLOR". Azzerare la memoria del computer con il comando "NEW" e battere il seguente esempio:

```
10 SCREEN 1
20 COLOR 1,3,6
30 PRINT "HELLO"
RUN
```

In questo esempio le lettere appaiono nere in primo piano (colore numero 1) contro uno sfondo verde chiaro (colore numero 3). Le aree del bordo sono rosso scuro (colore numero 6).

Una volta osservato questo esempio, battere:

```
SCREEN 0:COLOR 15,4,7
```

Quando si inizia a lavorare con MSX-BASIC, i colori standard nel modo testo 1 sono il bianco come colore del primo piano (colore numero 15) e blu scuro come colore di sfondo (colore numero 4). Negli altri modi dello schermo il colore standard delle aree del bordo sarà blu-verde (colore numero 7).

Disegno di una linea

Con l'istruzione "LINE" è possibile disegnare una linea sullo schermo tra due punti. Questi punti devono essere indicati con le rispettive coordinate X e Y espresse in valori relativi o assoluti. Il valore relativo è indicato dalla presenza della parola "STEP" quando vengono battute le coordinate. Ad esempio, un punto nella colonna 11, riga 16, ha le coordinate assolute (10,15). Presumendo che le ultime coordinate assolute X,Y usate siano (3,6), il punto nella colonna 11, riga 16, (coordinate 10,15) può anche essere indicato come STEP (7,9).

```
NEW
10 SCREEN 2
```

```
20 LINE (125,10).STEP(75,75),1
30 GOTO 30
40 RUN
```

In questo esempio le coordinate del secondo punto sono espresse rispetto al primo punto. Il numero "1" finale nella riga di programma 20 ha determinato il colore della linea, in questo caso nero (colore numero 1). Ciò significa che la riga di programma 20 si sarebbe anche potuta scrivere come segue:

```
20 LINE (125,10)-(200,85),1
```

Aggiungendo una lettera "B" dopo il numero del colore nell'istruzione "LINE" MSX-BASIC produce sullo schermo un rettangolo, di cui la linea rappresenta la diagonale. La linea stessa non compare più. Interrompere il programma premendo simultaneamente i tasti STOP e CTRL. Modificare ora il programma come segue:

```
20 LINE (125,10)-STEP(75,75),1,B
```

ed eseguirlo di nuovo.

Usando la lettera "BF" dopo il numero del colore nell'istruzione "LINE", MSX-BASIC disegna un rettangolo sullo schermo e riempie l'area con il colore indicato nell'istruzione "LINE".
Interrompere il programma premendo simultaneamente i tasti CTRL e STOP e modificare ora la riga 20 in:

```
20 LINE (125,10)-STEP(75,75),1,BF
```

Eseguire ora di nuovo il programma. Per interromperlo, premere simultaneamente i tasti CTRL e STOP.

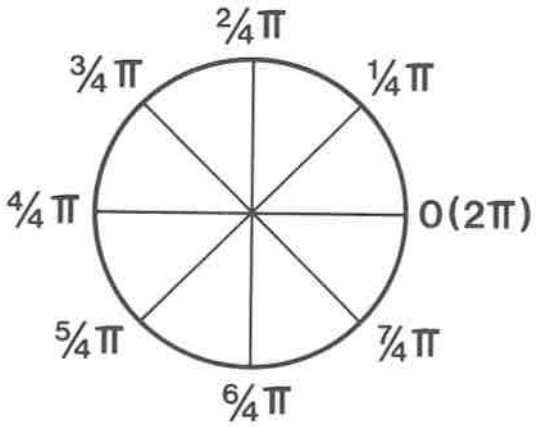
Disegno di un cerchio

Con l'istruzione "CIRCLE" è possibile disegnare un cerchio sullo schermo partendo da un punto specifico, determinato dalle coordinate X e Y. Queste coordinate possono avere un valore assoluto o relativo. Il raggio è indicato in termini di punti di immagine. Ecco un esempio:

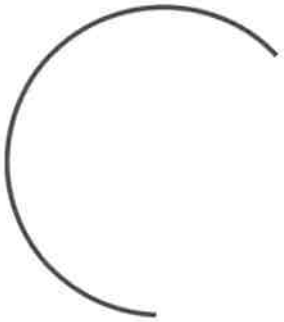
```
NEW
10 SCREEN 2
20 CIRCLE (90,80),20,1
30 GOTO 30
RUN
```

In questo esempio viene disegnato un cerchio nero (colore numero 1) intorno al punto centrale la cui coordinata X è pari a 90 e la cui coordinata Y è pari a 80, con un raggio di 20 punti di immagine (ciò significa ovviamente che il diametro è pari a 40 punti di immagine).
Con l'istruzione "CIRCLE" è inoltre possibile indicare un angolo di inizio ed un angolo di fine. Entrambi sono determinati dall'illustrazione che segue:

π = 3.1415926535898



Si presuma di voler disegnare una parte di un cerchio come indicato nell'illustrazione che segue:



Interrompere per prima cosa il programma dell'esempio precedente premendo simultaneamente i tasti CTRL e STOP. Quindi modificare la riga 20 come segue:

```
20 CIRCLE (90,80),20,1,0.7854,4.7124
```

Eseguire ora di nuovo il programma. Il punto di inizio è calcolato come segue:
3.1416 x 1/4 = 0.7854.
Lo stesso vale per il punto finale, e cioè 3.1416 x 6/4 = 4.7124.
Il punto iniziale ed il punto finale del cerchio possono essere collegati al punto centrale introducendo valori negativi per i punti di inizio e di fine. Interrompere per prima cosa il programma (premere contemporaneamente i tasti CTRL + STOP) e modificare la riga 20 in:

```
20 CIRCLE (90,80),20,1,-0.7854,-4.7124
```

Naturalmente è anche possibile collegare soltanto uno o entrambi i punti con il punto centrale.
MSX-BASIC è in grado di disegnare anche un'ellisse sullo schermo. Per farlo, basta introdurre un numero indicante la proporzione tra il raggio orizzontale ed il raggio verticale dell'ellisse. Interrompere innanzitutto il precedente programma e modificare la riga 20 in:

```
20 CIRCLE (90,80),20,1,,,2
```

Assicurarsi di aver battuto tutte le virgole prima di eseguire di nuovo il programma. In caso contrario, MSX-BASIC considera il "2" che è stato introdotto per indicare la proporzione tra i due raggi come il punto iniziale del cerchio. In questo esempio, il raggio verticale è 20 ed il raggio orizzontale 20 : 2 = 10. Interrompere il programma dopo averlo provato (premere contemporaneamente i tasti CTRL + STOP).

Colorazione

Con MSX-BASIC è anche possibile colorare un'area racchiusa da linee, usando l'istruzione "PAINT". Insieme a questa istruzione deve essere indicata la posizione per mezzo delle coordinate X e Y ed il numero del colore. Queste coordinate possono rappresentare un valore assoluto ma anche un valore relativo. Dal punto iniziale indicato, MSX-BASIC cercherà un'area racchiusa da linee del colore indicato dall'istruzione "PAINT", quindi riempirà l'area con quel colore. Provare con questo esempio:

```
NEW
10 SCREEN 2
20 CIRCLE (50,80),30,1
30 CIRCLE (90,80),30,1
40 PAINT (70,80),1
50 GOTO 50
RUN
```

In questo esempio, MSX-BASIC disegna sullo schermo due cerchi neri che si sovrappongono parzialmente. Con l'istruzione "PAINT" il cursore è disposto nell'area dove entrambi i cerchi si sovrappongono, che è colorata in nero (colore numero 1 nella riga 40). Interrompere di nuovo il programma e modificare la riga 20 in:

```
20 CIRCLE (50,80),30,6
```

Eseguire ora di nuovo il programma. L'istruzione "PAINT" riempirà completamente di nero il cerchio della riga 20, dato che è la sola area racchiusa da linee nere.
Notare che il punto di intersezione delle due righe avrà sempre il colore dell'ultima istruzione grafica. Nell'esempio, il cerchio rosso della riga 20 non forma più una linea ininterrotta. Interrompere per prima cosa il programma e modificare la riga 40 in:

```
40 PAINT (70,80),6
```

Eseguire ora il programma di nuovo. MSX-BASIC colorerà l'intero schermo in rosso, dato che non c'è un'area completamente racchiusa da linee rosse. Interrompere di nuovo il programma (CTRL + STOP).
Nel modo grafico 2 MSX-BASIC può cercare un colore di bordo come nell'esempio seguente:

```
NEW
10 SCREEN 3
20 LINE (125,10)-STEP(75,75),15,B
30 PAINT (150,15),6,15
40 GOTO 40
RUN
```

Qui si vede che il quadrato formato da una linea bianca (dalla riga 20) è colorato di rosso dall'istruzione nella riga 30. Lavorando nel modo grafico 2, l'istruzione "PAINT" può differire dal colore delle linee che racchiudono l'area. Interrompere il programma (CTRL + STOP).

Inserimento di un punto

Per far sì che un singolo punto di immagine compaia in un certo colore, deve essere usata l'istruzione "PSET". L'istruzione "PRESET" è usata per farlo scomparire di nuovo, come si può vedere nel successivo esempio:

```
NEW
10 SCREEN 2
20 CIRCLE (80,80),20,1
30 FOR I=I TO 1000
40 PSET (80,80),15
50 PRESET (80,80)
60 NEXT
RUN
```

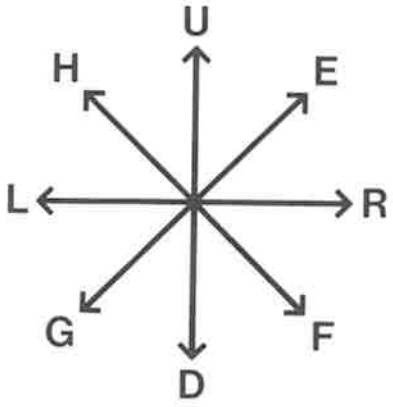
Nell'esempio, il centro del cerchio (riga 20) è indicato da un punto di immagine lampeggiante. Lavorando nel modo grafico 2, tale punto sarebbe composto da 4x4 punti di immagine.

Disegno

Con MSX-BASIC e con l'istruzione "DRAW" è possibile disegnare sullo schermo. "DRAW" prevede parecchi subcomandi:

1. *Subcomando per disegnare una linea*
   Iniziando dalla posizione del cursore, è possibile disegnare una linea in otto diverse direzioni:

   Un = Nord
   Dn = Sud
   Ln = Ovest
   Rn = Est
   En = Nord-Est
   Fn = Sud-Est
   Gn = Sud-Ovest
   Hn = Nord-Ovest



   La "n" in questi subcomandi determina la lunghezza della linea in termini di punti di immagine. Esempio:

```
NEW
10 SCREEN 2
20 PSET (80,80),15
30 DRAW "E50"
40 GOTO 40
RUN
```

   Si vedrà che MSX-BASIC disegna una linea retta, diagonalmente, in direzione Nord-Est. Notare che l'istruzione "DRAW" è considerata come valore alfanumerico, il che significa che occorre usare le virgolette.

   Per disegnare una linea dalla posizione del cursore ad un'altra posizione sullo schermo, occorre usare il subcomando "MXY", in cui X e Y sono le coordinate del punto

al quale deve arrivare la linea.
Queste coordinate X e Y possono avere un valore assoluto o un valore relativo. Se vengono usati i valori relativi, occorre farli precedere da un simbolo più (+) o meno (—). I successivi due esempi hanno lo stesso effetto.
Innanzitutto interrompere il programma corrente (CTRL + STOP), quindi battere:

```
NEW
10 SCREEN 2
20 PSET (80,80),15
30 DRAW "M90,75"
40 GOTO 40
RUN
```

Interrompere il programma e modificare la riga 30 come segue:

```
30 DRAW "M+10,-5"
```

Eseguire ora di nuovo il programma.

I subcomandi precedentemente usati possono essere preceduti dai subcomandi N e B. Se si usa il subcomando N, il cursore ritornerà alla sua posizione originale dopo l'esecuzione dell'istruzione "DRAW". Interrompere il programma (CTRL + STOP) e battere dapprima il seguente esempio:

```
NEW
10 SCREEN 2
20 PSET (80,80),15
30 DRAW "R10D10"
40 GOTO 40
RUN
```

In questo esempio MSX-BASIC disegnerà per prima cosa una linea verso destra (R10) e successivamente da quel punto una linea verso il basso (D10). Interrompere il programma e modificare la riga 30 come segue:

```
30 DRAW "NR10D10"
```

Eseguire di nuovo il programma. Si vedrà stavolta che il computer per prima cosa traccia una linea verso destra (R10) dopo di che il cursore ritorna alla sua posizione di partenza originale prima di disegnare una seconda linea verso il basso (D10). Se si usa il subcomando "B", il cursore viene spostato sulla posizione desiderata ma non viene disegnata nessuna linea. Interrompere il programma e battere le seguenti righe:

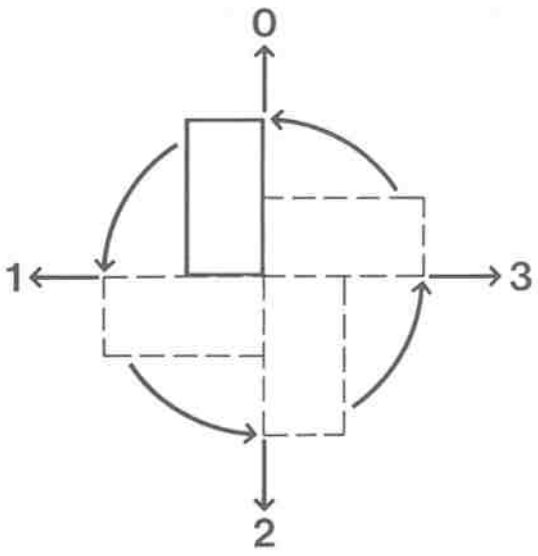
```
NEW
10 SCREEN 2
20 DRAW "BM80,80R10D10"
30 GOTO 30
RUN
```

2. *Subcomando per il colore*
Con il subcomando "Cn" viene indicato in quale colore il disegno deve essere eseguito. Il colore è indicato dalla "n", che è il numero del colore, come illustrato dall'Appendice C.

3. *Subcomando per disegnare determinate posizioni angolari*

Con il subcomando "An" è possibile decidere sotto quale angolo il disegno deve essere eseguito. "n" è un numero compreso tra 0 e 3.

0 = 0 gradi
1 = 90 gradi
2 = 180 gradi
3 = 270 gradi



4. *Subcomando per la scala*

Il subcomando "Sn" è usato per istruire il computer ad allargare o ridurre la dimensione del disegno. Qui "n" è un numero compreso tra 0 e 255. Il fattore di scala è diviso per quattro. La dimensione effettiva o lunghezza con i subcomandi "U", "D", "L", "R", "E", "F", "G", "H" ed il relativo subcomando "M" è la dimensione originariamente programmata moltiplicata per il fattore di scala. Ad esempio, l'istruzione: DRAW "S8R10" produce una linea verso destra, non di lunghezza pari a 10 punti di immagine, come indicato dal subcomando "R10" ma con una lunghezza di 20 punti di immagine (10 x 8 : 4 = 20) come risultato del subcomando "S8".

5. *Esecuzione di subcomandi*

Con MSX-BASIC tutti i subcomandi possono essere inseriti in una variabile alfanumerica in modo da poter usare la variabile stessa in combinazione con l'istruzione "DRAW". Interrompere il precedente programma (CTRL + STOP) e battere il seguente esempio:

```
NEW
10 SCREEN 2
20 A$="E20R20G20L20"
30 DRAW "A0S4BM80,80XA$;"
40 GOTO 40
RUN
```

Questo programma produrrà sullo schermo un rombo. I necessari subcomandi si possono trovare nella variabile "A$" che viene eseguita dall'istruzione "DRAW" dal subcomando "X". Occorre inserire un punto e virgola (;) dopo la variabile.

Tutte le costanti usate nei subcomandi dell'istruzione "DRAW" potranno essere sostituite da variabili, usando la formula seguente: "=nome variabile;"
Ad esempio:
```
X1=10:X2=5
DRAW "M+=X1;,-=X2;"
```

L'istruzione "DRAW" prevede molte possibilità, che sarà bene provare tutte. Ecco un altro esempio. Interrompere il programma corrente (CTRL + STOP) e battere:

```
NEW
10 SCREEN 2:COLOR 15,4,4
20 PSET (127,95),1
30 FOR I=I TO 189 STEP 4
40 A$="L"+STR$(I)+"D"+STR$(I+1)+"R"+STR$(I+2)+"U"+STR$(I+3)
50 DRAW "S4XA$;"
60 NEXT I
70 CLS:GOTO 20
RUN
```

È possibile interrompere il programma come al solito premendo simultaneamente i tasti CTRL e STOP.

2.12 Gli sprites

Con MSX-BASIC è possibile creare sullo schermo oggetti in movimento e cioè creare effetti di animazione. In gergo, questi oggetti sono detti "sprites". Una delle tipiche proprietà degli sprites è la possibilità di apparire in movimento in posizione più o meno avanzata sullo schermo. L'illustrazione di fondo, ad esempio, impostata con le istruzioni grafiche del capitolo precedente, rimane intatta.

Per usare gli sprites, occorre soddisfare due condizioni:

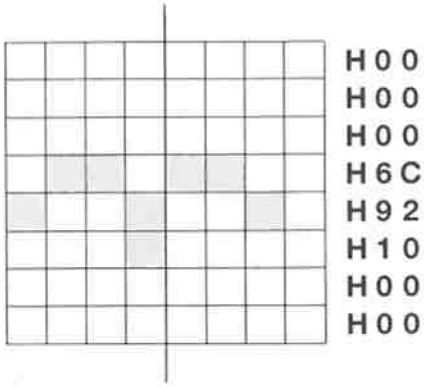
1. Lo sprite deve essere definito, ossia occorre determinare in altre parole a quale oggetto deve assomigliare.

2. Occorre decidere dove compariranno gli sprites sullo schermo.

Definizione degli sprites

Gli sprites possono essere definiti in formato grande o piccolo. Un formato piccolo significa che gli sprites saranno composti da 8x8 punti di immagine. Usando sprites di formato piccolo, è possibile definirne un massimo di 256. La definizione di uno sprite è espressa nella variabile "SPRITE$".

Si supponga di voler definire il seguente sprite:

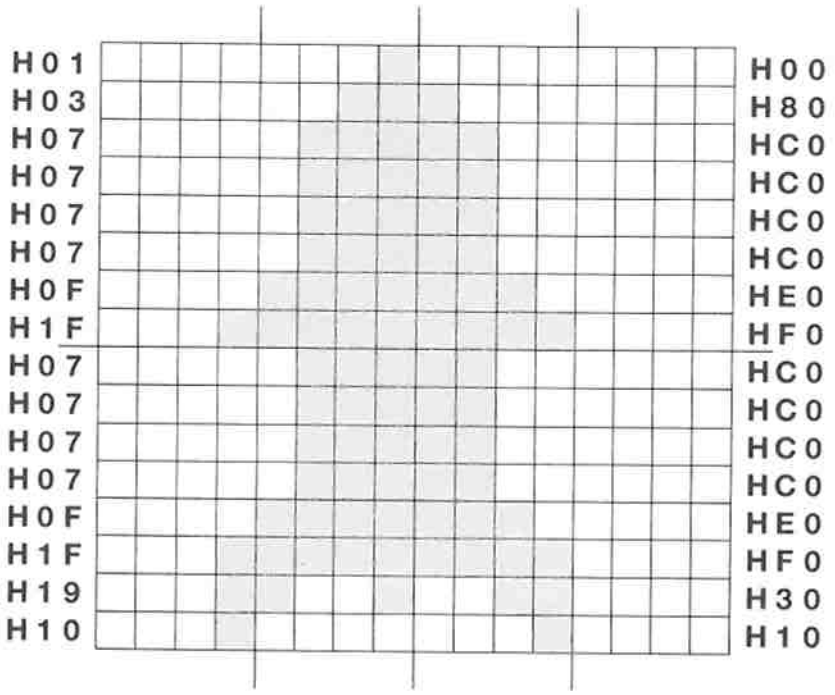


Ogni otto punti di immagine orizzontali formano un byte, che viene diviso in due gruppi di 4 bit. Questi sono definiti in forma esadecimale come indicato alla destra delle illustrazioni. La dimensione dello sprite è indicata nell'istruzione "SCREEN" che segue il modo dello schermo.
Il suddetto sprite potrebbe essere codificato come segue:

```
10 SCREEN 2,0
20 A$=CHR$(&H00)+CHR$(&H00)+CHR$(&H00)+CHR$(&H6C)+
CHR$(&H92)+CHR$(&H10)+CHR$(&H00) +CHR$(&H00)
30 SPRITE$(1)=A$
```

Gli sprites possono anche essere definiti in formato grande, nel qual caso ognuno di tali sprites si comporrà di 16x16 punti di immagine.

Usando sprites di formato grande, è possibile definirne un massimo di 64. Si supponga di volere creare uno sprite come indicato in questa illustrazione:



Ogni 16 punti orizzontali di immagine formano 2 byte, ognuno dei quali viene diviso in due gruppi di 4 bit ciascuno. Questi sono codificati in formato esadecimale. In MSX-BASIC, i byte sono in sequenza verticale. Lo sprite dell'illustrazione potrebbe essere codificato come segue:

```
10 SCREEN 2,2
20 A$=CHR$(&H01)+CHR$(&H03)+CHR$(&H07)+CHR$(&H07)+
CHR$(&H07)+CHR$(&H07)+CHR$(&H0F)+CHR$(&H1F)
30 B$=CHR$(&H07)+CHR$(&H07)+CHR$(&H07)+CHR$(&H07)+
CHR$(&H0F)+CHR$(&H1F)+CHR$(&H19)+CHR$(&H10)
40 C$=CHR$(&H00)+CHR$(&H80)+CHR$(&HC0)+CHR$(&HC0)+
CHR$(&HC0)+CHR$(&HC0)+CHR$(&HE0)+CHR$(&HF0)
50 D$=CHR$(&HC0)+CHR$(&HC0)+CHR$(&HC0)+CHR$(&HC0)+
CHR$(&HE0)+CHR$(&HF0)+CHR$(&H30)+CHR$(&H10)
60 SPRITE$(1)=A$+B$+C$+D$
```

Gli sprites possono anche essere codificati in un modo diverso, ad esempio per lo sprite in formato piccolo:

```
10 SCREEN 2,0
20 DATA 0,0,0,108,146,16,0,0
30 A$=""
40 FOR I=1 TO 8
50 READ A:A$=A$+CHR$(A)
60 NEXT I
70 SPRITE$(1)=A$
```

In questo esempio il valore decimale di otto byte è memorizzato in un'istruzione "DATA". Questi valori sono letti nella riga 50 e memorizzati nella variabile "A$" come valori alfanumerici.
Lo sprite di formato grande potrebbe essere codificato come segue:

```
10 SCREEN 2,2
20 DATA 1,3,7,7,7,7,15,31
30 DATA 7,7,7,7,15,31,25,16
40 DATA 0,128,192,192,192,192,224,240
50 DATA 192,192,192,192,224,240,48,16
60 A$=""
70 FOR I=1 TO 32
80 READ A:A$=A$+CHR$(A)
90 NEXT I
100 SPRITE$(1)=A$
```

In questo esempio, il valore decimale di tutti i 32 byte è memorizzato in 4 istruzioni "DATA". Questi valori sono letti nella riga 80 e memorizzati nella variabile "A$" come valori alfanumerici.

Inserimento degli sprites sullo schermo

Nell'istruzione "SCREEN" viene inserito un codice immediatamente dopo il modo schermo, che indica quali tipi di sprites saranno usati. Il codice usato ha il seguente significato:

0 = piccoli sprites, formato 8x8
1 = piccoli sprites, allargati al formato 16x16
2 = grandi sprites, formato 16x16
3 = grandi sprites, allargati al formato 32x32

Nella seguente riga di programma si indica per prima cosa il modo dello schermo (2), seguito dall'informazione relativa al formato degli sprites, in questo caso 32x32.

```
10 SCREEN 2,3
```

Non lo si è finora detto ma è evidente da quanto sopra che gli sprites possono essere allargati al doppio della dimensione originariamente definita.

Ci sono alcune limitazioni che riguardano l'uso degli sprites:

a. Gli sprites non possono essere usati nei modi testo 1 e 2
b. Non è possibile usare una combinazione di dimensioni di sprites diverse
c. È possibile inserire simultaneamente su una riga orizzontale dello schermo un massimo di 4 sprites.

Uno sprite viene inserito sullo schermo attraverso l'istruzione "PUT SPRITE", seguita dalle seguenti informazioni:

a. Numero priorità
b. Posizione sullo schermo
c. Colore dello sprite
d. Numero dello sprite.

Il numero di priorità deve essere compreso fra 0 e 31 ed ha il seguente significato: Quando due sprites sono inseriti nella stessa posizione sullo schermo, lo sprite con la priorità minore precederà quello con il numero di priorità maggiore.

La posizione sullo schermo è indicata dalle coordinate X e Y. La coordinata X può essere un numero da —32 a 255 e la coordinata Y un numero da —32 a 191. Ciò consente di disporre uno sprite al di fuori dello schermo. Le coordinate X e Y indicano la posizione del punto più in alto e più a sinistra dello sprite.
Anche in questo caso le coordinate possono avere un valore relativo o assoluto. Quando la coordinata Y ha un valore di 209, lo sprite scompare dallo schermo.

Il colore dello sprite è indicato dal numero di codice del colore (vedere Appendice C di questo manuale).

Il numero dello sprite infine determina quale sarà lo sprite che verrà disposto sullo schermo.

Interrompere il programma corrente (premere i tasti CTRL + STOP) ed impostare il seguente esempio:

```
NEW
10 SCREEN 2,0
20 DATA 0,3,31,63,255,255,63,7
30 DATA 0,128,248,255,255,254,252,192
40 DATA 0,0,0,108,146,16,0,0
50 FOR I=1 TO 3
60 A$=""
70 FOR J=1 TO 8
80 READ A:A$=A$+CHR$(A)
90 NEXT J
100 SPRITE$(I)=A$
110 NEXT I
120 PUT SPRITE 4,(180,50),15,1
130 PUT SPRITE 5,(188,50),15,2
140 FOR I=-32 TO 212
150 PUT SPRITE 3,(I,50),1,3
160 NEXT I
170 PUT SPRITE 3,(I,209)
180 FOR I=212 TO -32 STEP -1
190 PUT SPRITE 3,(I,50),1,3
200 NEXT I
210 GOTO 140
RUN
```

Se il programma è stato compilato correttamente, si vedrà comparire un uccello nero che vola attraverso una nube bianca. La nube bianca è stata definita con due sprites (sprite numero 1 e 2) mentre l'uccello nero è stato definito con lo sprite numero 3.
La nube bianca compare sullo schermo come conseguenza delle istruzioni nelle righe di programma 120 e 130.

L'uccello nero si muove da sinistra verso destra come indicato dall'iterazione "FOR. . .NEXT" nelle righe di programma da 140 a 160. L'iterazione "FOR. . .NEXT" nelle righe da 180 a 200 fa sì che l'uccello torni indietro, spostandosi da destra verso sinistra, ecc.

Interrompere il programma (premere contemporaneamente i tasti CTRL + STOP) e modificare la riga 10 in:

```
10 SCREEN 2,1
```

Ora eseguire di nuovo il programma. La dimensione degli sprites si è raddoppiata, come è facile constatare.

Interrompere di nuovo il programma (premere i tasti CTRL + STOP) e modificare la riga 190 in:

```
190 PUT SPRITE 6,(I,50),1,3
```

Eseguire ora il programma ancora una volta. L'uccello sta sempre volando da sinistra verso destra ma ora scompare dietro la nuvola nel viaggio di ritorno, dato che gli è stata impartita una priorità minore rispetto alla nuvola nella riga 130.

Collisione di sprites

MSX-BASIC consente di eseguire una data subroutine quando due sprites entrano in collisione. Ciò avviene attraverso l'istruzione "ON SPRITE GOSUB".

Interrompere il programma corrente (premere contemporaneamente i tasti CTRL + STOP) e modificare l'ultimo programma come segue:

```
5 SCREEN 2,0
10 ON SPRITE GOSUB 300
135 SPRITE ON
210 GOTO 135
300 PUT SPRITE 3,(I,209)
310 FOR J=50 TO 191
320 PUT SPRITE 3,(I,J),1,3
325 NEXT J
330 I=0:SPRITE OFF
340 RETURN 140
```

Ora eseguire di nuovo il programma. La riga 10 indica ora a quale subroutine MSX-BASIC deve saltare quando i due sprites si toccano. Nella riga 135 è attivata la funzione di collisione. Nella subroutine delle righe da 300 a 340 l'uccello è diretto verso il basso da un'iterazione "FOR...NEXT". Nella riga 330 la funzione di collisione viene disattivata e la riga 340 istruisce il computer ad interrompere la subroutine, ritornando alla riga 140 del programma principale.

È possibile interrompere di nuovo il programma premendo contemporaneamente i tasti CTRL e STOP come fatto in precedenza.

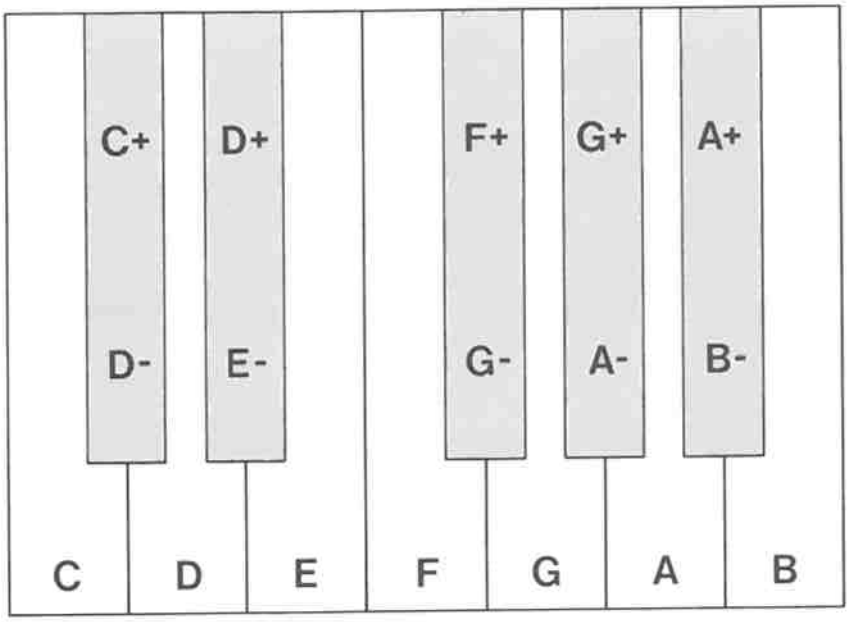
2.13 Musica

MSX-BASIC dispone di un'istruzione speciale — "PLAY" — quando si desidera usare il computer per fare musica. Dopo questa istruzione è possibile immettere i toni per tre voci. La musica per ciascuna di queste tre voci è immessa attraverso un certo numero di subcomandi, analogamente a quanto avveniva per l'istruzione "DRAW".

Nota: Negli esempi di programma che seguono, le note musicali verranno indicate con la notazione anglosassone, di cui viene data qui di seguito la corrispondenza con la notazione italiana:
DO = C; RE = D; MI = E; FA = F; SOL = G; LA = A; SI = B.

1. *Subcomando per suonare note di un'ottava*

   Per indicare quali note devono essere suonate, vengono usate le lettere da A a G (vedere sopra). Un segno più (+) alza la nota di un semitono (A+ = A diesis), mentre un segno meno (—) abbassa la nota di un semitono (A- = A bemolle).



   Tutto funziona basandosi sulla struttura della tastiera del pianoforte, il che significa che il segno "+" può essere usato soltanto quando il corrispondente tasto sul pianoforte è seguito da un tasto nero ed il segno "—" può essere usato soltanto quando il corrispondente tasto sul pianoforte è preceduto da un tasto nero.
   Le combinazioni tipo "F-" oppure "E+" non sono ammesse.
   Esempio:

```
PLAY "CEG"
```

2. *Subcomando per l'impostazione di un'ottava*

L'altezza di una nota è impostata da un subcomando "On", in cui "n" può assumere qualsiasi valore da 1 a 8. Un'ottava standard (tipo quella del pianoforte) va da "C" a "B". Esempio:

```
PLAY "O5CEGO4CEGO5CEG"
```

3. *Subcomando per suonare le note*

Invece di immettere un'ottava più una nota, è anche possibile immettere un numero di nota con il subcomando "Nn", in cui "n" è qualsiasi numero compreso tra 0 e 96. I seguenti due esempi hanno esattamente lo stesso effetto:

```
PLAY "O3CEG"
PLAY "N24N28N31"
```

4. *Subcomando per la lunghezza della nota*

Il subcomando "Ln" è usato per determinare la lunghezza o durata di una nota. In questa combinazione, "n" è un numero compreso tra 1 e 64.
Il numero ha il seguente significato:
1 = una nota intera
2 = metà di una nota
3 = un terzo di una nota
4 = un quarto di una nota
ecc.
Esempio:

```
PLAY "L8DEL4DCE"
```

5. *Subcomando per le pause*

Usando il subcomando "Rn" si introduce una pausa nell'esecuzione. Anche qui "n" è un numero compreso tra 1 e 64 ed ha gli stessi valori attribuiti al parametro "n" del precedente subcomando "Ln".

6. *Subcomando per cambiare la durata di una nota*

Se si inserisce un punto (.) dopo una nota, la sua durata viene aumentata della metà di quella originale. È anche possibile usare più di un punto dopo una nota. Esempio:

```
PLAY "A. . ."
```

In questo esempio la nota "A" avrà una durata pari a 27/8 volte quella normale (= 3/2x3/2 x 3/2).

7. *Subcomando per il tempo*

Il tempo è determinato dal subcomando "Tn", in cui "n" è un numero compreso tra 32 e 255 che rappresenta il numero di note da un quarto che è possibile suonare in un minuto.

8. *Subcomando per il volume*

Il volume di un tono è deciso dal subcomando "Vn", dove "n" rappresenta un numero compreso tra 0 e 15, con 15 che corrisponde al volume più elevato.
Esempio:

```
PLAY "V15CGR1V9CG"
```

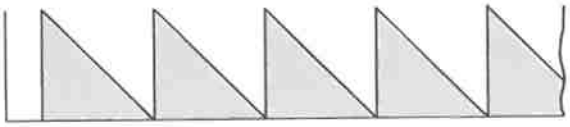
9. *Subcomando per il profilo o forma d'onda del suono*

Il subcomando "Sn" è usato per scegliere il profilo o forma d'onda del tono che viene eseguito. "n" può assumere un valore tra 0 e 15:

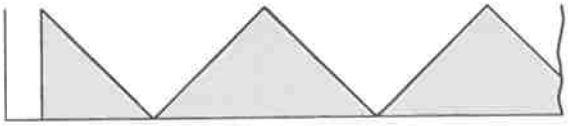
0 — 3 e 9

4 — 7 e 15

8

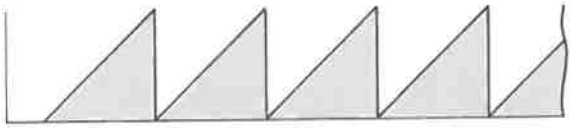


10



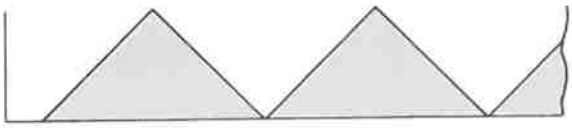
11

12



13

14

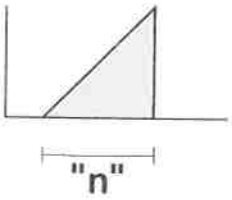


10. *Subcomando per la lunghezza del profilo o forma d'onda del suono*

"Mn" è il subcomando usato per scegliere la lunghezza del profilo o forma d'onda del tono che viene eseguito, dove "n" rappresenta qualsiasi valore compreso tra 1 e 65535.
Notare la differenza negli esempi seguenti:

```
PLAY "S15M500C"
PLAY "S15M1000C"
PLAY "S12M500C"
PLAY "S12M1000C"
```

Più basso è il numero dopo "M", più breve è il profilo.



11. *Esecuzione dei subcomandi*

MSX-BASIC rende possibile inserire tutti i subcomandi in una variabile alfanumerica e successivamente fare riferimento a questa variabile nell'istruzione "PLAY". L'ultimo esempio nella suddetta serie potrebbe anche essere scritto come segue:

```
NEW
10 A$="S12M1000C"
20 PLAY "XA$;"
RUN
```

Immediatamente dopo il nome della variabile occorre inserire un punto e virgola.

Tutte le costanti usate nei subcomandi dell'istruzione "PLAY" possono essere sostituite dalla notazione seguente: "=nome variabile;". Ecco un esempio:

```
X1=15
PLAY "N=X1;"
```

L'istruzione "PLAY" offre molte possibilità che sarebbe utile verificare. Ecco un altro esempio:

```
NEW
5 CLEAR 1000
10 A$="V8S1M9000T250L405"
20 B$="F+F+8E2R8EE8D2R8R4F+F+F+G2F+"
30 C$="O4BO5DDO4BO5D2EO4BO5D"
40 D$="O4B.O5DR4.B.A.DD1R2."
50 H$=B$+".R8"+B$+"O4A"+C$+"DDD8C+2R4O4A"+C$+".D8D.D8E.R8R2"+
B$+".R8"+D$
60 I$="R1R1R4AAAB2A.R8R1R1R4AAAB2AR1R1R1R1R1R1R1R1R1R1R4R8AAAB2A"
70 PLAY "XA$;","XA$;"
80 PLAY "XH$;","XI$;"
RUN
```

Riconosciuto il motivo?

Con la funzione "PLAY" (da non confondersi con l'istruzione "PLAY"!) è possibile controllare se una data istruzione "PLAY" è stata eseguita completamente. Provare questa funzione aggiungendo le righe seguenti al precedente programma:

```
85 CLS
90 PRINT PLAY(1):GOTO 90
```

ed eseguire il programma di nuovo.

Mentre viene suonata la melodia, sullo schermo viene scritto il valore —1. È possibile interrompere il programma premendo simultaneamente i tasti CTRL e STOP.

Con l'istruzione "SOUND" può anche essere inserito un valore specifico in uno dei registri del chip del suono. Usare questa istruzione soltanto se si ha familiarità con il modo in cui funziona il chip del suono.

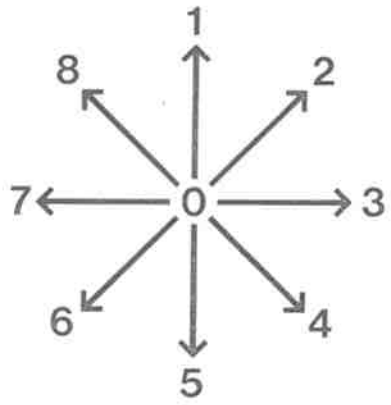
2.14 MSX-BASIC ed i joysticks

Al computer MSX è possibile collegare uno o due joysticks, facendo in modo che MSX-BASIC reagisca alla direzione in cui il joystick viene spostato. Lo stesso effetto si ottiene agendo sui tasti di comando del cursore. La direzione del joystick può essere controllata con la funzione "STICK(n)"; "n" deve essere un numero compreso tra 0 e 2 ha il seguente significato:

0 = I tasti di comando del cursore sono usati come joystick
1 = Joystick collegato al connettore 1
2 = Joystick collegato al connettore 2

Il valore ottenuto con la funzione "STICK(n)" è compreso tra 0 e 8:

0 = Neutro
1 = Nord
2 = Nord-Est
3 = Est
4 = Sud-Est
5 = Sud
6 = Sud-Ovest
7 = Ovest
8 = Nord-Ovest



Ecco un esempio:

```
NEW
10 A=STICK(0)
20 IF A=STICK(0) THEN 20
30 PRINT STICK(0)
40 GOTO 10
RUN
```

In questo esempio la variabile "A" nella riga 10 è resa uguale al valore dei tasti di comando del cursore. Fintantoché non viene premuto alcun tasto di comando del cursore, il valore di "A" sarà 0. Quando viene premuto uno dei tasti di comando del cursore, verrà eseguita la riga 30. Provare con tutti i tasti di comando del cursore. È possibile interrompere il programma nel modo solito premendo simultaneamente CTRL e STOP.

A MSX-BASIC si può anche chiedere se è stato premuto il pulsante di azione del comando manuale (lo stesso effetto del pulsante di azione si ha premendo la barra di spazio della tastiera). La richiesta si effettua attraverso la funzione "STRIG(n)", in cui

"n" è un numero da 0 a 4 e che assume il seguente significato:

0 = La barra di spazio è il pulsante di azione
1 = Pulsante di azione 1 del comando manuale collegato al connettore 1
2 = Pulsante di azione 1 del comando manuale collegato al connettore 2
3 = Pulsante di azione 2 del comando manuale collegato al connettore 1
4 = Pulsante di azione 2 del comando manuale collegato al connettore 2

Il valore ottenuto con la funzione "STRIG(n)" è 0 quando non viene premuto il pulsante di azione e —1 quando viene premuto il pulsante di azione.
Esempio:

```
NEW
10 IF STRIG(0)=-1 THEN PRINT STRIG(0)
20 GOTO 10
RUN
```

Nella riga 10 MSX-BASIC controlla se è stata premuta la barra di spazio, scrivendo sullo schermo il valore di "STRIG(0)".
Si può interrompere il programma premendo simultaneamente i tasti CTRL e STOP.

Lancio di un razzo

Nell'esempio che segue viene lanciato un razzo premendo la barra di spazio. Battere il seguente programma facendo attenzione a non commettere errori:

```
NEW
10 SCREEN 2,3:COLOR 1,4,1
20 DATA 0,0,0,0,3,7,15,63
30 DATA 63,127,127,63,31,1,0,0
40 DATA 0,0,0,0,192,252,255,255
50 DATA 255,255,255,255,255,255,28,0
60 DATA 0,0,0,1,15,31,255,255
70 DATA 255,255,255,255,255,255,30,0
80 DATA 0,0,0,128,224,248,248,252
90 DATA 252,248,248,240,240,224,0,0
100 DATA 0,0,0,28,126,127,255,127
110 DATA 31,15,7,3,0,0,0,0
120 DATA 0,0,0,0,96,240,248,252
130 DATA 254,254,252,252,248,0,0,0
140 DATA 0,0,0,0,0,0,3,15
150 DATA 63,255,255,127,63,31,0,0
160 DATA 0,0,0,0,0,248,252,254
170 DATA 254,252,252,252,252,248,248,0
180 DATA 0,0,0,0,0,0,0,0
190 DATA 0,0,0,0,0,0,0,0
200 DATA 3,3,7,7,7,7,7,15
210 DATA 15,15,15,31,31,31,31,31
220 DATA 0,0,0,0,0,0,0,0
230 DATA 0,0,0,0,0,0,0,0
240 DATA 255,255,255,16,40,16,0,0
250 DATA 0,0,0,0 0,0,0,0
260 DATA 255,255,255,0,1,0,0,0
270 DATA 0,0,0,0,0,0,0,0
280 DATA 240,240,240,128,64,128,0,0
```

```
290 DATA 0,0,0,0,0,0,0,0
300 DATA 8,28,28,62,62,62,127,255
310 DATA 62,62,62,62,127,255,201,128
320 DATA 0,0,0,0,0,0,0,128
330 DATA 0,0,0,0,0,128,128,128
340 DATA 8,28,28,62,62,62,62,28
350 DATA 28,20,0,0,0,0,0,0
360 DATA 0,0,0,0,0,0,0,0
370 DATA 0,0,0,0,0,0,0,0
380 FOR J=1 TO 9:A$=""
390 FOR I=1 TO 32:READ A:A$=A$+CHR$(A):NEXT I
400 SPRITE$(J)=A$:NEXT J
410 CIRCLE (88,120),16,6,-0.0001,-3.415,0.5
420 PAINT (88,118),6
430 LINE (0,192)-(255,120),12,BF
440 PUT SPRITE 1,(80,16),15,1
450 PUT SPRITE 2,(112,16),15,2
460 PUT SPRITE 10,(160,80),15,3
470 PUT SPRITE 11,(16,48),15,4
480 I=120:J=144:GOSUB 1000
490 IF STRIG(0)=-1 GOTO 2000
500 IF STICK(0)=3 THEN GOSUB 3000
510 IF STICK(0)=7 THEN GOSUB 4000
520 GOTO 490
1000 PUT SPRITE 5,(I,J),1,5:J=J+32
1010 PUT SPRITE 6,(I,J),1,6:I=I+32
1020 PUT SPRITE 7,(I,J),1,7:J=J-32
1030 PUT SPRITE 8,(I,J),7,8:I=I-32
1040 RETURN
2000 I=I+32:FOR K=144 TO -33 STEP -1
2010 PUT SPRITE 8,(I,K),7,8:M=K+32
2020 PUT SPRITE 9,(I,M),9,9
2025 FOR T=1 TO 1:NEXT T
2030 PUT SPRITE 9,(I,209)
2040 NEXT K:FOR K=1 TO 1000:NEXT K:COLOR 15,4,7:END
3000 I=I+1:IF I)200 THEN I=200
3010 GOSUB 1000:RETURN
4000 I=I-1:IF I(0 THEN I=0
4010 GOSUB 1000:RETURN
```

Ora eseguire il programma. Agendo sui tasti di comando del cursore, è possibile spostare la piattaforma di lancio a sinistra e a destra. Il razzo parte quando si preme la barra di spazio. Se il programma non funziona nel modo previsto assicurarsi che tutte le istruzioni siano state battute correttamente.

Capitolo 3

L'hardware MSX

3.1 Manutenzione

È possibile pulire il VG8000 con un panno asciutto. Non usare detergenti chimici!

Se si usa un registratore dati, le testine del registratore devono essere pulite regolarmente seguendo le istruzioni fornite con il registratore stesso. Sarà inoltre necessario far ricontrollare le testine del registratore dopo averlo usato per qualche tempo per accertarsi che si trovino nella posizione corretta.

Aver cura di riporre le cassette in un posto fresco, lontano dalla luce diretta del sole e da altre fonti di calore e di tenerle lontane dai campi magnetici. In assenza di queste precauzioni c'è il rischio di perdere le informazioni memorizzate sul nastro della cassetta.

Se il VG8000 non funziona correttamente (se si ode uno strano rumore oppure se si avverte uno strano odore dovuto al surriscaldamento), spegnere immediatamente il computer e portarlo dal rivenditore per un controllo.

Tutte le riparazioni devono essere eseguite dal rivenditore o per suo tramite. Si sconsiglia all'utente di aprire la console.

Le cose da non fare

Il VG8000, come tutte le altre apparecchiature elettriche ed elettroniche, è allergico all'eccessiva umidità. Prendere quindi tutte le precauzioni per evitare che nel computer entrino liquidi di qualsiasi genere.

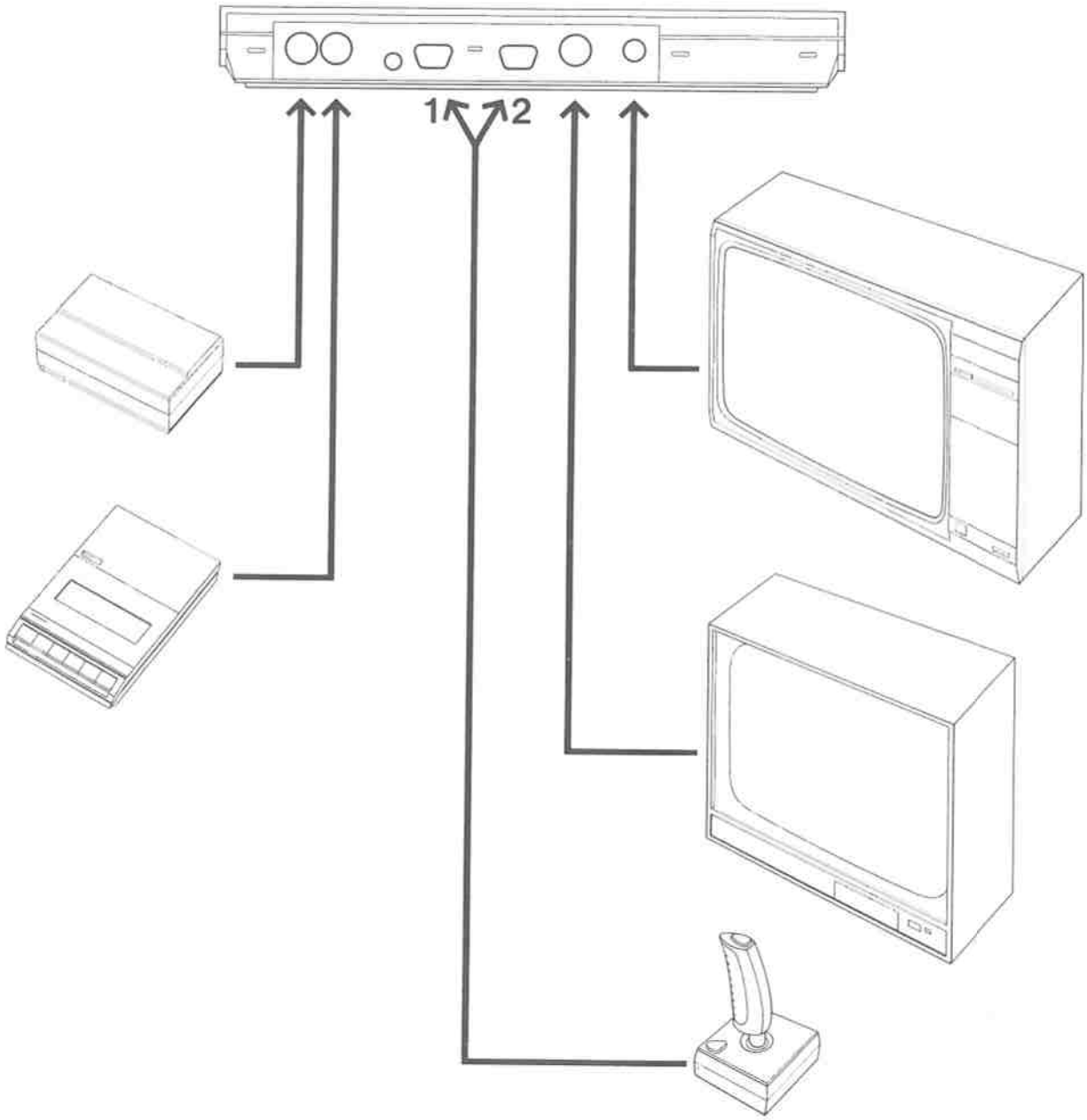
Le aperture di ventilazione sono state previste per assicurare il necessario raffreddamento del computer. Non tappare quindi queste aperture in modo da assicurare una corretta circolazione d'aria.

Tenere il VG8000 lontano dalle fonti di calore tipo stufe, radiatori e dalla luce diretta del sole.

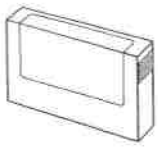
Non toccare con le dita i punti di contatto dei connettori per non creare le premesse di corrosione delle zone interessate.

3.2 Periferiche

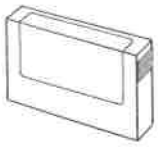
Nella parte posteriore del VG8000 sono previsti dei collegamenti standardizzati per le seguenti periferiche ed accessori:



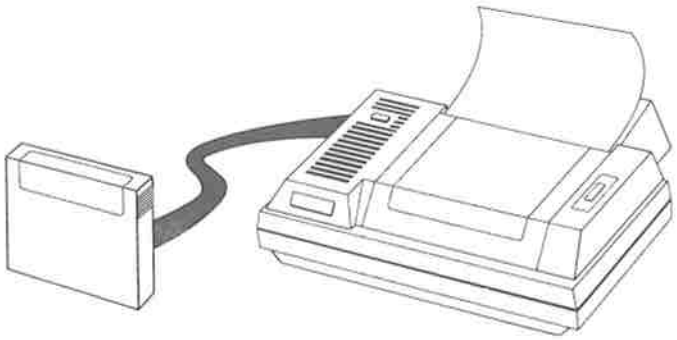
Utilizzando i connettori per cartucce, è possibile collegare le seguenti periferiche:



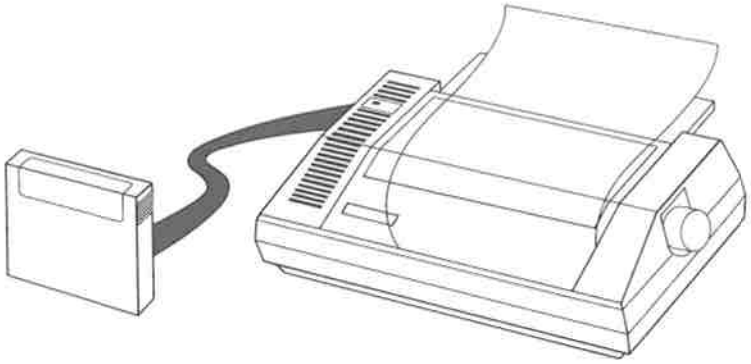
Cartuccia RAM da 16K



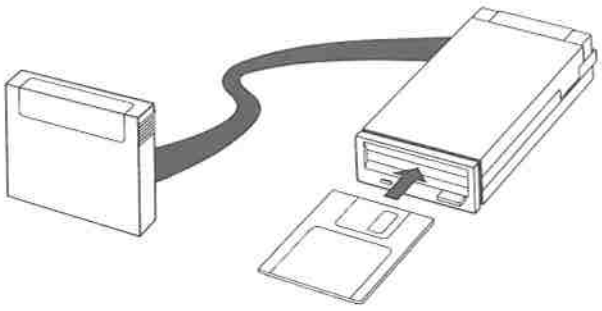
Cartuccia programma ROM



Stampante a 40 colonne



Stampante a 80 colonne



Unità a dischetti

Appendici

Appendice A

Elenco dei messaggi di errore

Messaggio di errore	Codice	Spiegazione
Bad file name (Nome file errato)	56	È stato usato un nome scorretto per indicare un file.
Bad file number (Numero file errato)	52	Il numero del file usato si riferisce ad un file che non è stato ancora aperto con l'istruzione "OPEN" o il numero del file è maggiore del numero più elevato introdotto con l'istruzione "MAXFILES".
Can't continue (Impossibile continuare)	17	L'operatore sta cercando di continuare un programma che: — è stato interrotto da un messaggio di errore — è stato modificato mentre la sua esecuzione era interrotta — non esiste.
Device I/O error (Errore di I/O periferica)	19	Si è verificato un errore di input o di output durante la lettura o la scrittura di un file o di un programma.
Direct statement (Istruzione diretta)	57	Durante il processo di caricamento di un file ASCII si è incontrata un'istruzione diretta.
Division by zero (Divisione per zero)	11	Svolgendo un calcolo, il computer deve eseguire una divisione per zero o si trova in una situazione in cui 0 viene elevato ad una potenza negativa.
Field overflow (Supero capacità campo)	50	Il numero totale di byte allocato nell'istruzione "FIELD" è maggiore di 256.
File already open (File già aperto)	54	Al computer viene chiesto di aprire un file con l'istruzione "OPEN", mentre quel file è già stato aperto. Oppure viene impartito un comando "KILL" per un file aperto.
File not open (File non aperto)	59	Con la sua istruzione di lettura e di scrittura l'operatore sta facendo riferimento ad un file che non è stato ancora aperto con l'istruzione "OPEN".

Illegal direct (Diretta illecita)	12	Viene usata un'istruzione che non è lecita nel modo diretto.
Illegal function call (Richiamo di funzione illecita)	5	Ad una funzione viene attribuito un parametro sbagliato. Ciò può verificarsi come risultato di: — un indice negativo o irragionevolmente grande — un parametro negativo o zero con la funzione "LOG" — un parametro negativo con la funzione "SQR" — un parametro scorretto con "MID$", "LEFT$", "RIGHT$", "INP", "OUT", "PEEK", "POKE", "TAB", "SPC", "STRING$", "SPACE$", "INSTR$", "ON...GOTO", "ON...GOSUB".
Input past end (Input dopo la fine)	55	L'operatore sta cercando di leggere dati da un file che è già stato letto completamente. Usare la funzione "EOF" per impedire questo errore.
Internal error (Errore interno)	51	È stata rilevata una situazione non prevista dall'interprete BASIC.
Line buffer overflow (Supero capacità buffer di riga)	25	L'operatore ha impostato una riga contenente troppi caratteri.
Missed operand (Operando mancante)	24	Un'espressione contiene una funzione, un'istruzione o un comando senza parametro.
NEXT without FOR (NEXT senza FOR)	1	In un'istruzione "NEXT", una variabile non corrisponde alla variabile della precedente istruzione "FOR" o un'istruzione "NEXT" non è preceduta da un'istruzione "FOR".
No RESUME (Manca RESUME)	21	Un errore non è terminato da un'istruzione "RESUME".
Out of DATA (Mancanza di dati)	4	Viene eseguita un'istruzione "READ" ma il programma non contiene altre istruzioni DATA con dati non letti.
Out of memory (Mancanza di memoria)	7	Un programma è troppo grande o contiene troppe iterazioni "FOR", "GOSUB" o troppe variabili. Può anche contenere espressioni che sono troppo complicate.

Out of string space (Mancanza di spazio per stringa)	14	Le variabili alfanumeriche superano lo spazio disponibile. Usare l'istruzione "CLEAR" per creare ulteriore spazio.
Overflow (Supero di capacità)	6	Il risultato di un calcolo è troppo grande per poter essere rappresentato nel formato dei numeri.
Redimensioned array (Tabella ridimensionata)	10	Sono state impartite due istruzioni "DIM" alla stessa variabile oppure una variabile già in uso è dimensionata con un'istruzione "DIM".
RESUME without error (RESUME senza errore)	22	Al computer viene chiesto di eseguire un'istruzione "RESUME" mentre non è stata ancora eseguita alcuna istruzione "ON ERROR GOTO".
RETURN without GOSUB (RETURN senza GOSUB)	3	Si incontra un'istruzione "RETURN" cui non corrisponde una precedente istruzione "GOSUB".
Sequential I/O only (Solo I/O sequenziale)	58	L'operatore sta cercando di leggere/scrivere casualmente in/da un file sequenziale.
String formula too complex (Formula della stringa troppo complessa)	16	L'espressione alfanumerica è troppo lunga o troppo complessa.
String too long (Stringa troppo lunga)	15	È stato compiuto un tentativo di inserire più di 255 caratteri alfanumerici in una stringa.
Subscript out of range (Indice fuori campo)	9	È stato compiuto un tentativo di fare riferimento ad una tabella con un indice sbagliato.
Syntax error (Errore di sintassi)	2	Un'istruzione, un comando o una funzione contengono caratteri scorrettamente posizionati, simboli di punteggiatura errati, parentesi spaiate, ecc.
Type mismatch (Tipo errato)	13	Ad una variabile alfanumerica è stato attribuito un valore numerico e viceversa oppure ad una funzione che dovrebbe avere un valore numerico è stato attribuito un parametro alfanumerico.
Unidentified line number (Numero riga non identificato)	8	Viene impartito un comando o un'istruzione che fa riferimento ad una riga di programma inesistente.
Unidentified user function (Funzione utente non identificata)	18	Viene richiamata una funzione definita dall'utente prima di aver definito la funzione stessa con l'istruzione "DEF".
Unprintable error (Errore non stampabile)	23 26-49 60-255	Questi codici non sono stati ancora definiti. L'utente può usarli per inserire proprie definizioni di errore.

`Verify error` (Ricercare errore)	20	Questo messaggio di errore viene emesso quando risulta che il programma in memoria differisce da quello sulla cassetta dopo che è stata eseguita l'istruzione "`CLOAD?`".

Appendice B

Funzioni matematiche

Le funzioni matematiche derivate che non sono state incluse direttamente in MSX-BASIC possono essere definite come segue:

Funzioni derivate	Definizione MSX-BASIC
SECANTE	= 1/COS(X)
COSECANTE	= 1/SIN(X)
COTANGENTE	= 1/TAN(X)
SENO INVERSO	= ATN(X/SQR(-X*X+1))
COSENO INVERSO	= -ATN(X/SQR(-X*X+1))+1.5708
SECANTE INVERSA	= ATN(X/SQR(X*X-1))+SGN(SGN(X)-1) *1.5708
COSECANTE INVERSA	= ATN(X/SQR(X*X-1))+(SGN(X)-1) *1.5708
COTANGENTE INVERSA	= ATN(X)+1.5708
SENO IPERBOLICO	= (EXP(X)-EXP(-X))/2
COSENO IPERBOLICO	= (EXP(X)+EXP(-X))/2
TANGENTE IPERBOLICA	= EXP(-X)/EXP(X)+EXP(-X)*2+1
SECANTE IPERBOLICA	= 2/(EXP(X)+EXP(-X))
COSECANTE IPERBOLICA	= 2/(EXP(X)-EXP(-X))
COTANGENTE IPERBOLICA	= EXP(-X)/(EXP(X)-EXP(-X))*2+1
SENO IPERBOLICO INVERSO	= LOG(X+SQR(X*X+1))
COSENO IPERBOLICO INVERSO	= LOG(X+SQR(X*X-1))
TANGENTE IPERBOLICA INVERSA	= LOG((1+X)/(1-X))/2
SECANTE IPERBOLICA INVERSA	= LOG((SQR(-X*X+1)+1)/X)
COSECANTE IPERBOLICA INVERSA	= LOG((SGN(X)*SQR(X*X+1)+1)/X)
COTANGENTE IPERBOLICA INVERSA	= LOG((X+1)/(X-1))/2

Appendice C

Tabella dei colori

Numero del colore	Colore
0	Trasparente
1	Nero
2	Verde
3	Verde chiaro
4	Blu scuro
5	Azzurro
6	Rosso scuro
7	Blu-verde
8	Rosso
9	Rosso chiaro
10	Giallo scuro
11	Giallo chiaro
12	Verde scuro
13	Magenta
14	Grigio
15	Bianco

Appendice D

Funzioni di controllo

Nella tabella che segue si troverà un sommario di tutte le funzioni di controllo MSX-BASIC. Queste funzioni vengono eseguite premendo il tasto "CTRL" simultaneamente ad un altro tasto.
Ciascuna funzione di controllo ha un proprio codice. Questo codice può essere usato, ad esempio, nella funzione "CHR$".
Esempio:

```
PRINT CHR$(7)
```

per produrre un segnale acustico (beep).

Al termine di questa tabella si troverà una descrizione delle varie funzioni di controllo.

Codice	Tasto CTRL + TASTO:	Tasto speciale	Funzione
1	A		Caratteri alternativi
2	B		Cursore alla parola precedente
3	C		Comando stop AUTO
4	D		Nessuna
5	E		Troncamento riga
6	F		Cursore alla parola successiva
7	G		Segnale acustico
8	H	BS	Ritorno unitario
9	I	TAB	Tabulazione
10	J		Riga successiva
11	K	HOME	Cursore all'angolo superiore sinistro
12	L	CLR	Cancellazione schermo
13	M	RETURN	Ritorno a capo
14	N		Cursore a fine riga
15	O		Nessuna
16	P		Nessuna
17	Q		Nessuna
18	R	INS	Inserimento
19	S		Nessuna
20	T		Nessuna
21	U		Cancellazione riga
22	V		Nessuna
23	W		Nessuna
24	X	SELECT	Nessuna
25	Y		Nessuna
26	Z		Nessuna
27	[	ESC	Nessuna
28	\	▶	Cursore a destra
29	]	◀	Cursore a sinistra
30	^	▲	Cursore verso l'alto
31	—	▼	Cursore verso il basso

Descrizione delle funzioni di controllo

Caratteri alternativi	Con questa funzione il successivo carattere battuto sarà un carattere alternativo (vedere Appendice E).
Cursore alla parola precedente	Il cursore si sposterà al primo carattere della precedente parola. Per MSX-BASIC l'ordine dei caratteri è da A a Z, da a a z o da 0 a 9.
Comando Stop AUTO	Questa funzione di comando porta MSX-BASIC a livello comandi se era stato attivato il comando "AUTO".
Troncamento riga	Saranno rimossi tutti i caratteri ed i simboli a partire dalla posizione del cursore fino alla fine della riga.
Cursore alla parola successiva	Il cursore si muove al primo carattere della successiva parola. Per MSX-BASIC, la sequenza dei caratteri è da A a Z, da a a z o da 0 a 9.
Segnale acustico	Questa funzione produce un segnale acustico.
Ritorno unitario	Questa funzione di controllo rimuove il carattere immediatamente alla sinistra del cursore. Il cursore stesso e tutti i caratteri che lo seguono si sposteranno di una posizione verso sinistra.
Tabulazione	Il cursore viene spostato alla successiva posizione di tabulazione. C'è una tabulazione ogni otto posizioni.
Riga successiva	Il cursore viene spostato alla prima posizione della riga successiva.
Cursore all'angolo superiore sinistro	Il cursore viene spostato all'angolo superiore sinistro dello schermo senza per altro cancellare lo schermo.
Cancellazione schermo	Il cursore viene spostato all'angolo superiore sinistro e lo schermo viene cancellato.
Ritorno a capo	La riga logica (o istruzione) viene convalidata.
Cursore a fine riga	Il cursore viene spostato alla prima posizione che segue l'ultimo carattere della riga.

Inserimento	Serve ad inserire caratteri addizionali tra quelli esistenti. Se questa funzione di controllo viene ripetuta, i nuovi caratteri sostituiranno quelli esistenti.
Cancellazione riga	L'intera riga viene rimossa.
Cursore a destra	Il cursore viene spostato di una posizione verso destra.
Cursore a sinistra	Il cursore viene spostato di una posizione verso sinistra.
Cursore verso l'alto	Il cursore viene spostato di una riga verso l'alto.
Cursore verso il basso	Il cursore viene spostato di una riga verso il basso.

Appendice E

Codici di caratteri

In questa appendice si troveranno tutti i caratteri che possono essere usati con MSX-BASIC unitamente ai corrispondenti codici. Questi codici possono essere usati ad esempio nella funzione "CHR$".
I seguenti due esempi producono esattamente lo stesso effetto.

```
PRINT "A"
```

e

```
PRINT CHR$(65)
```

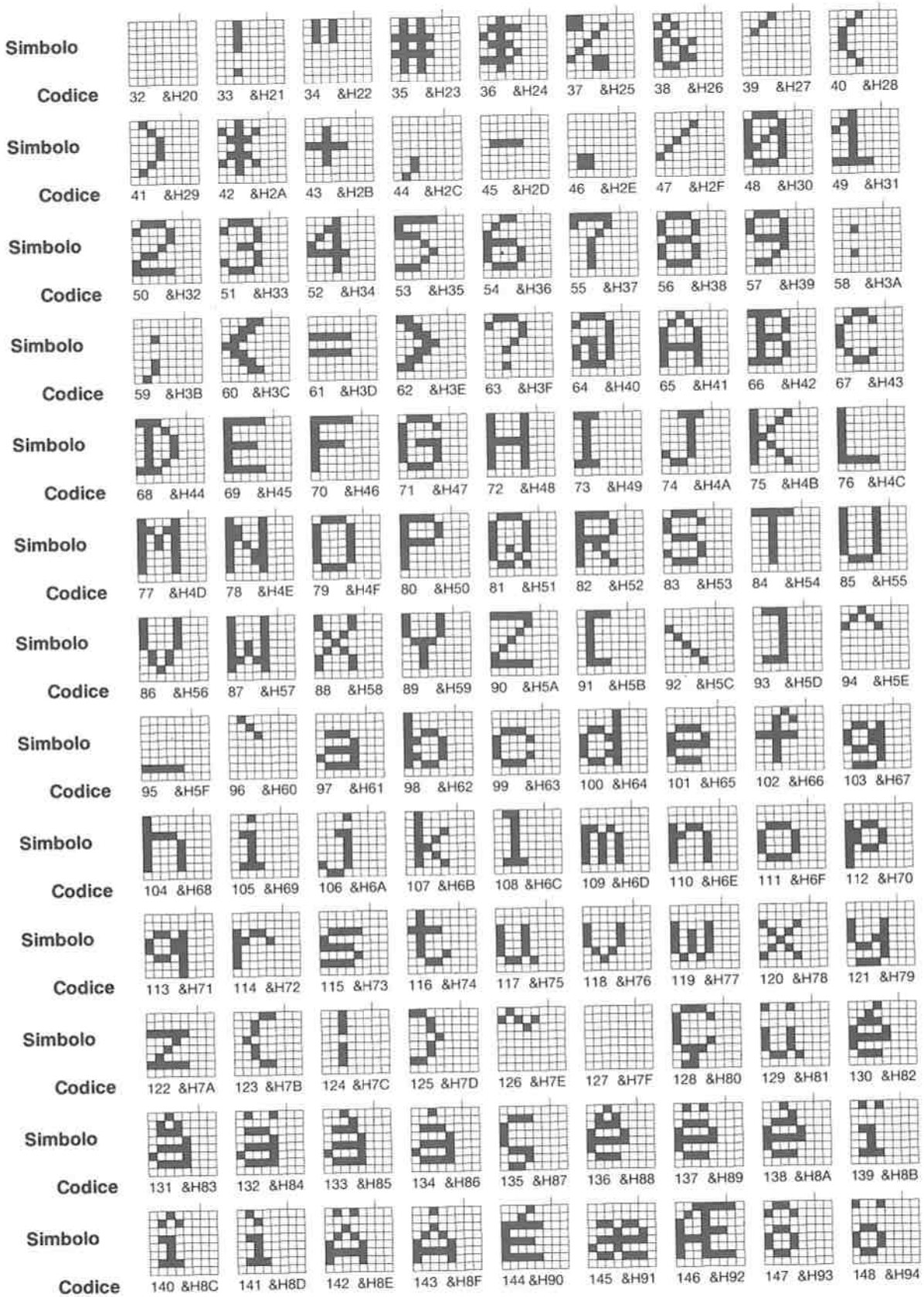
Nel modo testo 1 (40 caratteri per riga), sullo schermo non compaiono le due colonne di destra dei punti di immagine dei caratteri. Provare l'esempio seguente:

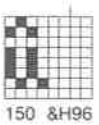
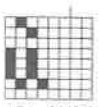
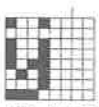
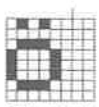
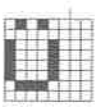
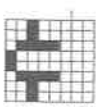
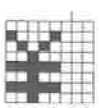
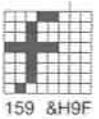
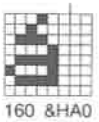
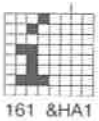
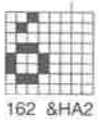
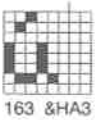
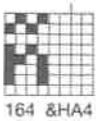
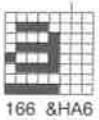
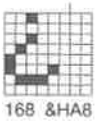
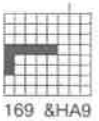
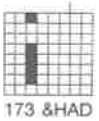
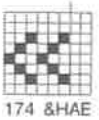
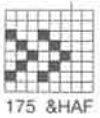
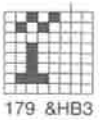
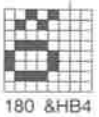
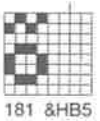
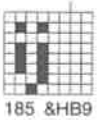
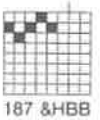
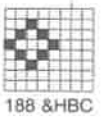
```
NEW
10 SCREEN 0
20 PRINT CHR$(210)
RUN
```

Modificare ora la riga 10 come segue:

```
10 SCREEN 1
```

ed eseguire di nuovo il programma per riscontrare la differenza.


Simbolo
Codice
32 &H20 33 &H21 34 &H22 35 &H23 36 &H24 37 &H25 38 &H26 39 &H27 40 &H28
Simbolo
Codice
41 &H29 42 &H2A 43 &H2B 44 &H2C 45 &H2D 46 &H2E 47 &H2F 48 &H30 49 &H31
Simbolo
Codice
50 &H32 51 &H33 52 &H34 53 &H35 54 &H36 55 &H37 56 &H38 57 &H39 58 &H3A
Simbolo
Codice
59 &H3B 60 &H3C 61 &H3D 62 &H3E 63 &H3F 64 &H40 65 &H41 66 &H42 67 &H43
Simbolo
Codice
68 &H44 69 &H45 70 &H46 71 &H47 72 &H48 73 &H49 74 &H4A 75 &H4B 76 &H4C
Simbolo
Codice
77 &H4D 78 &H4E 79 &H4F 80 &H50 81 &H51 82 &H52 83 &H53 84 &H54 85 &H55
Simbolo
Codice
86 &H56 87 &H57 88 &H58 89 &H59 90 &H5A 91 &H5B 92 &H5C 93 &H5D 94 &H5E
Simbolo
Codice
95 &H5F 96 &H60 97 &H61 98 &H62 99 &H63 100 &H64 101 &H65 102 &H66 103 &H67
Simbolo
Codice
104 &H68 105 &H69 106 &H6A 107 &H6B 108 &H6C 109 &H6D 110 &H6E 111 &H6F 112 &H70
Simbolo
Codice
113 &H71 114 &H72 115 &H73 116 &H74 117 &H75 118 &H76 119 &H77 120 &H78 121 &H79
Simbolo
Codice
122 &H7A 123 &H7B 124 &H7C 125 &H7D 126 &H7E 127 &H7F 128 &H80 129 &H81 130 &H82
Simbolo
Codice
131 &H83 132 &H84 133 &H85 134 &H86 135 &H87 136 &H88 137 &H89 138 &H8A 139 &H8B
Simbolo
Codice
140 &H8C 141 &H8D 142 &H8E 143 &H8F 144 &H90 145 &H91 146 &H92 147 &H93 148 &H94

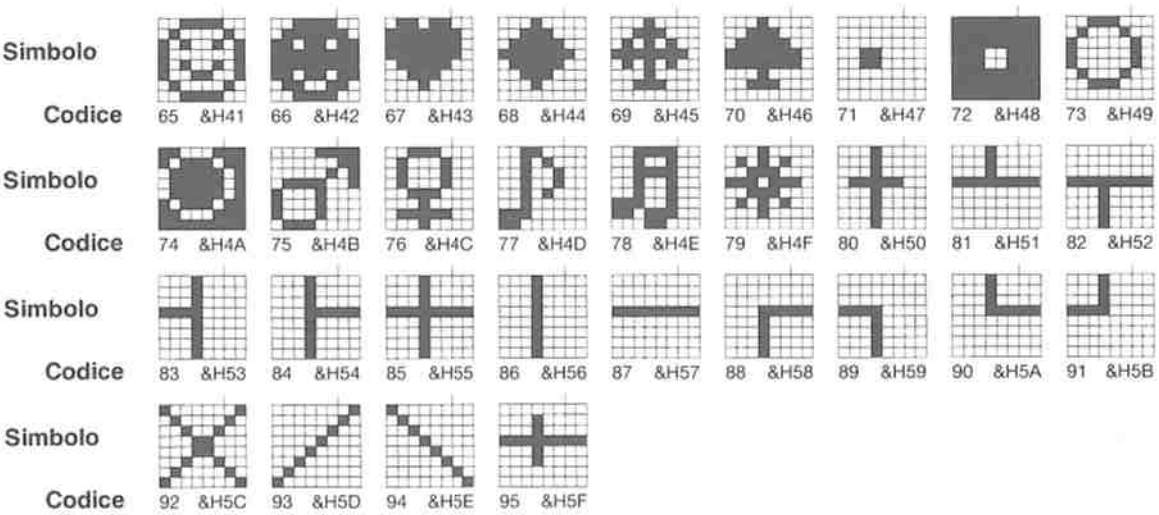
Simbolo									
Codice	149 &H95	150 &H96	151 &H97	152 &H98	153 &H99	154 &H9A	155 &H9B	156 &H9C	157 &H9D
Simbolo									
Codice	158 &H9E	159 &H9F	160 &HA0	161 &HA1	162 &HA2	163 &HA3	164 &HA4	165 &HA5	166 &HA6
Simbolo									
Codice	167 &HA7	168 &HA8	169 &HA9	170 &HAA	171 &HAB	172 &HAC	173 &HAD	174 &HAE	175 &HAF
Simbolo									
Codice	176 &HB0	177 &HB1	178 &HB2	179 &HB3	180 &HB4	181 &HB5	182 &HB6	183 &HB7	184 &HB8
Simbolo									
Codice	185 &HB9	186 &HBA	187 &HBB	188 &HBC	189 &HBD	190 &HBE	191 &HBF	192 &HC0	193 &HC1
Simbolo									
Codice	194 &HC2	195 &HC3	196 &HC4	197 &HC5	198 &HC6	199 &HC7	200 &HC8	201 &HC9	202 &HCA
Simbolo									
Codice	203 &HCB	204 &HCC	205 &HCD	206 &HCE	207 &HCF	208 &HD0	209 &HD1	210 &HD2	211 &HD3
Simbolo									
Codice	212 &HD4	213 &HD5	214 &HD6	215 &HD7	216 &HD8	217 &HD9	218 &HDA	219 &HDB	220 &HDC
Simbolo									
Codice	221 &HDD	222 &HDE	223 &HDF	224 &HE0	225 &HE1	226 &HE2	227 &HE3	228 &HE4	229 &HE5
Simbolo									
Codice	230 &HE6	231 &HE7	232 &HE8	233 &HE9	234 &HEA	235 &HEB	236 &HEC	237 &HED	238 &HEE
Simbolo									
Codice	239 &HEF	240 &HF0	241 &HF1	242 &HF2	243 &HF3	244 &HF4	245 &HF5	246 &HF6	247 &HF7
Simbolo									
Codice	248 &HF8	249 &HF9	250 &HFA	251 &HFB	252 &HFC	253 &HFD	254 &HFE	255 &HFF	

Caratteri alternativi

Oltre ai suddetti simboli ce ne sono numerosi altri, indicati dai codici da 65 a 95 se preceduti da "CHR$(1)".
Esempio:

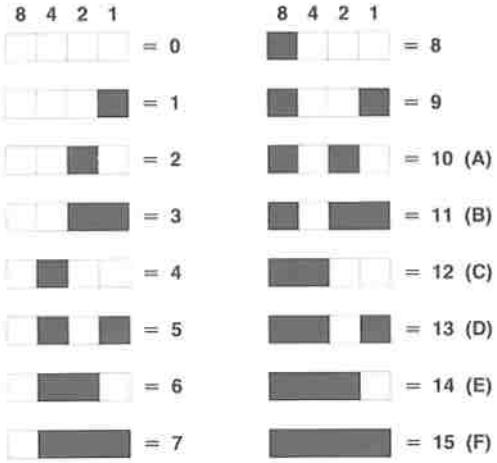
```
NEW
10 SCREEN 1
20 PRINT CHR$(65)
30 PRINT CHR(1)+CHR$(65)
RUN
```

La riga 20 produrrà sullo schermo il carattere normale "65" (="A"), mentre la riga 30 produrrà il carattere alternativo "65" (=un viso).



Matrici di caratteri

Ciascuna riga di una matrice di caratteri è divisa in 2 × 4 quadrati. I quattro quadrati hanno i valori di 8, 4, 2 e 1 rispettivamente.
Quattro quadrati insieme possono avere un valore compreso tra 0 e 15:



I valori compresi tra 0 e 15 sono rappresentati dalle lettere da A a F. Il valore minimo di una riga di una matrice di caratteri è 00 ed il massimo valore è FF.

Appendice F

Parole riservate

Le seguenti parole non possono essere usate come nome per una variabile:

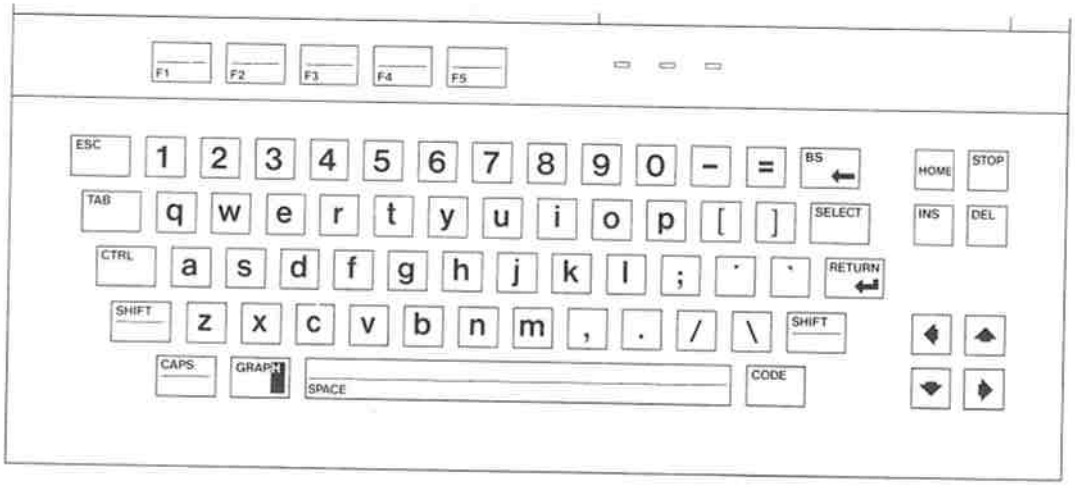
ABS
AND
AS
ASC
ATN
AUTO
BASE
BEEP
BIN$
BLOAD
BSAVE
CALL
CDBL
CHR$
CINT
CIRCLE
CLEAR
CLOAD
CLOSE
CLS
COLOR
CONT
COPY
COS
CSAVE
CSNG
CSRLIN
CVD
CVI
CVS
DATA
DEF
DEFINT
DEFDBL
DEFSNG
DEFSTR
DEFUSR
DELETE
DIM
DRAW
DSKF
ELSE
END
EOF
EQV
ERASE
ERL
ERR
ERROR
EXP
FIELD
FILES
FIX
FN
FOR
FRE
GET
GOSUB
GOTO
HEX$
IF
IMP
INKEY$
INP
INPUT
INPUT$
INSTR
INT
INTERVAL
KEY
KILL
LEFT$
LEN
LET
LINE
LIST
LLIST
LOAD
LOC
LOCATE
LOF
LOG
LPOS
LPRINT
LSET
MAXFILES
MERGE
MID$
MKD$
MKI$
MOD
MOTOR
MKS$
NAME
NEW
NEXT
NOT
OCT$
OFF
ON
OPEN
OR
OUT
PAD
PAINT
PDL
PEEK
PLAY
POINT
POKE
POS
PRINT
PSET
PRESET
PUT
READ
REM
RENUM
RESTORE
RESUME
RETURN
RIGHT$
RND
RSET
RUN
SAVE
SCREEN
SGN
SIN
SOUND
SPACE$
SPC
SPRITE
SPRITE$
SQR
STEP
STICK
STOP
STR$
STRIG
STRING$
SWAP
TAB
TAN
THEN
TIME
TROFF
TRON
USING
USR
VAL
VARPTR
VDP
VPEEK
VPOKE
WAIT
WIDTH
XOR

Appendice G

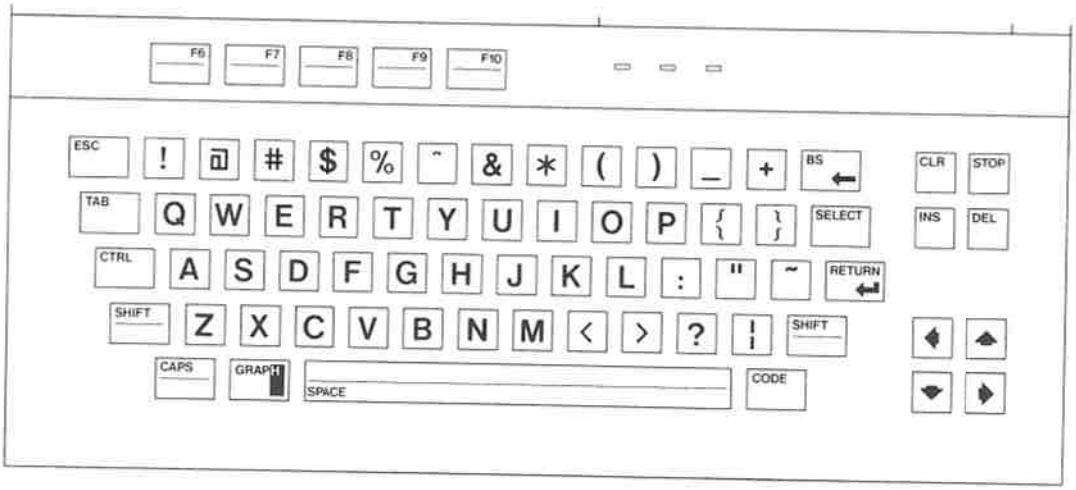
La tastiera

Questa appendice fornisce un sommario completo di tutti i caratteri che si possono produrre attraverso la tastiera.

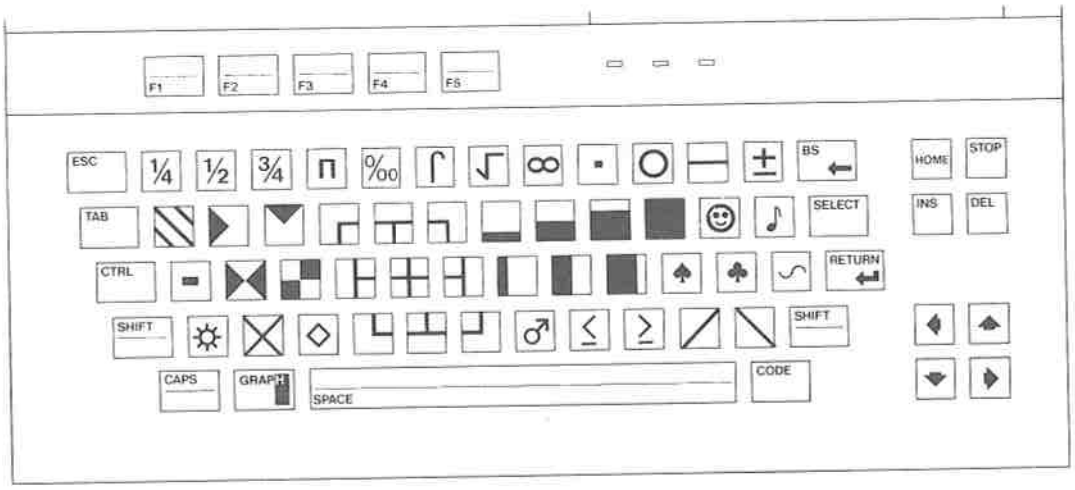
A. Caratteri standard



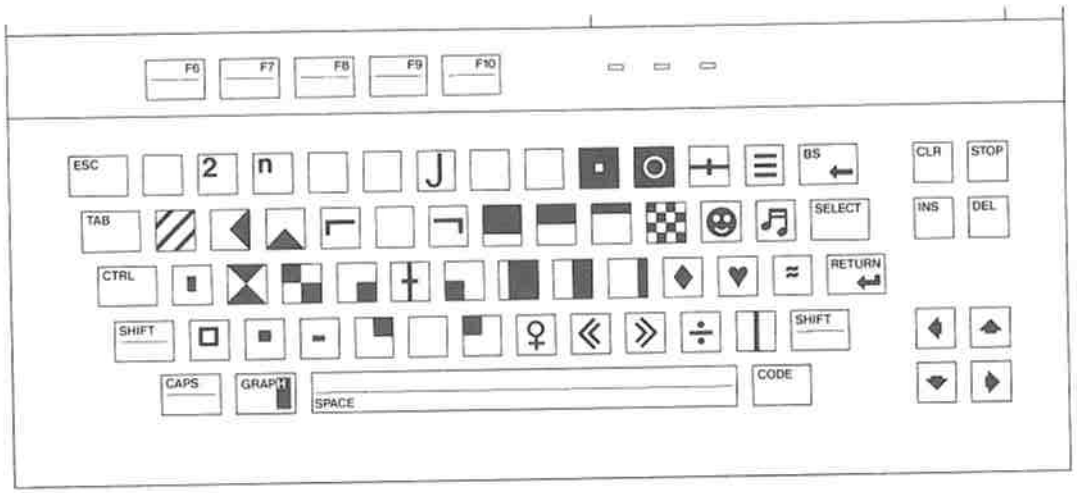
B. Caratteri disponibili quando viene premuto il tasto del carattere simultaneamente al tasto SHIFT



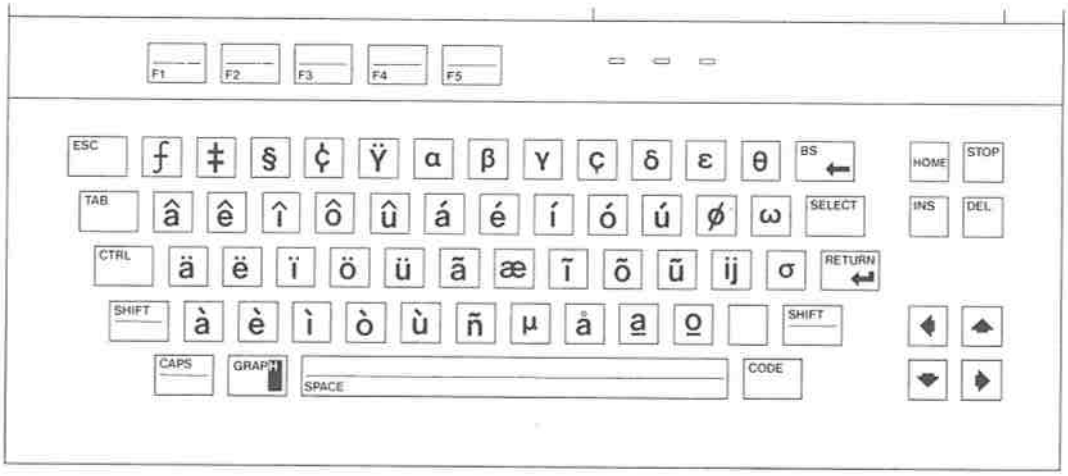
C. Caratteri disponibili dopo che è stato premuto il tasto GRAPH



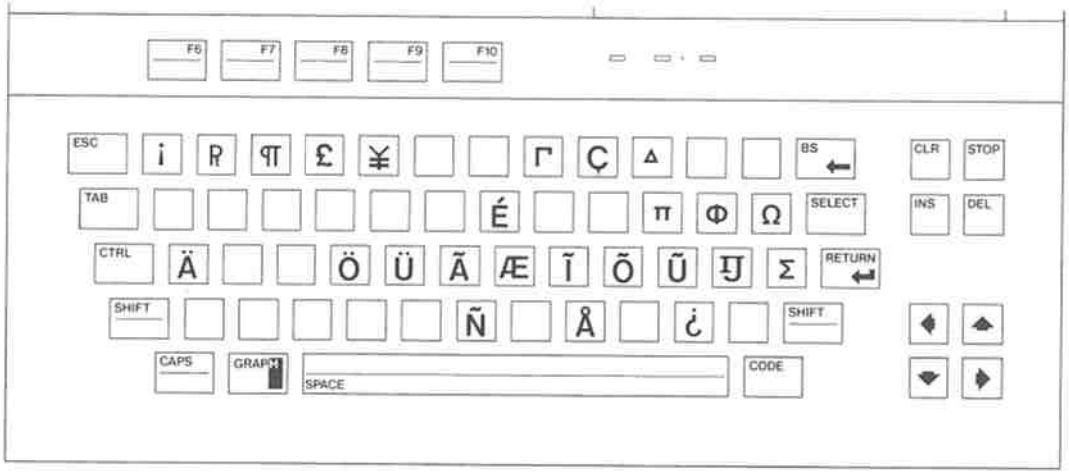
D. Caratteri disponibili dopo che è stato premuto il tasto GRAPH e quando viene premuto il tasto del carattere simultaneamente al tasto SHIFT



E. Caratteri disponibili dopo che è stato premuto il tasto CODE



F. Caratteri disponibili dopo che è stato premuto il tasto CODE e viene premuto il tasto del carattere simultaneamente al tasto SHIFT



Appendice H

Specifiche tecniche

1. Serie di chip

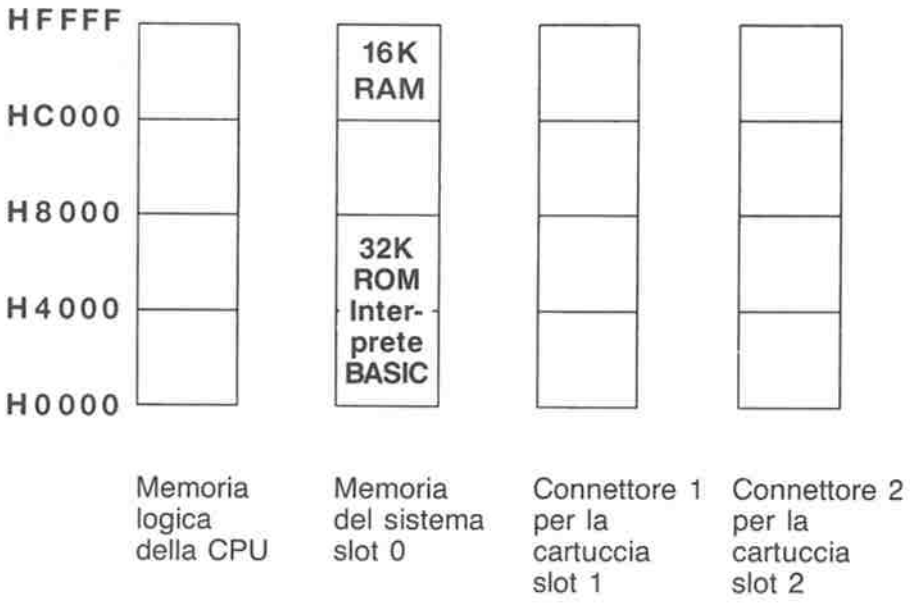
   CPU Unità centrale di elaborazione Z80A, 3,5 MHz

   VDP Processore Video
   TI TMS-9929A o chip analogo

   PSG Generatore di suoni programmabile
   GI AY-3-8910 o chip analogo

   PPI Interfaccia programmabile per periferiche I 8255

2. Mappa della memoria



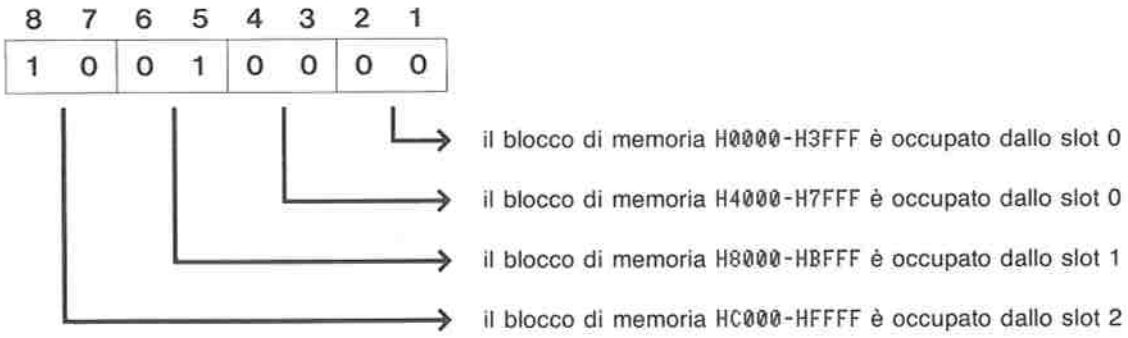
Il computer MSX ha inoltre una RAM video da 16K (vedere Appendice I di questo manuale).

Con le memoria RAM da 16K, gli indirizzi di memoria da `&HC000` a `&HC31F` e gli indirizzi di memoria da `&HF380` a `&HFFFF` sono usati dal computer come area di lavoro.

Con le memorie RAM da 32K, gli indirizzi di memoria da `&H8000` fino a `&H831F` e gli indirizzi di memoria da `&HF380` fino a `&HFFFF` sono usati dal computer come area di lavoro.

Lo spazio di memoria logica della cpu è occupato da spazi fisici di memoria delle varie slot di memoria. Il modo in cui queste slot di memoria sono occupate è indicato nel registro di selezione delle slot, che si trova nell'uscita A del PPI.

Ecco un esempio:



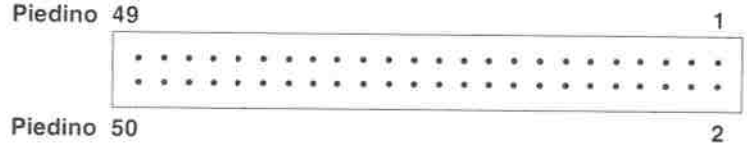
L'indirizzo dell'uscita A si può trovare nella tabella degli indirizzi di input/output.

3. Tabella degli indirizzi di input/output

INDIRIZZO	R/W	DESCRIZIONE	NOTA
HA8	W	Scrittura dati Porta A	8255
	R	Lettura dati Porta A	PPI
HA9	W	Scrittura dati Porta B	
	R	Lettura dati Porta B	
HAA	W	Scrittura dati Porta C	
	R	Lettura dati Porta C	
HAB	W	Impostazione di modo	
HA0	W	Bloccaggio indirizzo	AY-3-8910
HA1	W	Scrittura dati	PSG
HA2	R	Lettura dati	
H98	W	Scrittura dati V-RAM	TMS 9929A
	R	Lettura dati V-RAM	VDP
H99	W	Serie di indirizzi di comando	
	R	Lettura di stato	

Con MSX-BASIC, esiste la possibilità di comunicare direttamente con le uscite di input/output usando le istruzioni "OUT" e "WAIT" e la funzione "INP".

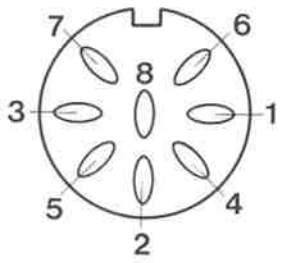
4. Connettori per cartuccia



Piedino	Nome	I/O	Piedino	Nome	I/O	Piedino	Nome	I/O
1	CS1	O	2	CS2	O	3	CS12	O
4	SLTSL	O	5	Riservato		6	RFSH	O
7	WAIT	I	8	INT	I	9	M1	O
10	BusDIR	I	11	IORQ	O	12	MERQ	O
13	WR	O	14	RD	O	15	RESET	O
16	Riservato		17	A9	O	18	A15	O
19	A11	O	20	A10	O	21	A7	O
22	A6	O	23	A12	O	24	A8	O
25	A14	O	26	A13	O	27	A14	O
28	A0	O	29	A3	O	30	A2	O
31	A5	O	32	A4	O	33	D1	I/O
34	D0	I/O	35	D3	I/O	36	D2	I/O
37	D5	I/O	38	D4	I/O	39	D7	I/O
40	D6	I/O	41	GND		42	CLOCK	O
43	GND		44	SW1		45	+ 5V	
46	SW2		47	+ 5V		48	+ 12V	
49	SOUNDIN	I	50	—12V				

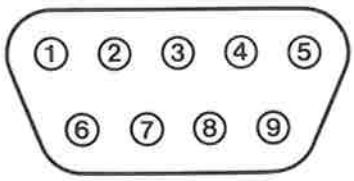
5. Connettore per registratore dati

Piedino	Nome	I/O
1	GND	
2	GND	
3	GND	
4	CMTOUT	O
5	CMTIN	I
6	REM+	O
7	REM—	O
8	GND	



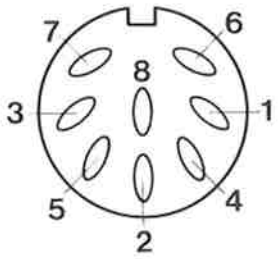
6. Connettore per joystick

Piedino	Nome	I/O
1	FWD	I
2	BACK	I
3	LEFT	I
4	RIGHT	I
5	+ 5V	
6	TRG 1	I/O
7	TRG 2	O
8	Output	O
9	GND	



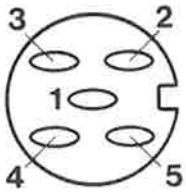
7. Connettore per monitor

Piedino	Nome
1	+ 5V
2	GND
3	AUDIO
4	Luminanza
5	VIDEO
6	+ 12V
7	
8	



8. Connettore per alimentatore

Piedino	Nome
1	
2	0 V
3	+ 12V
4	5 V
5	— 12V



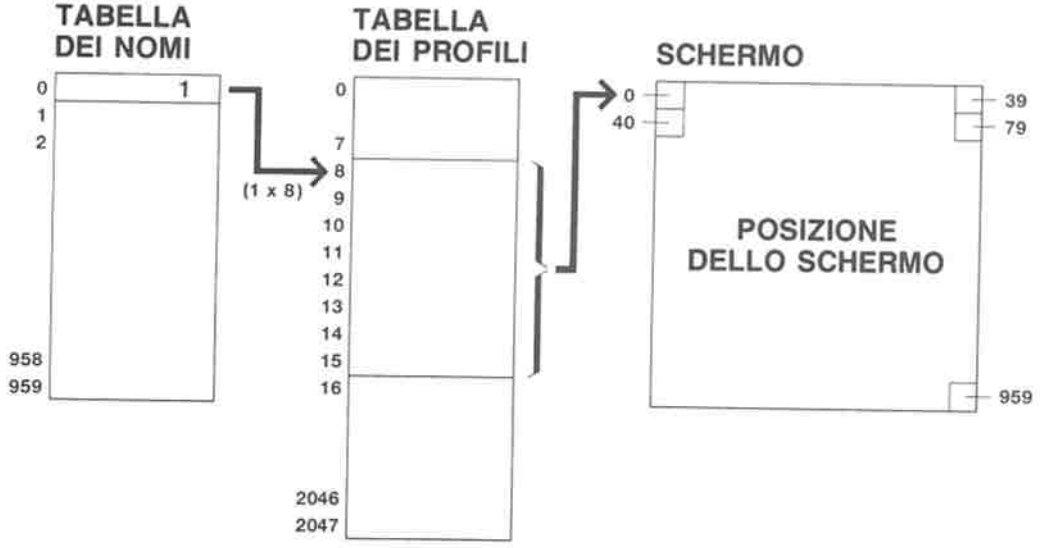
Appendice I

Il processore video (VDP)

Il computer MSX è munito di un VDP che fornisce tutte le informazioni necessarie per produrre un'immagine sullo schermo. Perché ciò sia possibile, il VDP deve avere una memoria RAM del video da 16K.
Le informazioni della RAM del video possono essere richiamate con la funzione "VPEEK" e possono essere memorizzate immediatamente con l'istruzione "VPOKE".
Tutte le istruzioni di stampa, ad esempio "PRINT" e tutte le istruzioni grafiche, ad esempio "DRAW" vengono automaticamente elaborate da MSX-BASIC ed inserite nella memoria RAM del video. Tutto ciò che compare sullo schermo è registrato nella memoria RAM del video in tabelle che vengono usate in relazione al modo dello schermo.

Modo testo 1

In questo modo lo schermo è diviso in 24 righe con 40 caratteri ciascuna. A questo scopo ci sono due tabelle nella memoria RAM del video: una tabella dei nomi ed una tabella dei profili. La tabella dei nomi contiene tutti i codici dei caratteri e tutti i simboli che compaiono sullo schermo. Nella tabella dei profili sono stati riservati 8 byte per nome per definire come si presenta il carattere. La posizione nella tabella dei nomi indica anche la posizione sullo schermo.



La posizione in cui si trovano le tabelle nella memoria RAM del video è indicata dalla variabile "BASE(n)". "BASE(0)" dà il primo indirizzo della tabella dei nomi, mentre "BASE(2)" dà il primo indirizzo della tabella dei profili.
Esempio:

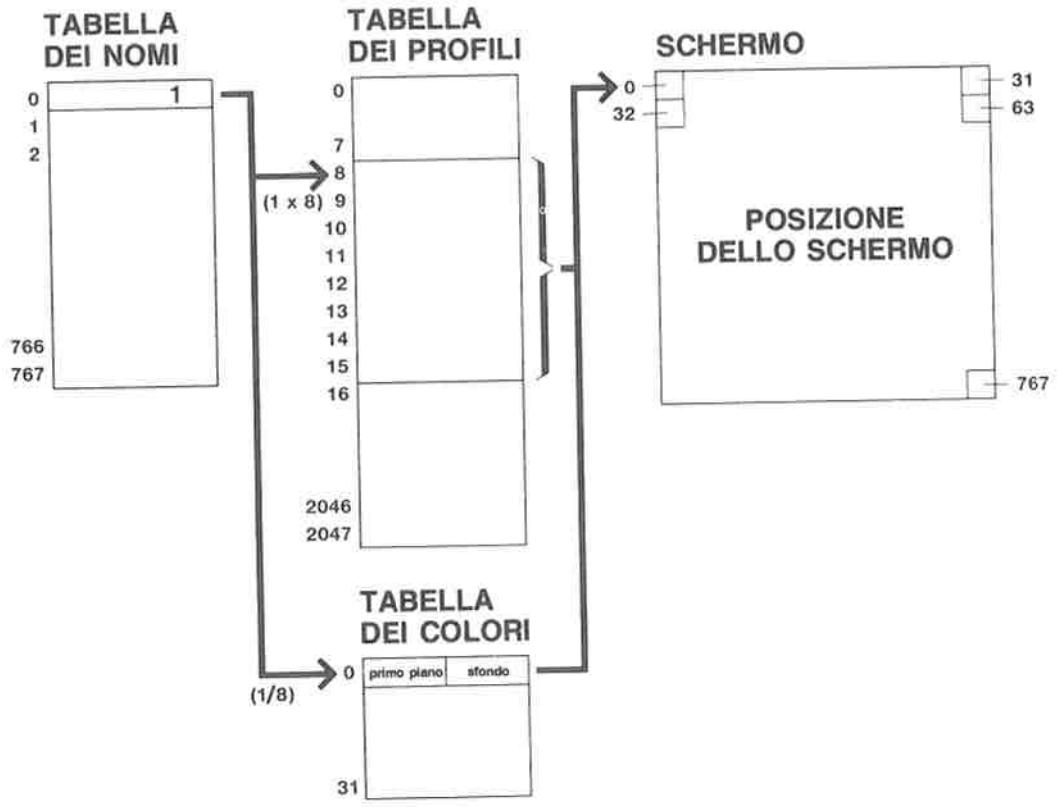
```
NEW
10 SCREEN 0:WIDTH 40
20 PRINT "HALLO"
30 PRINT BASE(0),BASE(2)
```

```
40 A=VPEEK(BASE(0)):PRINT A
50 B=A*8:FOR I=B TO B+7
60 C$="00000000"+BIN$(VPEEK(BASE(2)+I))
65 PRINT RIGHT$(C$,8)
70 NEXT I
RUN
```

Nel suddetto esempio, il primo indirizzo della tabella dei nomi e della tabella dei profili è scritto nella riga 30. Nella riga 40 viene scritto il codice del carattere (nome) della lettera "H". Successivamente viene riprodotto in forma binaria il profilo della lettera "H". Confrontare queste 8 righe con il codice del carattere della lettera "H" nell'Appendice E.

Modo testo 2

In questo modo lo schermo è diviso in 24 righe di 32 caratteri ciascuna. A questo scopo ci sono tre tabelle nella memoria RAM del video: una tabella dei nomi, una tabella dei profili ed una tabella dei colori. Le tabelle dei profili e dei nomi funzionano allo stesso modo di quelle dello schermo nel modo testo 1. La tabella dei colori contiene 32 diverse combinazioni dei colori di primo piano e di sfondo.
I codici dei caratteri da 0 a 7 nella tabella dei nomi usano la prima combinazione della tabella dei colori. Le seguenti otto entrate usano una seconda combinazione dalla tabella dei colori, ecc. Una combinazione della tabella dei colori è memorizzata in un byte suddiviso in due gruppi di 4 bit. I primi 4 bit indicano il colore del primo piano, i secondi quattro bit il colore dello sfondo. La posizione della tabella dei nomi nella memoria RAM del video è registrata nella variabile "BASE(5)", la posizione della tabella dei profili nella variabile "BASE(7)" e la posizione della tabella dei colori nella variabile "BASE(6)".



Si proverà ora a cambiare come segue il precedente esempio:

```
NEW
10 SCREEN 1:WIDTH 32
30 PRINT BASE(5),BASE(6),BASE(7)
40 A=VPEEK(BASE(5)):PRINT A
60 PRINT C$="00000000"+BIN$(VPEEK(BASE(7)+I))
80 D=VPEEK(BASE(6))
90 PRINT USING "##";D/16
100 PRINT USING "##";D MOD 16
RUN
```

Con questo esempio vengono scritti i primi indirizzi della tabella dei nomi, della tabella dei profili e della tabella dei colori seguiti dalla forma del carattere e dal nome di quel carattere. In questo caso la lettera "H".
Una volta eseguito ed osservato questo esempio, battere:

```
SCREEN 0:WIDTH 40
```

Modo grafico 1

In questo modo lo schermo è diviso in 3x8 righe con 32 posizioni su ciascuna riga. Questo modo dello schermo comprende anche una tabella dei nomi che a sua volta indica la posizione sullo schermo. Il numero nella tabella dei nomi indica dove è possibile trovare il profilo ed il colore del carattere che deve essere scritto. Per ciascun nome nella tabella dei nomi è possibile definire un profilo nella tabella dei profili. A questo scopo la tabella dei profili è lunga 6144 byte (768 nomi x 8 byte). Per ogni byte della tabella dei profili può essere definito un colore di sfondo e di primo piano nella tabella dei colori. Questo è il motivo per cui la tabella dei colori ha la stessa lunghezza della tabella dei profili. La posizione della tabella dei nomi nella memoria RAM del video è registrata nella variabile "BASE(10)" e la posizione della tabella di profili nella variabile "BASE(2)". La posizione della tabella dei colori è registrata nella variabile "BASE(11)".

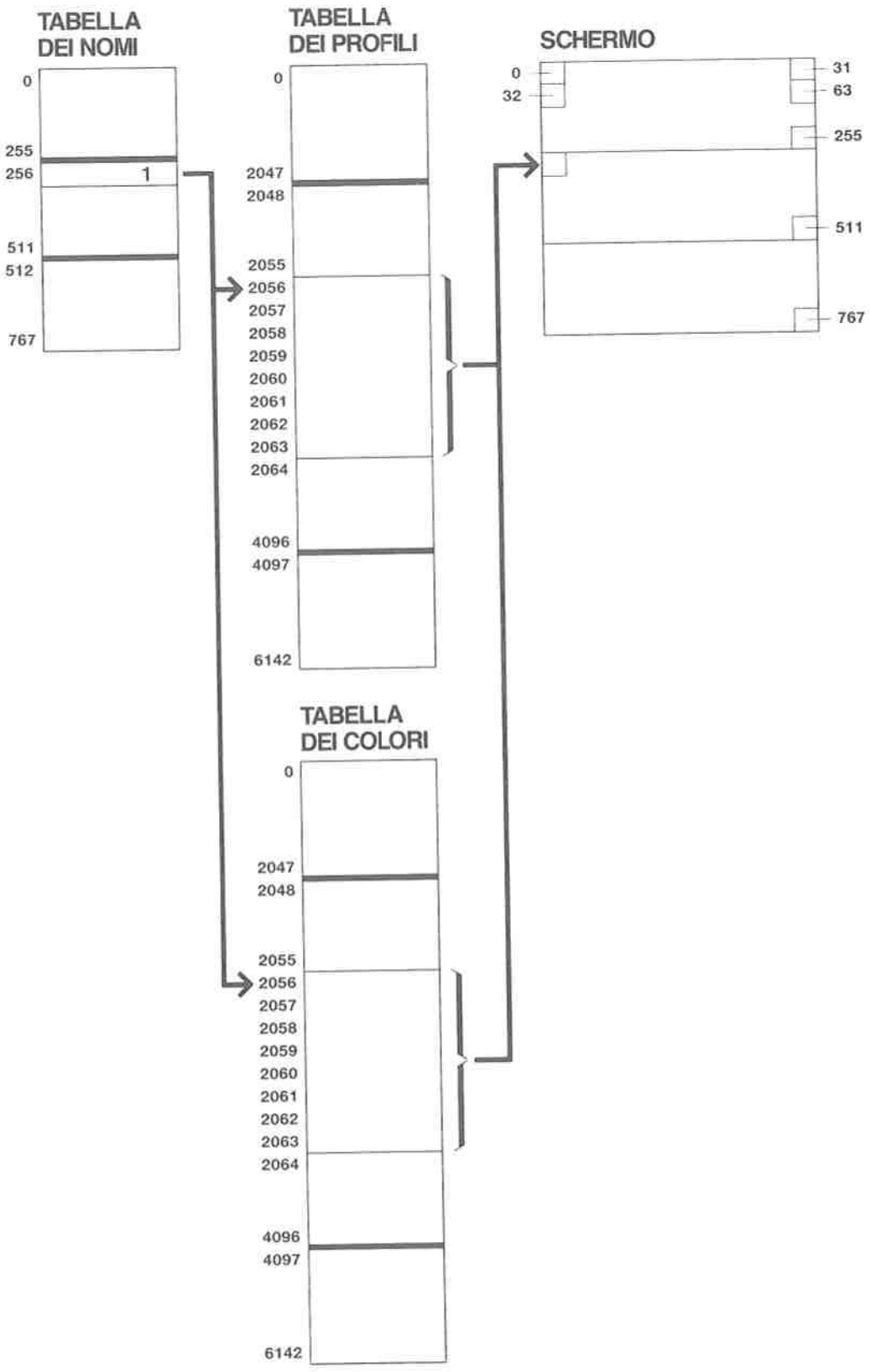
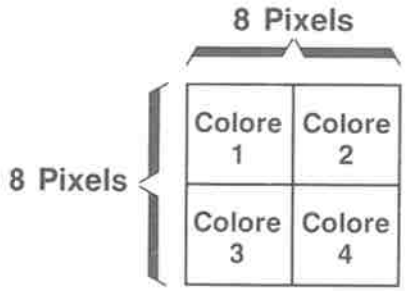


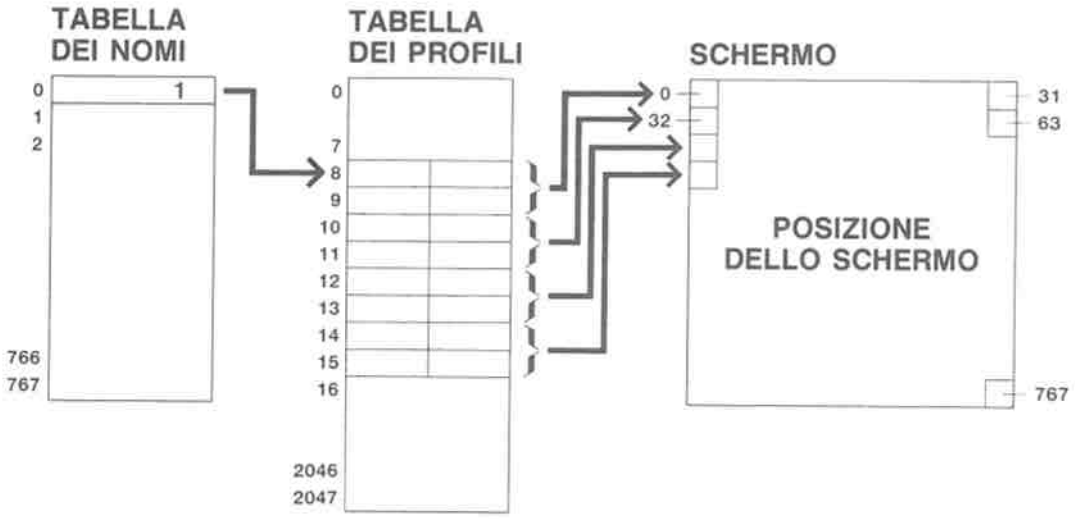
TABELLA
DEI NOMI
0
255
256
1
511
512
767
TABELLA
DEI PROFILI
0
2047
2048
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
4096
4097
6142
SCHERMO
0
32
31
63
255
511
767
TABELLA
DEI COLORI
0
2047
2048
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
4096
4097
6142

Modo grafico 2

In questo modo lo schermo è diviso in 24 righe con 32 caratteri ciascuna. Ogni posizione si compone di quattro quadrati con una dimensione di 4x4 punti di immagine ciascuno. Ciascun quadrato contiene un colore. In questo modo dello schermo si troverà una tabella dei nomi che indica anche la posizione sullo schermo. Il numero sulla tabella dei nomi indica dove si possono trovare nella tabella dei profili i colori delle quattro posizioni sottostanti.



Ciascun byte nella tabella dei profili è stato diviso in 2 gruppi da 4 bit ed ogni combinazione di 4 bit indica un numero di colore.
Ciò significa che ogni nome nella tabella dei nomi richiede 8 byte nella tabella dei profili.
La posizione della tabella dei nomi nella memoria RAM del video è contenuta nella variabile "BASE(15)". La posizione della tabella dei profili è contenuta nella variabile "BASE(17)".



Sprites

Anche tutti gli sprites sono inseriti nella memoria RAM del video. La definizione dello sprite è registrata nella tabella dei profili degli sprites mentre la posizione dello sprite sullo schermo è registrata nella tabella degli attributi degli sprites.

La tabella dei profili degli sprites si compone di 8 byte per gli sprites di formato piccolo e di 32 byte per quelli di formato grande.

Nella tabella degli attributi degli sprites è possibile indicare un massimo di 32 posizioni di sprites.
Ciascuna entrata di sprite nella tabella degli attributi degli sprites si compone di 4 byte:

	7 6 5 4 3 2 1 0 Bit
Byte 0	Posizione Y
1	Posizione X
2	Nome sprite
3	Colore

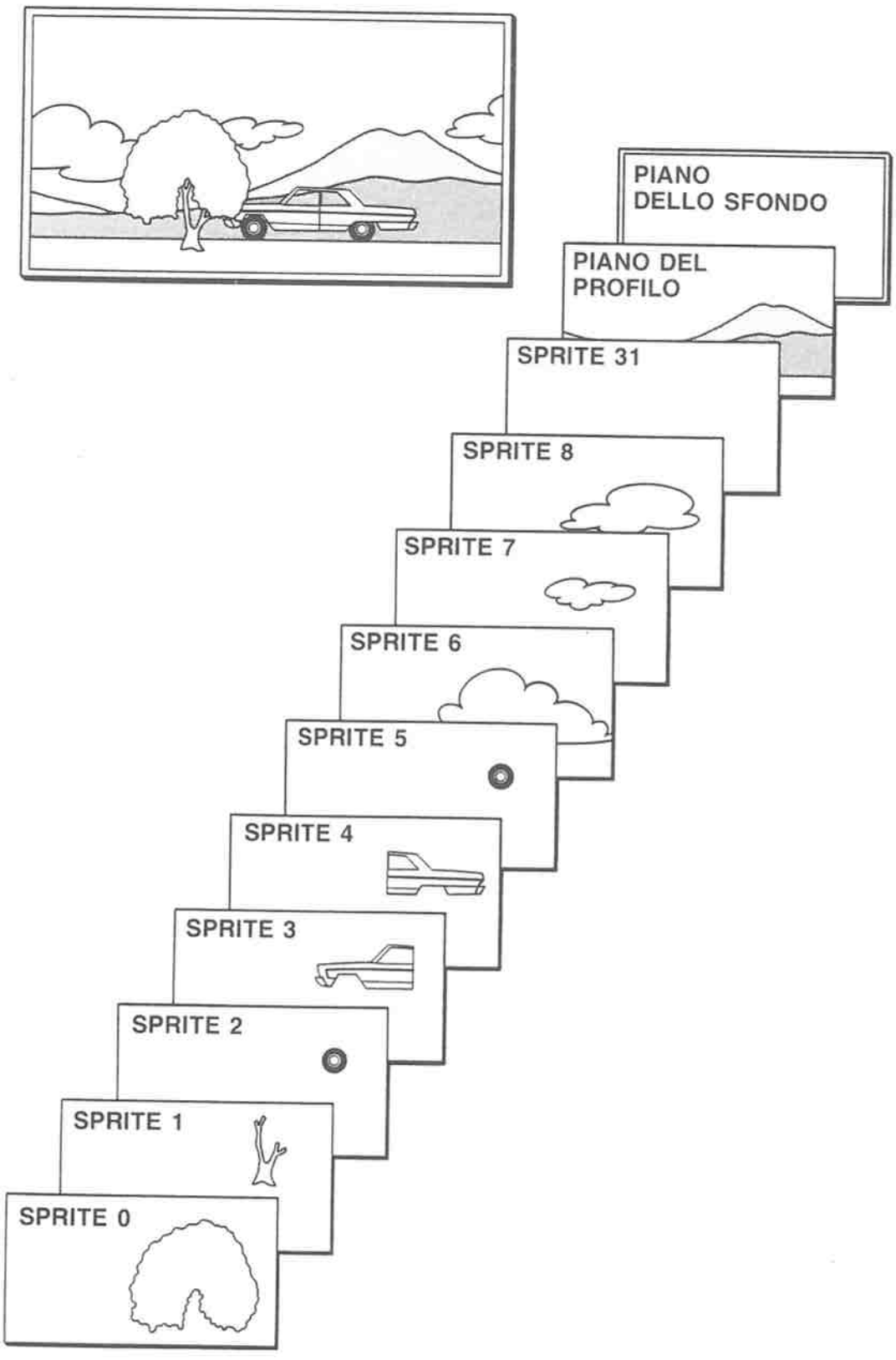
Nel byte posizione Y è possibile inserire un valore che va da 0 a 255. Questo numero determina la riga sullo schermo fornendo le seguenti informazioni:

0-191	è la riga dello schermo
208	significa che lo sprite non è più proiettato sullo schermo
224-255	rappresenta le righe dello schermo da —31 a —1. Rende possibile inserire uno sprite al disopra della riga superiore.

Nel byte posizione X può essere inserito un valore da 0 a 255. Questo numero determina la colonna sullo schermo. Se il bit 7 del byte del colore equivale a 1, la posizione di colonna dello sprite equivale al valore della posizione X meno 32. Se la posizione X è uguale a 0 e quando il bit 7 del byte del colore risulta essere 1, lo sprite verrà inserito nella colonna —32. Ciò rende possibile inserire uno sprite davanti alla prima colonna.

La posizione X e Y indicata si applica al punto dell'immagine posto all'estremità superiore sinistra dello sprite.

Lo sprite che è stato inserito per primo nella tabella degli attributi degli sprite ha la priorità più elevata (sprite 0).


PIANO
DELLO SFONDO
PIANO DEL
PROFILO
SPRITE 31
SPRITE 8
SPRITE 7
SPRITE 6
SPRITE 5
SPRITE 4
SPRITE 3
SPRITE 2
SPRITE 1
SPRITE 0

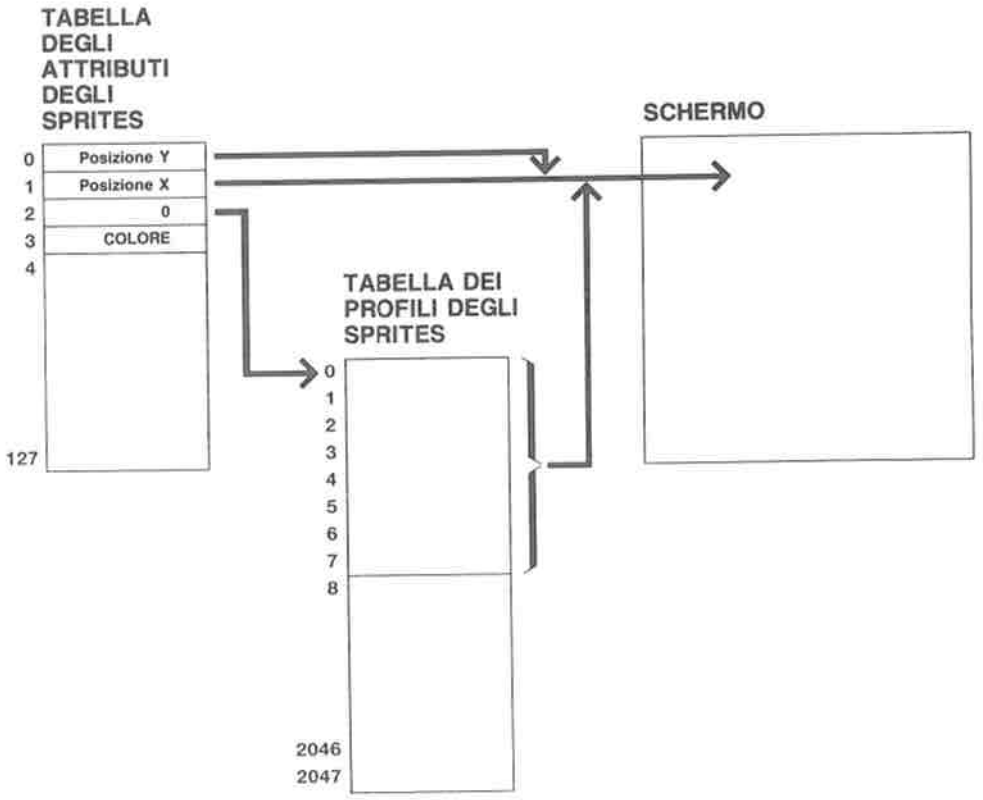
Nella illustrazione della pagina precedente è possibile vedere un'auto in un paesaggio:

L'albero è costituito dagli sprites 0 e 1
L'automobile dagli sprites da 2 a 5
Le nuvole dagli sprites 6, 7 e 8

Dato che gli sprites 0 e 1 hanno priorità rispetto agli sprites 2 e 5, l'albero appare davanti all'auto.
Il nome nell'entrata dello sprite della tabella degli attributi degli sprites è un numero che indica dove si può trovare il profilo dello sprite nella tabella dei profili.

Il primo indirizzo della tabella degli attributi degli sprites nella memoria RAM del video è contenuto nella variabile "BASE(8)" nel modo testo 2, nella variabile "BASE(13)" nel modo grafico 1 e nella variabile "BASE(18)" nel modo grafico 2.

Il primo indirizzo della tabella dei profili degli sprites è contenuto nella variabile "BASE(9)" nel modo testo 2, nella variabile "BASE(14)" nel modo grafico 1 e nella variabile "BASE(19)" nel modo grafico 2.



Cancellare la memoria con il comando "NEW" e provare il seguente esempio:

```
NEW
10 SCREEN 2:WIDTH 29
20 DATA 16,56,68,254,16,16,16,16
30 FOR I=0 TO 7
40 READ A:VPOKE(BASE(9)+I),A
50 NEXT I
60 PRINT:PRINT:PRINT
```

```
70 PRINT "UNO SPRITE MODO TESTO 2"
80 VPOKE (BASE(8)+0),40
90 VPOKE (BASE(8)+2),0
100 VPOKE (BASE(8)+3),1
110 FOR I=0 TO 255
120 VPOKE (BASE(8)+1),I
130 NEXT I
140 GOTO 110
RUN
```

Questo esempio mostra che è possibile usare gli sprites nel modo testo 2 comunicando direttamente con la memoria RAM del video.
Premere simultaneamente CTRL e STOP per interrompere il programma.
Successivamente battere:

```
SCREEN 0:WIDTH 40
```

Registri

Il processore video ha 9 registri a sua disposizione: i registri dal numero 0 al numero 7 ed il registro 8. Con MSX-BASIC il contenuto dei registri è caricato nelle variabili "VDP(n)", in cui "n" è un numero fra 0 e 8. Ad eccezione del registro 8, queste variabili possono essere cambiate. Il significato dei registri è il seguente:

Bit	0	1	2	3	4	5	6	7	Bit
Registro 0	0	0	0	0	0	0	A	D	
Registro 1	1	0	E	B	C	R	S	M	
Registro 2	0	0	0	0	Tabella dei nomi				
Registro 3	Tabella dei colori								
Registro 4	0	0	0	0	0	Tabella dei profili			
Registro 5	0	Tabella degli attributi degli sprites							
Registro 6	0	0	0	0	0	Tabella dei profili degli sprites			
Registro 7	Colore testo 1				Colore testo 2				
Registro 8	T	N	V	5° sprite					

ABC
000 = modo testo 2
100 = modo grafico 1
001 = modo grafico 2
010 = modo testo 1

D = 0 significa nessun input esterno per il processore video
D = 1 significa che c'è un input esterno

E = 0 significa nessuna interruzione per il processore video
E = 1 significa che c'è un'interruzione per il processore video

R = è stato riservato

S = 0 significa che gli sprites sono nel formato 8x8
S = 1 significa che gli sprites sono nel formato 16x16

M = 0 significa che gli sprites non sono ingranditi
M = 1 significa che gli sprites sono ingranditi

Tabella dei nomi = primo indirizzo della tabella dei nomi nella memoria RAM del video

Tabella dei colori = primo indirizzo della tabella dei colori nella memoria RAM del video

Tabella dei profili = primo indirizzo della tabella dei profili nella memoria RAM del video

Tabella degli attributi degli sprites = primo indirizzo della tabella degli attributi degli sprites nella memoria RAM del video

Tabella dei profili degli sprites = primo indirizzo nella tabella dei profili degli sprites nella memoria RAM del video

Colore testo 1 = colore di primo piano nel modo testo 1
Colore testo 2 = colore di sfondo nel modo testo 1

T Il flag di stato T nel registro 8 è impostato a 1 al termine della scansione totale d'immagine dell'ultima riga nel video attivo. È ripristinato a 0 dopo che è stato letto il registro di stato o il processore video è stato ripristinato esternamente.

N Il flag di stato N nel registro 8 è impostato a 1 se due o più sprites entrano in collisione. La collisione si verifica quando due sprites qualsiasi sullo schermo hanno un pixel che si sovrappone. Il flag N è azzerato dopo che è stato letto un registro da 0 a 8 o il processore video è stato ripristinato esternamente. Il registro 8 dovrebbe essere letto immediatamente dopo l'accensione per assicurare che il flag di collisione sia ripristinato. Il processore video controlla la coincidenza di ciascuna posizione di pixel durante la generazione dei pixel, indipendentemente dalla sua posizione sullo schermo. Ciò si verifica ogni cinquantesimo di secondo con TMS9929A. Pertanto, spostando gli sprites per più di una posizione di pixel durante questi intervalli, è possibile che gli sprites abbiano più pixel coincidenti o siano addirittura completamente sovrapposti quando il processore video controlla la coincidenza.

V e 5° sprite — Il flag di stato V nel registro 8 è impostato a 1 ogniqualvolta ci sono 5 o più sprites sulla riga orizzontale (righe da 0 a 192) ed il flag del blocco è uguale a 0. Il flag di stato V è azzerato dopo che è stato letto un registro 8 o dopo che il processore video è stato ripristinato esternamente. Il numero del quinto sprite è stato inserito nei 5 bit inferiori del registro 8 quando il flag V è impostato a valore logico 1 ed è valido ogniqualvolta il flag V è 1.

Le tabelle nella memoria RAM del video possono essere inserite in una posizione diversa nel programma dall'utente da quella assegnata da MSX-BASIC posto che vengano rispettate le regole seguenti:

1. La tabella dei nomi deve iniziare all'indirizzo 0 o ad un indirizzo corrispondente ad un multiplo di 1024.

2. La tabella dei colori deve iniziare all'indirizzo 0 o ad un indirizzo corrispondente ad un multiplo di 64.

3. La tabella dei profili deve iniziare all'indirizzo 0 o ad un indirizzo corrispondente ad un multiplo di 2048.

4. La tabella degli attributi degli sprites deve iniziare all'indirizzo 0 o ad un indirizzo corrispondente ad un multiplo di 128.

5. La tabella dei profili degli sprites deve iniziare all'indirizzo 0 o ad un indirizzo corrispondente ad un multiplo di 2048.

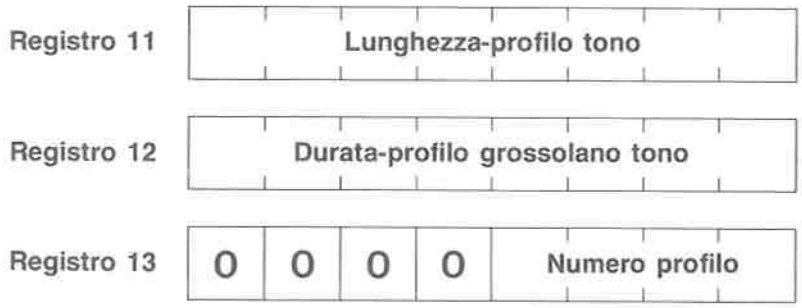
Appendice J

Il generatore di suoni programmabile

Il computer MSX è munito di un generatore di suoni programmabile (PSG) che produce il suono. A questo scopo, il PSG prevede 16 registri, cui può essere direttamente attribuito un valore con l'istruzione MSX-BASIC "SOUND". Contrariamente all'istruzione "PLAY" il suono continua fino a che il registro interessato non è riportato a 0 con un'altra istruzione "SOUND".

I registri hanno i seguenti significati:

	7	6	5	4	3	2	1	0	Bit
Registro 0	Canale tono numero A								
Registro 1	0	0	0	0	Tono grossolano A				
Registro 2	Tono numero B								
Registro 3	0	0	0	0	Tono grossolano B				
Registro 4	Canale tono numero C								
Registro 5	0	0	0	0	Tono grossolano C				
Registro 6	0	0	0	Tono di rumore					
Registro 7	0	0	RC	RB	RA	TC	TB	TA	
Registrto 8	0	0	0	M	Volume A				
Registro 9	0	0	0	M	Volume B				
Registro 10	0	0	0	M	Volume C				



L'altezza per i canali A, B e C è inserita nei registri da 0 a 5. Nel registro 7 si stabilisce a quale canale è destinato il tono o il rumore indicato, prodotto dal chip del suono. Se il registro 7 ha il contenuto binario di 00111110 significa che un tono è riprodotto soltanto attraverso il canale A.
Il volume per tutti i canali è inserito nei registri da 8 a 10. Il bit M con valore 1, corrisponde al volume massimo.
I registri 11 e 12 determinano la durata della forma d'onda o profilo, scelta nel registro 13.
Il numero del profilo nel registro 13 ha lo stesso significato dei subcomandi nell'istruzione "PLAY".
Il registro 14 è usato come uscita A di input/output ed il registro 15 come uscita B di input/output.

Numero del tono

Il tono per ciascun canale è determinato come segue:

$$TF = \frac{CF}{16 \times TF} \qquad CT + \frac{T}{256} = \frac{TP}{256}$$

TP = Periodo del tono
CF = Frequenza di orologio (= 1.78977 MHz)
TF = Frequenza del tono
T = Tono (registro 0, 2, 4)
CT = Tono grossolano (registro 1, 3, 5)

Esempio:
Un tono di 440 Hz per il canale A è determinato come segue:

$$TP = \frac{1.78977 \times 10^{6}}{16 \times 440} = 254 \qquad CT + \frac{T}{256} = \frac{254}{256}$$

$$CT = 0 \qquad T = 254$$

Registro 0 = 254 e registro 1 = 0

Tono del rumore

Il rumore è determinato come segue:

$$NP = \frac{CF}{16 \times NF} \qquad NT = NP$$

NP = Periodo del rumore

CF = Frequenza di orologio (= 1.78977 MHz)

NF = Frequenza del rumore

NT = Tono del rumore (registro 6)

Esempio:
Un tono di rumore di 30 kHz è determinato come segue:

$$NP = \frac{1.78977 \times 10^{6}}{16 \times (30 \times 10^{3})} = 4 \qquad NT = 4$$

Registro 5 = 4

Durata del profilo

La durata del profilo è determinata come segue:

$$SP = \frac{CF}{256 \times SF} \qquad CT + \frac{T}{256} = \frac{SP}{256}$$

SP = Periodo del profilo

CF = Frequenza di orologio (= 1.78977 MHz)

SF = Frequenza del profilo

T = Durata del profilo del tono (registro 11)

CT = Durata del profilo del tono grossolano (registro 12)

Esempio:
Una frequenza di profilo di 0,5 Hz è determinata come segue:

$$SP = \frac{1.78977 \times 10^{6}}{256 \times 0.5} = 14000 \qquad CT + \frac{T}{256} = \frac{14000}{256}$$

$$CT = 54 \qquad T = 176$$

Registro 11 = 176 e registro 12 = 54

Registro 11 = 240 e registro 12 = 60

Cancellare la memoria con il comando "NEW" e battere il seguente esempio:

```
10 FOR I=0 TO 13
20 SOUND I,0
30 NEXT I
```

```
40 SOUND 7,&H3E
50 SOUND 8,&H0F
60 FOR I=&H10 TO &HF0
70 SOUND 0,I
80 NEXT I
90 SOUND 6,&H0F
100 SOUND 7,&H07
110 SOUND 8,&H10
120 SOUND 9,&H10
130 SOUND 10,&H10
140 SOUND 12,&H40
150 SOUND 13,&H00
RUN
```

Nelle righe da 10 a 30 tutti i registri sono portati a zero in un'iterazione "FOR...NEXT".
Nella riga 40 è ammesso soltanto un tono di canale. Nella riga 50 viene determinato il volume del canale A.
Nelle iterazioni "FOR...NEXT" delle righe da 60 a 80, vengono inseriti i toni successivi per il canale A.
Nella riga 90 viene determinato il tono di rumore.
Nella riga 100 si indica che il rumore deve essere riprodotto sui canali A, B e C.
Nelle righe da 110 a 130 viene regolato il volume massimo per tutti e tre i canali.
Nella riga 140 si determina la durata del profilo.
Nella riga 150 si determina il profilo del rumore.

Il mare

Il seguente esempio riproduce il rumore del mare:

```
NEW
10 FOR I=0 TO 13
20 SOUND I,0
30 NEXT I
40 SOUND 6,&H10
50 SOUND 7,&H07
60 SOUND 8,&H10
70 SOUND 9,&H10
80 SOUND 10,&H10
90 SOUND 12,&H40
100 SOUND 13,H0E
RUN
```

Tutti i suoni che si possono ascoltare sono prodotti dal chip del suono.
Dopo aver eseguito il programma, MSX-BASIC è a livello comandi mentre il suono continua e si interrompe soltanto quando i registri 6 e 13 vengono portati a zero oppure quando si premono simultaneamente i tasti CTRL e STOP.

Phonola

Scan by Agnul75